Imitation and Reinforcement Learning for Motor Primitives with Perceptual Coupling

Jens Kober, Betty Mohler, Jan Peters

Abstract Traditional motor primitive approaches deal largely with open-loop policies which can only deal with small perturbations. In this paper, we present a new type of motor primitive policies which serve as closed-loop policies together with an appropriate learning algorithm. Our new motor primitives are an augmented version version of the dynamical system-based motor primitives [6] that incorporates perceptual coupling to external variables. We show that these motor primitives can perform complex tasks such as Ball-in-a-Cup or Kendama task even with large variances in the initial conditions where a skilled human player would be challenged. We initialize the open-loop policies by imitation learning and the perceptual coupling with a handcrafted solution. We first improve the open-loop policies and subsequently the perceptual coupling using a novel reinforcement learning method which is particularly well-suited for dynamical system-based motor primitives.

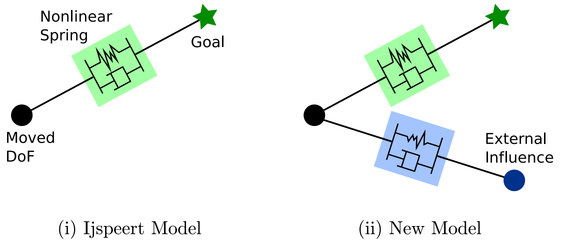
1 Introduction

The recent introduction of motor primitives based on dynamical systems [6,7,21,22] have allowed both imitation learning and Reinforcement Learning to acquire new behaviors fast and reliably. Resulting successes have shown that it is possible to rapidly learn motor primitives for complex behaviors such as tennis swings [6, 7], T-ball batting [14], drumming [16], biped locomotion [13, 22] and even in tasks with potential industrial application [26]. However, in their current form these motor primitives are generated in such a way that they are either only coupled to internal variables [6, 7] or only include manually tuned phase-locking, e.g., with an external beat [16] or between the gait-generating primitive and the contact time of the feet [13, 22]. Furthermore, they incorporate the possibility to update parameters

Jens Kober · Betty Mohler · Jan Peters
Max Planck Institute for Biological Cybernetics, Tübingen, Germany
e-mail: {kober,mohler,jrpeters}@tuebingen.mpg.de

of a movement in real-time thus enabling perceptual coupling. E.g., changing the the goal of a movement can couple it to a target, i.e., an external variable. However, this perceptual coupling only is effective for the end of the movement and the rest of the movement is not coupled to the external variable. 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. If changes of the internal variables constantly influences the behavior of the external variable more complex perceptual coupling is required as the coupling needs to incorporate knowledge of the behavior of the external variable. In this paper, it is our goal to augment the Ijspeert-Nakanishi-Schaal approach [6, 7] 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., [29]), we assume that there is an object of internal focus described by a state x and one of external focus 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 clearly 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 y and x can be in completely different spaces.

Integrating the 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 [21]. However, learning perceptual coupling to an external variable is not as straightforward. It requires many trials in 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 is analogous to the case of a human learning tennis: a teacher can show a forehand but a lot of self-practice is needed for a proper tennis game.



(i) Ijspeert Model

(ii) New Model

Fig. 1 Illustration of the behavior of the motor primitives (i) and the augmented motor primitives (ii).

2 Augmented Motor Primitives with Perceptual Coupling

There are several frameworks for motor primitives used in robotics (e.g., [9]). In this section, we first introduce the general idea behind dynamic system motor primitives as suggested in [6, 7] and, subsequently, show how perceptual coupling can be introduced. Subsequently, we show how the perceptual coupling can be realized by augmenting the acceleration-based framework from [21].

2.1 Perceptual Coupling for Motor Primitives

The basic idea in the original work of Ijspeert, Nakanishi and Schaal [6,7] is that motor primitives can be parted into two components, i.e., a canonical system h which drives transformed systems g_k for every considered degree of freedom k. As a result, we have a system of differential equations given by

$$\dot{z} = h(z), \tag{1}$$

$$\dot{\mathrm{x}} = \mathrm{g}(\mathrm{x}, z, \mathrm{w}), \tag{2}$$

which determines the variables of internal focus x (e.g., Cartesian or joint positions). Here, z denotes the state of the canonical system, which is indicates the current phase of the movement, and w the internal parameters for transforming the output of the canonical system. The schematic in Figure 2 illustrates this traditional setup in black. In Section 2.2, we will discuss good choices for these dynamical systems as well as their coupling based on the most current formulation [21].

When taking an external variable y into account, there are three different ways how this variable influences the motor primitive system which one can consider, i.e., (i) it could only influence Eq.(1) which would be appropriate for synchronization problems and phase-locking (similar as in [12,16]), (ii) only affect Eq.(2) which allows the continuous modification of the current state of the system by another variable and (iii) the combination of (i) and (ii). While (i) and (iii) are the right solution if phase-locking or synchronization are needed, the coupling in the canonical system will destroy many of the nice properties of the system and make it prohibitively hard to learn in practice. Furthermore, as we focus on discrete movements in this paper, we focus on the case (ii) which has not been used to date. In this case, we have a modified dynamical system

$$\dot{z} = h(z), \tag{3}$$
$$\dot{\mathrm{x}} = \hat{\mathrm{g}}(\mathrm{x}, \mathrm{y}, \bar{\mathrm{y}}, z, \mathrm{v}), \tag{4}$$
$$\dot{\bar{\mathrm{y}}} = \bar{\mathrm{g}}(\bar{\mathrm{y}}, z, \mathrm{w}), \tag{5}$$

where y denotes the state of the external variable, $\bar{\mathrm{y}}$ the expected state of the external variable and $\dot{\bar{\mathrm{y}}}$ 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 one in [21] while allowing to couple the external variables into the system.

2.2 *Realization for Discrete Movements*

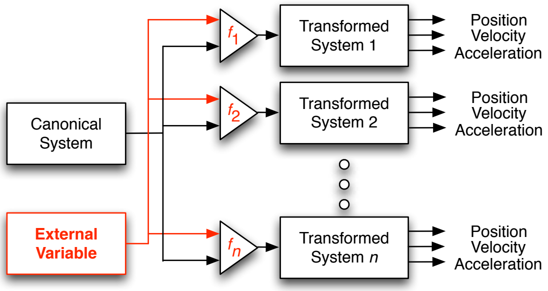


Fig. 2 General schematic illustrating both the original motor primitive framework by [7, 21] in black and the augmentation for perceptual coupling in red.

The original formulation in [6, 7] was a major breakthrough as the right choice of the dynamical systems in Equations (1, 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), fast learning from a teachers presentation is possible. In this section, we first discuss how the most current formulation from the motor primitives as discussed in [21] is instantiated from Section 2.1. Subsequently, we show how it can be augmented in order to incorporate perceptual coupling.

While the original formulation in [6, 7] used a second-order canonical system, it has since then been shown that a single first order system suffices [21], i.e., we have

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

which represents the phase of the trajectory. It has a time constant $\tau = \frac{1}{T}$ (where T is the movement duration) and a parameter α_h which is chosen such that $z \approx 0$ at T thus ensuring that the influence of the transformation function (8) vanishes. 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 $2k$-th component of 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 g in the following form

$$\dot{x}_{2k+1} = \tau \alpha_g \left(\beta_g \left(t_k - x_{2k}\right) - x_{2k+1}\right) + \tau \left(\left(t_k - x_{2k}^0\right) + a_k\right) f_k, \tag{6}$$

$$\dot{x}_{2k} = \tau x_{2k+1}. \tag{7}$$

This function has the same time constant τ as the canonical system, parameters α_g, β_g set such that the system is critically damped, a goal parameter t_k corresponding to the final position of x_{2k}, the initial position x_{2k}^0, an amplitude modifier a_k which can be set arbitrarily, and a transformation function f_k. This transformation function transforms the output of the canonical system so that the transformed system can represent complex nonlinear patterns and is given by

$$f_k(z) = \sum_{i=1}^{N} \psi_i(z) w_i z, \tag{8}$$

where w are adjustable parameters and uses normalized Gaussian kernels without scaling such as

$$\psi_i = \frac{\exp\left(-h_i \left(z - c_i\right)^2\right)}{\sum_{j=1}^{N} \exp\left(-h_j \left(z - c_j\right)^2\right)} \tag{9}$$

for localizing the interaction in phase space where we have centers c_i and width h_i.

In order to learn a motor primitive with perceptual coupling, we need two components. First, we need to learn the normal or average behavior $\bar{\mathrm{y}}$ of the variable of external focus y (e.g., the relative positions of an object) which can be represented by a single motor primitive $\bar{\mathrm{g}}$, i.e., we can use the same type of function from Equations (2, 5) for $\bar{\mathrm{g}}$ which are learned based on the same z and given by Equations (6, 7). Additionally, we have the system $\hat{\mathrm{g}}$ for the variable of internal focus x which determines our actual movements which incorporates the inputs of the normal behavior $\bar{\mathrm{y}}$ as well as the current state y of the external variable. We obtain the system $\hat{\mathrm{g}}$ by inserting a modified coupling function $\hat{\mathrm{f}}(z, \mathrm{y}, \bar{\mathrm{y}})$ instead of the original $\mathrm{f}(z)$ in g. Function $\mathrm{f}(z)$ is modified in order to include perceptual coupling to y and we obtain

$$\hat{f}_k(z, \mathrm{y}, \bar{\mathrm{y}}) = \sum_{i=1}^{N} \psi_i(z) \hat{w}_i z + \sum_{j=1}^{M} \hat{\psi}_j(z) \left(\kappa_{jk}^{\mathrm{T}}(\mathrm{y} - \bar{\mathrm{y}}) + \delta_{jk}^{\mathrm{T}}(\dot{\mathrm{y}} - \dot{\bar{\mathrm{y}}})\right), \tag{10}$$

where $\hat{\psi}_j(z)$ denote Gaussian kernels as in Equation (9) with centers $\hat{c}_j$ and width $\hat{h}_j$. Note, that it can be useful to set $N > M$ for reducing the number of parameters. All parameters are given by $\mathrm{v} = [\hat{\mathrm{w}}, \kappa, \delta]$. Here, $\hat{\mathrm{w}}$ are just the standard transformation parameters while κ_{jk} and δ_{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 simplify to the original approach. However, in the noisy, imperfect case, this perceptual coupling can ensure success even in extreme cases.

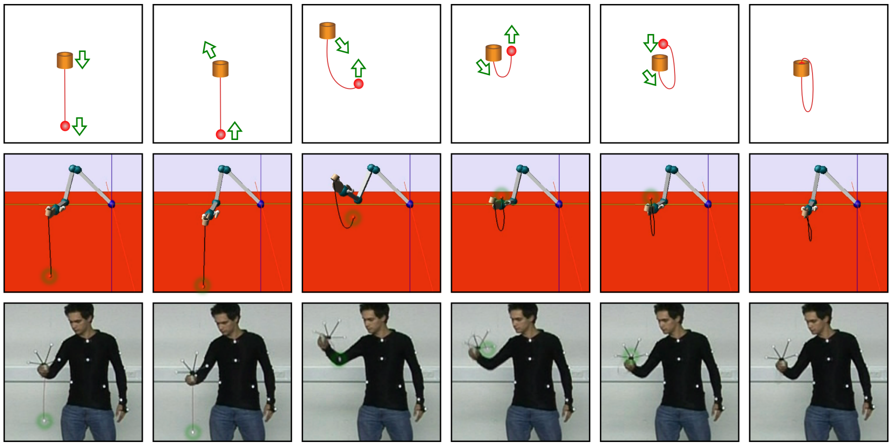


Fig. 3 This figure shows schematic drawings of the Ball-in-a-Cup motion, the final learned robot motion as well as a motion-captured human motion. The green arrows show the directions of the momentary movements. The human cup motion was taught to the robot by imitation learning with 91 parameters for 1.5 seconds. Please also refer to the video on the first author's website.

3 Learning for Perceptually Coupled Motor Primitives

While the transformation function $f_k(z)$ (8) can be learned from few or even just a single trial, this simplicity no longer transfers to learning the new function $\hat{f}_k(z, \mathrm{y}, \bar{\mathrm{y}})$ (10) as perceptual coupling requires that the coupling to an uncertain external variable is learned. 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. As an alternative, we could follow "Nature as our teacher", and create a concerted approach of imitation and self-improvement by trial-and-error. For doing so, we first have a teacher who presents several trials and, subsequently, we improve our behavior by reinforcement learning.

3.1 Imitation Learning with Perceptual Coupling

Imitation learning is applied to a large number of problems in robotics (e.g., [4, 5, 17]). Here we can largely follow the original work in [6, 7, 21] and only need minor modifications. 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 [6, 7], we need a bootstrapping step as we determine first the parameters for the system described by Equation (5) and, subsequently, use the learned results in the learning of the system in Equation (4). These steps can be performed efficiently in the context of dynamical systems motor primitives as the transformation functions (8) of Equations (4) and (5) are linear in parameters. As a result, we can choose the weighted squared error

$$\varepsilon_m^2 = \sum_{i=1}^{n} \psi_i^m \left(f_i^{\mathrm{ref}} - \mathrm{z}_i^{\mathrm{T}} \mathrm{w}^m \right)^2 \tag{11}$$

as cost function and minimize it for all parameter vectors w^m with $m \in \{1, 2, \ldots, M\}$. Here, the corresponding weighting function are denoted by ψ_i^m and the basis functions by $\mathrm{z}_i^{\mathrm{T}}$. The reference or target signal f_i^{ref} is the desired transformation function and $i \in \{1, 2, \ldots, n\}$ indicates the number of the sample. The error in Equation (11) can be rewritten as

$$\varepsilon_m^2 = \left(\mathrm{f}^{\mathrm{ref}} - \mathrm{Z}\mathrm{w}^m \right)^{\mathrm{T}} \Psi \left(\mathrm{f}^{\mathrm{ref}} - \mathrm{Z}\mathrm{w}^m \right) \tag{12}$$

with $\mathrm{f}^{\mathrm{ref}}$ giving the value of f_i^{ref} for all samples i, $\Psi = \mathrm{diag}\left(\psi_i^m, \ldots, \psi_n^m\right)$ and $\mathrm{Z}_i = \mathrm{z}_i^{\mathrm{T}}$. As a result, we have a standard locally-weighted linear regression problem that can be solved straightforwardly and yields the unbiased estimator

$$\mathrm{w}^m = \left(\mathrm{Z}^{\mathrm{T}} \Psi \mathrm{Z} \right)^{-1} \mathrm{Z}^{\mathrm{T}} \Psi \mathrm{f}^{\mathrm{ref}}. \tag{13}$$

This general approach has originally been suggested in [7]. Estimating the parameters of the dynamical system is slightly more daunting, i.e., the movement duration is extracted using motion detection (velocities are zero at the start and at the end) and the time-constant is set accordingly.

This local regression yields good values for the parameters of $f_k(z)$. Subsequently, we can perform the exact same step for $\hat{f}_k(z, \mathrm{y}, \bar{\mathrm{y}})$ 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 w and $\hat{\mathrm{w}}$, the parameters κ and δ cannot be learned by imitation as these require deviation from the nominal behavior for the external variable.

However, as discussed before, pure imitation for 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.

3.2 Reinforcement Learning for Perceptually Coupled Motor Primitives

Reinforcement learning [24] is widely used in robotics (e.g., [18]) but reinforcement learning of discrete motor primitives is a very specific type of learning problem where it is hard to apply generic reinforcement learning algorithms [14, 15]. For this reason, the focus of this paper is largely on domain-appropriate reinforcement learning algorithms which operate on parametrized policies for episodic control problems.

3.2.1 Reinforcement Learning Setup

When modeling our problem as a reinforcement learning problem, we always have a state $\mathrm{s} = [z, \mathrm{y}, \bar{\mathrm{y}}, \mathrm{x}]$ with high dimensions (as a result, standard RL methods which discretize the state-space can no longer be applied), and the action $\mathrm{a} = [\mathrm{f}(z) + \varepsilon, \hat{\mathrm{f}}(z, \mathrm{y}, \bar{\mathrm{y}}) + \hat{\varepsilon}]$ is the output of our motor primitives. Here, the exploration is denoted by ε and $\hat{\varepsilon}$, and we can give a stochastic policy $\mathrm{a} \sim \pi(\mathrm{s})$ as distribution over the states with parameters $\theta = [\mathrm{w}, \mathrm{v}] \in \mathbb{R}^n$. After a next time-step δt, the actor transfers to a state s_{t+1} and receives a reward r_t. As we are interested in learning complex motor tasks consisting of a single stroke [21, 29], we focus on finite horizons of length T with episodic restarts [24] and learn the optimal parametrized policy for such problems. The general goal in reinforcement learning is to optimize the *expected return* of the policy with parameters θ defined by

$$J(\theta) = \int_{\mathbb{T}} p(\tau) R(\tau) d\tau, \tag{14}$$

where $\tau = [\mathrm{s}_{1:T+1}, \mathrm{a}_{1:T}]$ denotes a sequence of states $\mathrm{s}_{1:T+1} = [\mathrm{s}_1,\ \mathrm{s}_2,\ \ldots,\ \mathrm{s}_{T+1}]$ and actions $\mathrm{a}_{1:T} = [\mathrm{a}_1,\ \mathrm{a}_2,\ \ldots,\ \mathrm{a}_T]$, the probability of an episode τ is denoted by $p(\tau)$ and $R(\tau)$ refers to the return of an episode τ. Using Markov assumption, we can write the path distribution as $p(\tau) = p(\mathrm{x}_1) \prod_{t=1}^{T+1} p(\mathrm{s}_{t+1} | \mathrm{s}_t, \mathrm{a}_t) \pi(\mathrm{a}_t | \mathrm{s}_t, t)$ where $p(\mathrm{s}_1)$ denotes the initial state distribution and $p(\mathrm{s}_{t+1} | \mathrm{s}_t, \mathrm{a}_t)$ is the next state distribution conditioned on last state and action. Similarly, if we assume additive, accumulated rewards, the return of a path is given by $R(\tau) = \frac{1}{T} \sum_{t=1}^{T} r(\mathrm{s}_t, \mathrm{a}_t, \mathrm{s}_{t+1}, t)$, where $r(\mathrm{s}_t, \mathrm{a}_t, \mathrm{s}_{t+1}, t)$ denotes the immediate reward.

While episodic Reinforcement Learning (RL) problems with finite horizons are common in motor control, few methods exist in the RL literature (c.f., model-free method such as Episodic REINFORCE [28] and the Episodic Natural Actor-Critic eNAC [14] as well as model-based methods, e.g., using differential-dynamic programming [2]). In order to avoid learning of complex models, we focus on model-free methods and, to reduce the number of open parameters, we rather use a novel Reinforcement Learning algorithm which is based on expectation-maximization. Our new algorithm is called Policy learning by Weighting Exploration with the Returns (PoWER) and can be derived from the same higher principle as previous policy gradient approaches, see [8] for details.

3.2.2 Policy learning by Weighting Exploration with the Returns (PoWER)

When learning motor primitives, we intend to learn a deterministic mean policy $\bar{\mathrm{a}} = \theta^{\mathrm{T}}\mu(\mathrm{s}) = \mathrm{f}(\mathrm{z})$ which is linear in parameters θ and augmented by additive exploration $\varepsilon(\mathrm{s},t)$ in order to make model-free reinforcement learning possible. As a result, the explorative policy can be given in the form $\mathrm{a} = \theta^{\mathrm{T}}\mu(\mathrm{s},t) + \varepsilon(\mu(\mathrm{s},t))$. Previous work in [14, 15], with the notable exception of [19], has focused on state-independent, white Gaussian exploration, i.e., $\varepsilon(\mu(\mathrm{s},t)) \sim \mathscr{N}(0,\Sigma)$, and has resulted into applications such as T-Ball batting [14] and constrained movement [3]. However, from our experience, such unstructured exploration at every step has several disadvantages, i.e., (i) it causes a large variance in parameter updates which grows with the number of time-steps, (ii) it perturbs actions too frequently, as the system acts as a low pass filter the perturbations average out and thus, their effects are 'washed' out and (iii) can damage the system executing the trajectory.

Alternatively, as introduced by [19], one could generate a form of structured, state-dependent exploration $\varepsilon(\mu(\mathrm{s},t)) = \varepsilon_t^{\mathrm{T}}\mu(\mathrm{s},t)$ with $[\varepsilon_t]_{ij} \sim \mathscr{N}(0,\sigma_{ij}^2)$, where σ_{ij}^2 are meta-parameters of the exploration that can be optimized in a similar manner. Each σ_{ij}^2 corresponds to one θ_{ij}. This argument results into the policy $\mathrm{a} \sim \pi(\mathrm{a}_t|\mathrm{s}_t,t) = \mathscr{N}(\mathrm{a}|\mu(\mathrm{s},\mathrm{t}),\hat{\Sigma}(\mathrm{s},t))$. This form of policies improves upon the shortcomings of directly perturbed policies mentioned above. Based on the EM updates for Reinforcement Learning as suggested in [8, 15], we can derive the update rule

$$\theta' = \theta + \frac{E_\tau\left\{\sum_{t=1}^{T}\varepsilon_t Q^\pi(\mathrm{s}_t,\mathrm{a}_t,t)\right\}}{E_\tau\left\{\sum_{t=1}^{T}Q^\pi(\mathrm{s}_t,\mathrm{a}_t,t)\right\}}, \tag{15}$$

where

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

is the state-action value function. Note that this algorithm does not need the learning rate as a meta-parameter.

In order to reduce the number of trials in this on-policy scenario, we reuse the trials through importance sampling [1, 24]. To avoid the fragility sometimes resulting from importance sampling in reinforcement learning, samples with very small importance weights are discarded.

The more shape parameters w are used the more details can be captured in a motor primitive and it can ease the imitation learning process. However, if the motor primitives need to be refined by RL, each additional parameter slows down the learning process The parameters σ_{ij}^2 determine the exploration behavior where larger values lead to greater changes in the mean policy and, thus, may lead to faster convergence but can also drive the robot in unsafe regimes. The optimization of the parameters decreases the exploration during convergence.

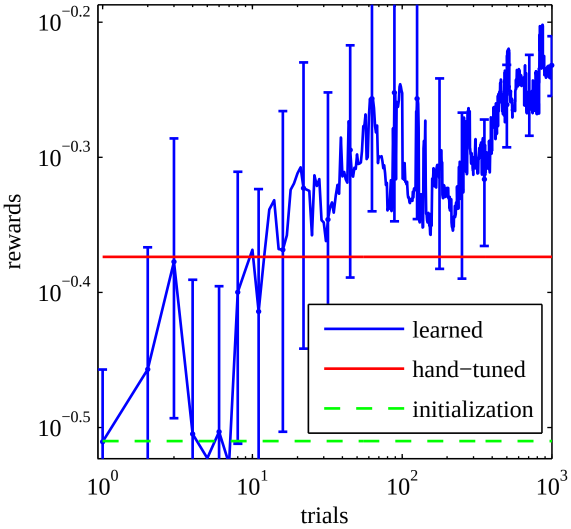


Fig. 4 This figure shows the expected return for one specific perturbation of the learned policy in the Ball-in-a-Cup scenario (averaged over 3 runs with different random seeds and the standard deviation indicated by the error bars). Convergence is not uniform as the algorithm is optimizing the returns for a whole range of perturbations and not for this 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 trials and converging after approximately 600 trials. Note that this plot is a double logarithmic plot and, thus, single unit changes are significant as they correspond to orders of magnitude.

4 Evaluation & Application

In this section, we demonstrate the effectiveness of the augmented framework for perceptually coupled motor primitives as presented in Section 2 and show that our concerted approach of using imitation for initialization and reinforcement learning for improvement works well in practice, particularly with our novel PoWER algorithm from Section 3. We show that this method can be used in learning a complex, real-life motor learning problem Ball-in-a-Cup in a physically realistic simulation of an anthropomorphic robot arm. This problem is a good benchmark for testing the motor learning performance and we show that we can learn the problem roughly at the efficiency of a young child. This algorithm successfully creates a perceptual coupling even to perturbations that are very challenging for a skilled adult player.

4.1 Robot Application: Ball-in-a-Cup

We have applied the presented algorithm in order to teach a physically-realistic simulation of an anthropomorphic SARCOS robot arm how to perform the traditional American children's game Ball-in-a-Cup, also known as Balero, Bilboquet or Kendama [27]. The toy has a small cup which is held in one hand (or, in our case, is attached to the end-effector of the robot) and the cup has a small ball hanging down on a string (the string has a length of 40cm for our toy). Initially, the ball is hanging down vertically in a rest position. The player needs to move fast in order to induce a motion in the ball through the string, toss it up and catch it with the cup, a possible movement is illustrated in Figure 3 in the top row.

Note that learning Ball-in-a-Cup and Kendama have previously been studied in robotics and we are going to contrast a few of the approaches here. While we learn directly in the joint space of the robot, Takenaka et al. [25] recorded planar human cup movements and determined the required joint movements for a planar, three degree of freedom (DoF) robot so that it could follow the trajectories while visual feedback was used for error compensation. Both Sato et al. [20] and Shone [23] used motion planning approaches which relied on very accurate models of the ball while employing only one DoF in [23] or two DoF in [20] so that the complete state-space could be searched exhaustively. Interestingly, exploratory robot moves were used in [20] to estimate the parameters of the employed model. The probably most advanced preceding work on learning Kendama was done by Miyamoto [10] who used a seven DoF anthropomorphic arm and recorded human motions to train a neural network to reconstruct via-points. Employing full kinematic knowledge, the authors optimize a desired trajectory. We previously learned a policy without perceptual coupling on a real seven DoF anthropomorphic Barrett WAM$^{\text{TM}}$ [8] developing the method used below to get the initial success.

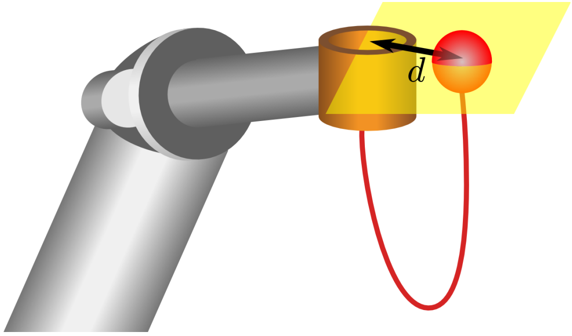


Fig. 5 This figure illustrates how the reward is calculated. The plane represents the level of the upper rim of the cup. For a successful rollout the ball has to be moved above the cup first. The reward is then calculated as the distance of the center of the cup and the center of the ball on the plane at the moment the ball is passing the plane in a downward direction.

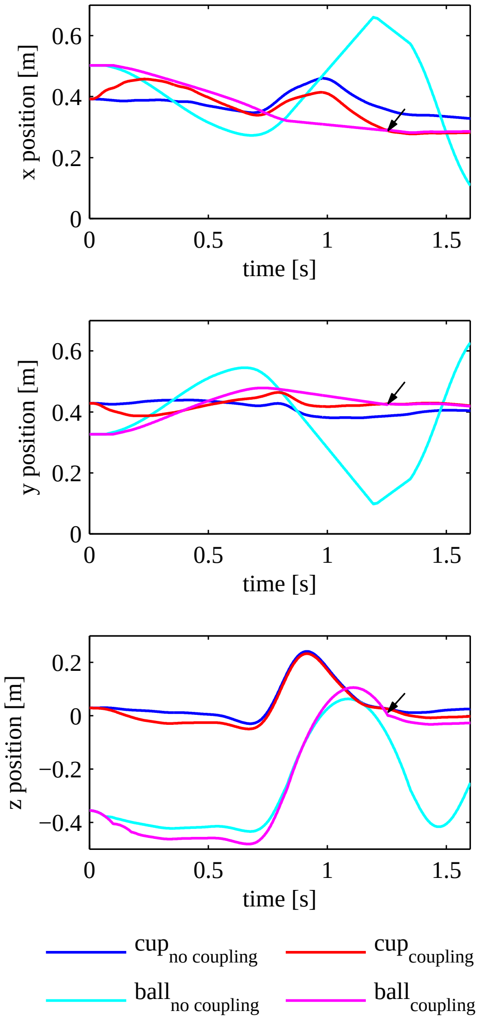
The state of the system is described in Cartesian coordinates of the cup (i.e., the operational space) and the Cartesian coordinates of the ball. The actions are the cup accelerations in Cartesian coordinates with each direction represented by a motor primitive. An operational space control law [11] 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 $r_t = \exp(-\alpha(x_c - x_b)^2 - \alpha(y_c - y_b)^2)$ the moment where the ball passes the rim of the cup with a downward direction and $r_t = 0$ all other times (see Figure 5). The cup position is 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 $\alpha = 10000$. 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 and the movements of the ball are very sensitive to perturbations. A small perturbation of the initial condition or the trajectory will drastically change the movement of the ball and hence the outcome of the trial if we do not use any form of perceptual coupling to the external variable "ball".

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 or deducing from similar previously learned tasks how to maneuver the ball above the cup in such a way that it can be caught. 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 on a consistent basis is even a hard task for adults, as our whole lab can testify. In contrast to a 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. Afterwards we also acquire the perceptual coupling by reinforcement learning.

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 3(c). The extracted cup-trajectories were used to initialize the motor primitives using locally-weighted regression for imitation learning. The simulation of the Ball-in-a-Cup behavior was verified using the tracked movements. 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 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 w for non perturbed initial conditions. The robot constantly succeeds at bringing the ball into the cup after approximately 60-80 iterations given no noise and perfect initial conditions.

One set of the found trajectories is then used to calculate the baseline $\bar{\mathrm{y}} = (\mathrm{h} - \mathrm{b})$ and $\dot{\bar{\mathrm{y}}} = (\dot{\mathrm{h}} - \dot{\mathrm{b}})$, where h and b are the hand and ball trajectories. This set is also used to set the standard cup trajectories.

Fig. 6 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 trial is marked by black arrows at the point where the ball enters the cup.



Without perceptual coupling the robot misses for even tiny perturbations of the initial conditions. Hand tuned coupling factors work quite well for small perturbations. In order to make them more robust we use reinforcement learning using the same joint reward as before. The initial conditions (positions and velocities) of the ball are perturbed completely randomly (no PEGASUS Trick) using Gaussian random values with variances set according to the desired stability region. The PoWER algorithm converges after approximately 600-800 iterations. This is roughly comparable to the learning speed of a 10 year old child (Figure 4). For the training we used concurrently standard deviations of 0.01m for x and y and of 0.1 m/s for $\dot{x}$ and $\dot{y}$. The learned perceptual coupling gets the ball in the cup 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 6 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.

The coupling factors represent the actions to be taken in order to get back to the desired relative positions and velocities of the ball with respect to the hand. This corresponds to an implicit model of how cup movements affect the ball movements. The factors at the beginning of the motion are small as there is enough time to correct the errors later on. At the very end the hand is simply pulled directly under the ball so it can fall into the cup. The perceptual coupling is robust to small changes of the parameters of the toy (string length, ball weight). 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, $\bar{\mathrm{y}}$ and $\dot{\bar{\mathrm{y}}}$ 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.

5 Conclusion

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 dynamical systems motor primitive framework to many other motor control problems. As manual tuning can only work in limited setups, an automatic acquisition of this perceptual coupling is essential.

In this paper, we have contributed an augmented version of the motor primitive framework originally suggested by [6, 7, 21] such that it incorporates perceptual coupling while keeping a distinctively similar structure to the original approach and, thus, preserving most of the important properties. We present a concerted learning approach which relies on an initialization by imitation learning and, subsequent, self-improvement by reinforcement learning. We introduce a particularly well-suited algorithm for this reinforcement learning problem called PoWER. 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.

References

1. Andrieu, C., de Freitas, N., Doucet, A., Jordan, M.I.: An introduction to MCMC for machine learning. Machine Learning **50**(1), 5–43 (2003)
2. Atkeson, C.G.: Using local trajectory optimizers to speed up global optimization in dynamic programming. In: J.E. Hanson, S.J. Moody, R.P. Lippmann (eds.) Advances in Neural Information Processing Systems 6 (NIPS), pp. 503–521. Morgan Kaufmann, Denver, CO, USA (1994)
3. Guenter, F., Hersch, M., Calinon, S., Billard, A.: Reinforcement learning for imitating constrained reaching movements. Advanced Robotics, Special Issue on Imitative Robots **21**(13), 1521–1544 (2007)
4. Howard, M., Klanke, S., Gienger, M., Goerick, C., Vijayakumar, S.: Methods for learning control policies from variable-constraint demonstrations. In: From Motor to Interaction Learning in Robots. Springer (2009)
5. Howard, M., Klanke, S., Gienger, M., Goerick, C., Vijayakumar, S.: A novel method for learning policies from variable constraint data. Autonomous Robots (2009)
6. Ijspeert, A.J., Nakanishi, J., Schaal, S.: Movement imitation with nonlinear dynamical systems in humanoid robots. In: Proc. IEEE Int. Conf. on Robotics and Automation (ICRA), pp. 1398–1403. Washington, DC (2002)
7. Ijspeert, A.J., Nakanishi, J., Schaal, S.: Learning attractor landscapes for learning motor primitives. In: S. Becker, S. Thrun, K. Obermayer (eds.) Advances in Neural Information Processing Systems 16 (NIPS), vol. 15, pp. 1547–1554. MIT Press, Cambridge, MA (2003)
8. Kober, J., Peters, J.: Policy search for motor primitives in robotics. In: Advances in Neural Information Processing Systems (NIPS) (2008)
9. Kulic, D., Nakamura, Y.: Incremental learning of full body motion primitives. In: From Motor to Interaction Learning in Robots. Springer (2009)
10. Miyamoto, H., Schaal, S., Gandolfo, F., Gomi, H., Koike, Y., Osu, R., Nakano, E., Wada, Y., Kawato, M.: A kendama learning robot based on bi-directional theory. Neural Networks **9**(8), 1281–1302 (1996)
11. Nakanishi, J., Mistry, M., Peters, J., Schaal, S.: Experimental evaluation of task space position/orientation control towards compliant control for humanoid robots. In: Proc. IEEE/RSJ 2007 Int. Conf. on Intell. Robotics Systems (IROS) (2007)
12. Nakanishi, J., Morimoto, J., Endo, G., Cheng, G., Schaal, S., Kawato, M.: A framework for learning biped locomotion with dynamic movement primitives. In: Proc. IEEE-RAS Int. Conf. on Humanoid Robots (HUMANOIDS). IEEE, Los Angeles, CA: Nov.10-12, Santa Monica, CA (2004)
13. Nakanishi, J., Morimoto, J., Endo, G., Cheng, G., Schaal, S., Kawato, M.: Learning from demonstration and adaptation of biped locomotion. Robotics and Autonomous Systems (RAS) **47**(2-3), 79–91 (2004)
14. Peters, J., Schaal, S.: Policy gradient methods for robotics. In: Proc. IEEE/RSJ 2006 Int. Conf. on Intell. Robots and Systems (IROS), pp. 2219 – 2225. Beijing, China (2006)
15. Peters, J., Schaal, S.: Reinforcement learning for operational space. In: Proc. Int. Conference on Robotics and Automation (ICRA). Rome, Italy (2007)
16. Pongas, D., Billard, A., Schaal, S.: Rapid synchronization and accurate phase-locking of rhythmic motor primitives. In: Proc. IEEE 2005 Int. Conf. on Intell. Robots and Systems (IROS), vol. 2005, pp. 2911–2916 (2005)
17. Ratliff, N., Silver, D., Bagnell, J.: Learning to search: Functional gradient techniques for imitation learning. Autonomous Robots **27**(1), 25–53 (2009)
18. Riedmiller, M., Gabel, T., Hafner, R., Lange, S.: Reinforcement learning for robot soccer. Autonomous Robots **27**(1), 55–73 (2009)
19. Rückstieß, T., Felder, M., Schmidhuber, J.: State-dependent exploration for policy gradient methods. In: Proceedings of the European Conference on Machine Learning (ECML), pp. 234–249 (2008)

20. Sato, S., Sakaguchi, T., Masutani, Y., Miyazaki, F.: 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)
21. Schaal, S., Mohajerian, P., Ijspeert, A.J.: Dynamics systems vs. optimal control — a unifying view. Progress in Brain Research **165**(1), 425–445 (2007)
22. Schaal, S., Peters, J., Nakanishi, J., Ijspeert, A.J.: Control, planning, learning, and imitation with dynamic movement primitives. In: Proc. Workshop on Bilateral Paradigms on Humans and Humanoids, IEEE 2003 Int. Conf. on Intell. Robots and Systems (IROS). Las Vegas, NV, Oct. 27-31 (2003)
23. Shone, T., Krudysz, G., Brown, K.: Dynamic manipulation of kendama. Tech. rep., Rensselaer Polytechnic Institute (2000)
24. Sutton, R., Barto, A.: Reinforcement Learning. MIT PRESS (1998)
25. Takenaka, K.: 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)
26. Urbanek, H., Albu-Schäffer, A., van der Smagt, P.: Learning from demonstration repetitive movements for autonomous service robotics. In: Proc. IEEE/RSL 2004 Int. Conf. on Intell. Robots and Systems (IROS), pp. 3495–3500. Sendai, Japan (2004)
27. Wikipedia: (2008). URL http://en.wikipedia.org/wiki/Ball_in_a_cup
28. Williams, R.J.: Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine Learning **8**, 229–256 (1992)
29. Wulf, G.: Attention and motor skill learning. Human Kinetics, Urbana Champaign, IL (2007)