Towards Robot Skill Learning: From Simple Skills to Table Tennis

Jan Peters[1,2], Jens Kober[1,2], Katharina Mülling[1,2],
Oliver Krämer[1,2], Gerhard Neumann[2]

[1]Technische Universität Darmstadt, 64293 Darmstadt, Germany
mail@jan-peters.net, muelling@tuebingen.mpg.de, je.kober@gmail.com,
oli@robot-learning.de
[2]Max Planck Institute for Intelligent Systems, 72076 Tübingen, Germany
neumann@ias.tu-darmstadt.de

Abstract. Learning robots that can acquire new motor skills and refine existing ones have been a long-standing vision of both robotics, and machine learning. However, off-the-shelf machine learning appears not to be adequate for robot skill learning, as it neither scales to anthropomorphic robotics nor do fulfills the crucial real-time requirements. As an alternative, we propose to divide the generic skill learning problem into parts that can be well-understood from a robotics point of view. In this context, we have developed machine learning methods applicable to robot skill learning. This paper discusses recent progress ranging from simple skill learning problems to a game of robot table tennis.

1 Introduction

Despite the many impressive motor skills exhibited by anthropomorphic robots, the generation of motor behaviors has changed little since classical robotics. The roboticist models the task dynamics as accurately as possible while using human insight 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. Such highly engineered approaches are feasible in highly structured industrial or research environments. However, for robots to leave factory floors and research environments, the strong reliance on hand-crafted models of the environment and the robots needs to be reduced. Instead, a general framework is needed for allowing robots to learn their tasks with minimal programming and in less structured, uncertain environments. Such an approach clearly has to be based on machine learning *combined* with robotics insights to make the high-dimensional domain of anthropomorphic robots accessible. To accomplish this aim, three major questions need to be addressed:

1. How can we develop efficient motor learning methods?
2. How can anthropomorphic robots learn basic skills similar to humans?
3. Can complex skills be composed with these elements?

In the next sections, we will address these questions. We focus on model-free methods, which do not maintain an internal behavior simulator (i.e., a forward model) but operate directly on the data. Note that most methods transfer straightforwardly to model-based approaches.

2 Motor Learning Methods

After formalizing the necessary assumptions on robotics from a machine learning perspective, we show the concepts behind the our learning methods.

2.1 Modeling Assumptions.

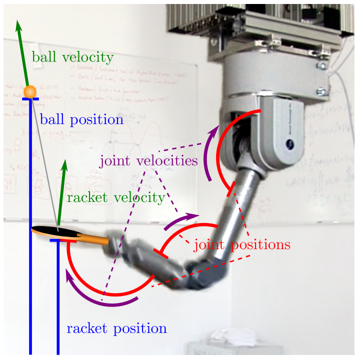

Fig. 1. Modeling of the learning task of paddling a ball.

For addressing these questions, we focus on anthropomorphic robot systems which always are in a state $\mathbf{x} \in \Re^n$ that includes both the internal state of the robot (e.g., joint angles, velocities, acceleration in Fig. 1, but also internal variables) as well as external state variables (e.g., ball position and velocity), and execute motor commands $\mathbf{u} \in \Re^m$ at a high frequency (usually 500–1000Hz). The actions are taken in accordance to a parametrized, stationary, stochastic policy, i.e., a set of rules with exploration $\mathbf{u} \sim \pi_\theta(\mathbf{u}|\mathbf{x}) = p(\mathbf{u}|\mathbf{x}, \boldsymbol{\theta})$ where the parameters $\boldsymbol{\theta} \in \Re^N$ allow for learning. The stochasticity in the policy allows capturing the variance of the teacher, can ease algorithm design, and there exist well-known problems where the optimal stationary policy is stochastic. Frequently used policies are linear in state feature $\boldsymbol{\phi}(\mathbf{x})$ and have Gaussian exploration, i.e., $\pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{x}) = \mathcal{N}(\mathbf{u}|\boldsymbol{\phi}^T(\mathbf{x})\boldsymbol{\theta}, \sigma^2)$. After every motor command, the system transfers to a next state $\mathbf{x}' \sim p(\mathbf{x}'|\mathbf{x}, \mathbf{u})$, and receives a learning signal $r(\mathbf{x}, \mathbf{u})$. The learning signal can be a general reward (i.e., in full reinforcement learning), but can also contain substantially more structure (e.g., prediction errors in model learning or proximity to a demonstration in imitation), see [1].

During experiments, the system obtains a stream of data consisting of episodes $\boldsymbol{\tau} = [\mathbf{x}_1, \mathbf{u}_1, \mathbf{x}_2, \mathbf{u}_2, \ldots, \mathbf{x}_{T-1}, \mathbf{u}_{T-1}, \mathbf{x}_T]$ of length T, often also called trajectories or paths. These paths are obviously distributed according to

$$p_{\boldsymbol{\theta}}(\boldsymbol{\tau}) = p(\mathbf{x}_1)\textstyle\prod_{t=1}^{T} p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t)\pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{x}), \tag{1}$$

where $p(\mathbf{x}_1)$ denotes the start-state distribution. We will refer to the distribution of teacher's demonstrations or past data $p(\boldsymbol{\tau})$ by simply omitting $\boldsymbol{\theta}$. The rewards of a path can be formulated as a weighted sum of immediate rewards $r(\boldsymbol{\tau}) = \sum_{t=1}^{T} a_t r(\mathbf{x}_t, \mathbf{u}_t)$. Most motor skill learning problems can be phrased as optimizing the expected returns $J(\boldsymbol{\theta}) = E_{\boldsymbol{\theta}}\{r(\boldsymbol{\tau})\} = \int p_{\boldsymbol{\theta}}(\boldsymbol{\tau}) r(\boldsymbol{\tau}) d\boldsymbol{\tau}$.

2.2 Method Development Approach

The problem of learning robot motor skills can be modeled as follows: (1) The robots starts with an initial training data set obtained from demonstrations from which it learns an initial policy. (2) It subsequently learns how to improve this policy by repetitive training over multiple episodes. The first goal is accomplished by imitation learning while the second requires reinforcement learning. In addition, model learning is often needed for improved execution [2].

Imitation Learning. The goal of imitation learning is to successfully reproduce the policy of the teacher $\pi(\mathbf{u}|\mathbf{x})$. Many approaches exist in the literature [3, 4]. However, this problem can be well-understood for stochastic policies: How can we reproduce the stochastic policy π given a demonstrated path distribution $p(\boldsymbol{\tau})$? The path distribution $p_{\boldsymbol{\theta}}(\boldsymbol{\tau})$ generated by the policy $\pi_{\boldsymbol{\theta}}$ should be as close as possible to the teacher's, i.e., it minimizes the Kullback-Leibler Divergence

$$D(p(\boldsymbol{\tau})||p_{\boldsymbol{\theta}}(\boldsymbol{\tau})) = \int p(\tau)\log\frac{p(\boldsymbol{\tau})}{p_{\boldsymbol{\theta}}(\boldsymbol{\tau})}d\tau = \int p(\boldsymbol{\tau})\textstyle\sum_{t=1}^{T}\log\dfrac{\pi(\mathbf{u}_t|\mathbf{x}_t)}{\pi_{\boldsymbol{\theta}}(\mathbf{u}_t|\mathbf{x}_t)}d\boldsymbol{\tau},$$

where the model of the system and the start-state distribution naturally cancel out. As $\log\pi(\mathbf{u}_t|\mathbf{x}_t)$ is an additive constant, the path rewards become

$$r(\boldsymbol{\tau}) \propto -\textstyle\sum_{t=1}^{T}\log\pi_{\boldsymbol{\theta}}(\mathbf{u}_t|\mathbf{x}_t) = -\textstyle\sum_{t=1}^{T}\left\|\mathbf{u} - \boldsymbol{\phi}^T(\mathbf{x})\boldsymbol{\theta}\right\|^2,$$

where the second part only holds true for our policy which is linear in the features and has Gaussian exploration. Clearly, the model-free imitation learning problem can be solved in one shot in this way [4].

Reinforcement Learning. For general rewards, the problem is not straightforward as the expected return has no notion of data. Instead, for such a brute-force problem, learning can only happen indirectly as in value function methods [1] or using small steps in the policy space, as in policy gradient methods [5]. Instead of circumventing this problem, we realized that there exits a tight lower bound

$$J(\boldsymbol{\theta}) = \int p_{\boldsymbol{\theta}}(\boldsymbol{\tau})r(\boldsymbol{\tau})d\boldsymbol{\tau} \geq D(p(\boldsymbol{\tau})r(\boldsymbol{\tau})||p_{\boldsymbol{\theta}}(\boldsymbol{\tau})).$$

Hence, reinforcement learning becomes a series of reward-weighted self-imitation steps (Intuitively: "*Do what you are but better*") with the resulting policy update

$$\boldsymbol{\theta}' = \text{argmax}_{\boldsymbol{\theta}'} D(R(\boldsymbol{\tau})p_{\boldsymbol{\theta}}(\boldsymbol{\tau})||p_{\boldsymbol{\theta}'}(\boldsymbol{\tau}))$$

which is guaranteed to converge to a local optimum. Taking such an approach, which stays close to the training data is often crucial for robot reinforcement learning as the robot avoids trying arbitrary, potentially destructive actions. The resulting methods have led to a series of highly successful robot reinforcement learning methods such as reward-weighted regression [5], LAWER [6], PoWER [4], and Cost-regularized Kernel Regression [7].

3 Application in Robot Skill Learning

The imitation and reinforcement learning approaches have so far been general, despite being geared for the robotics scenario. To apply these methods in robotics, we need appropriate policy representations. Such representation are needed both for simple and complex tasks.

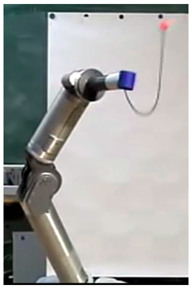

Fig. 2. Swing the ball into the cup

3.1 Learning Simple Tasks with Motor Primitives

We chose policy features based on dynamical systems, which are an extension the ground-breaking work of Ijspeert, Nakanishi & Schaal refined in [4]. We will use these features to represent elementary movements, or Movement Primitives (MP). The methods above are straightforward to apply by using a single motor primitive as a parametrized policy. Such elementary policies $\pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{x})$ have both shape parameters $\boldsymbol{w}$ as well as task parameters $\boldsymbol{\gamma}$ where $\boldsymbol{\theta} = [\boldsymbol{w}, \boldsymbol{\gamma}]$. For example, an elementary policy can be used to learn a dart throwing movement by learning the shape parameters $\boldsymbol{w}$ without considering the task parameters $\boldsymbol{\gamma}$. However, when playing a dart game (e.g., around the clock), the robot has to adapt the elementary policy (which represents the throwing movement) to new fields on the dart board. In this case, the shape parameters $\boldsymbol{w}$ can be kept at fixed value and the goal-adaptation happens purely through the task parameters $\boldsymbol{\gamma}$.

Learning only the shape parameters of rhythmic motor primitives using just imitation learning, we have been able to learn ball paddling [4] as shown in Fig. 1. Using the combination of imitation and reinforcement learning, our robot managed to learn ball-in-a-cup in Fig. 2 to perfection within less than a hundred trials using only shape parameters [4]. By learning dart throwing with the shape parameters, and, subsequently, adapting the dart throwing movement to the context, we have managed to learn dart games based on context as well as another, black-jack-style sequential throwing game [7]. The latter two have been accomplished by learning a task parameter policy $\boldsymbol{\gamma} \sim \hat{\pi}(\boldsymbol{\gamma}|\mathbf{x})$.

3.2 Learning a Complex Task with Many Motor Primitives

When single primitives no longer suffice, a robot learning system does not only need context but also multiple motor primitives, as for example, in *robot table tennis*, see Fig. 3. A combination of primitives allows the robot to deal with many situations where only few primitives are activated in the same context [8]. The new policy combines multiple primitives as follows

$$\mathbf{u} \sim \pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{x}) = \textstyle\sum_{i=1}^{K} \pi_{\boldsymbol{\theta}_0}(i|\mathbf{x}) \pi_{\boldsymbol{\theta}_i}(\mathbf{u}|\mathbf{x}).$$

The policy $\pi_{\boldsymbol{\theta}_0}(i|\mathbf{x})$ represents the probability of selecting primitive i, represented by $\pi_{\boldsymbol{\theta}_i}(\mathbf{u}|\mathbf{x})$, based on the incoming ball and the opponent's position. The resulting system learned to return 69% of all balls after imitation learning, and could self-improve against a ball gun to up to 94% successful returns.

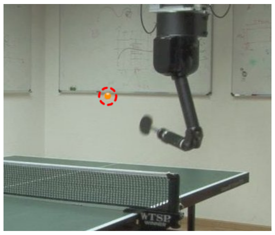

Fig. 3. Learning to Play Robot Table Tennis

4 Conclusion

In this paper, we reviewed the imitation and reinforcement learning methods used to learn a large variety of motor skills. These range from simple tasks, such as ball-paddling, ball-in-a-cup, dart games, etc up to playing robot table tennis.

References

1. Kober, J., Bagnell, D., Peters, J.: Reinforcement learning in robotics: A survey. International Journal of Robotics Research (IJRR) (2013)
2. Nguyen-Tuong, D., Peters, J.: Model learning in robot control: a survey. Cognitive Processing (4) (2011)
3. Argall, B., Chernova, S., Veloso, M., Browning, B.: A survey of robot learning from demonstration. Robotics and Autonomous Systems (2009)
4. Kober, J., Peters, J.: Imitation and reinforcement learning. IEEE Robotics and Automation Magazine (2010)
5. Peters, J.: Machine Learning of Motor Skills for Robotics. PhD thesis (2007)
6. Neumann, G., Peters, J.: Fitted Q-iteration by Advantage Weighted Regression. In: Advances in Neural Information Processing Systems 22 (NIPS). (2009)
7. Kober, J., Wilhelm, A., Oztop, E., Peters, J.: Reinforcement learning to adjust parametrized motor primitives to new situations. Autonomous Robots (2012)
8. Mülling, K., Kober, J., Krömer, O., Peters, J.: Learning to select and generalize striking movements in robot table tennis. IJRR (2013)