A Probabilistic Approach to Robot Trajectory Generation

Alexandros Paraschos[1] and Gerhard Neumann[1] and Jan Peters[1,2]

***Abstract*— Motor Primitives (MPs) are a promising approach for the data-driven acquisition as well as for the modular and re-usable generation of movements. However, a modular control architecture with MPs is only effective if the MPs support co-activation as well as continuously blending the activation from one MP to the next. In addition, we need efficient mechanisms to adapt a MP to the current situation. Common approaches to movement primitives lack such capabilities or their implementation is based on heuristics. We present a probabilistic movement primitive approach that overcomes the limitations of existing approaches. We encode a primitive as a probability distribution over trajectories. The representation as distribution has several beneficial properties. It allows encoding a time-varying variance profile. Most importantly, it allows performing new operations — a product of distributions for the co-activation of MPs conditioning for generalizing the MP to different desired targets. We derive a feedback controller that reproduces a given trajectory distribution in closed form. We compare our approach to the existing state-of-the art and present real robot results for learning from demonstration.**

I. Introduction

Movement primitives (MP) [1], [2], [3] are considered a state-of-the-art-approach for learning robot movement generation. They have been used successfully to solve many complex tasks, including the 'Ball-in-the-Cup' game [4], Ball-Throwing [5], Pancake-Flipping [6] and bipedal gait generation [7]. The original motivation for using MPs is to compose complex behavior out of simpler building blocks of movement, or movement primitives. To this end, parallel activation of MPs in a modular control architecture is highly desirable. Parallel activation can drastically increase the expressibility of the modular architecture, and it allows for continuously blending the activation from one MP to the next. Moreover, to get reusable building blocks, we need to be able to modulate a MP, i.e., a MP needs to be able to generalize to different desired target states, via-points or execution speeds. Additionally, a MP representation should allow for learning from demonstrations and trial and error, be applicable for stroke-based as well as periodic movements and the policy of the MP should be able to represent optimal behavior in the given environment. While most of these properties are implemented by one or more MP representations [1], [2], [8], a unified framework is missing that combines all these properties in a principled way without relying on heuristics.

In this paper, we concentrate on trajectory-based movement representations. Such representations use time- or phase-dependent policies [1], [6], [9]. For example, the widely used dynamic movement primitive (DMP) approach [1] uses a phase signal to implement execution speed adaptation for the movement. DMPs are based on second-order dynamical systems, which are composed of a linear spring-damper system and a learnable non-linear forcing function. Integrating the dynamic system results in the desired trajectory, which is subsequently followed by feedback control laws. The DMP framework is well-established in robotics as they are straightforward to obtain from imitation [1] and reinforcement learning [10], can be used for both rhythmic and stroke-based movements, the movement can be generalized to different final-positions, and the movement speed can be adjusted by a temporal scaling parameter. The original DMP framework has been extended in [3] to generalize also to different final velocities and in [6] to use a time-varying spring-damper system which effectively modulates the stiffness of the executed movement.

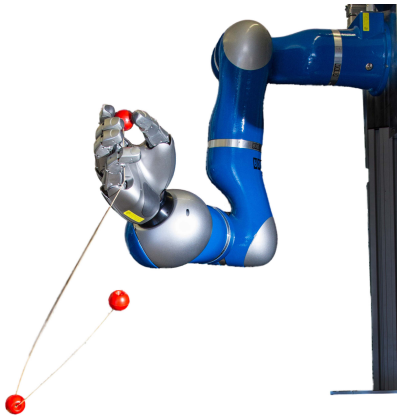


Fig. 1. The KUKA light-weight arm playing Astrojax using ProMPs.

However, there are a few desirable properties for a MP representation that are not fulfilled by current trajectory-based MP representations. Most importantly, multiple MPs for the same degrees of freedom (DoF) cannot be activated simultaneously without further considerations on prioritized control and partial cancellation of the movement. For example, a DMP generates the acceleration of the desired trajectory and it is unclear how to combine two such desired accelerations into a meaningful movement. Furthermore, a DMP cannot encode optimal behaviour in the presence of stochasticity, as following a single trajectory is always sub-optimal for stochastic systems [11]. Stochastic systems require to encode the variance of the resulting trajectories as we need to control the accuracy of the movement at relevant time points. DMPs can be efficiently used to imitate a single

[1] Intelligent Autonomous Systems lab, Technische Universitat Darmstadt, Germany {paraschos,neumann}@ias.tu-darmstadt.de
[2] Robot Learning Group, Max-Planck Institute for Intelligent Systems, Tuebingen, Germany mail@jan-peters.net

trajectory, however, how to generalize the shape of a DMP from multiple demonstrations is an open problem. Typically, the generalization to new final positions or velocities is based on heuristics and cannot be directly obtained by imitation learning. While some of the limitations of DMPs have been fixed by some extensions [8], [3], [5], such extensions are typically based on heuristics or only fix single limitations of the DMPs. For example, multiple MPs can be combined [8], but it is unclear how this combination affects the resulting motion.

In this paper, we introduce a probabilistic approach to robot trajectory generation that we call Probabilistic Movement Primitives (ProMP). ProMPs represent a movement primitive as a distribution over trajectories. Trajectory distributions can be easily multiplied which yields the 'intersection' of two distribution. This operation allows for a simultaneous activation of several primitives or a smooth switching from one activated primitive to the next. We can also condition our trajectory distribution to reach a desired position or velocity at any point in time as long as the position lies within the distribution. The trajectory distribution adapts to the new desired position while simultaneously trying to stay close to the demonstrations, i.e., the generalization to new desired targets is also learned from demonstration. Furthermore, the use of trajectory distributions allows us to represent a time-varying variance profile of the trajectories, and, hence, to encode optimal behaviors for stochastic systems [11]. Similarly to the DMP approach, our MPs can be easily obtained from imitation, can be used for both point-to-point and rhythmic movements, and the execution speed can be adapted by replacing time by a phase variable. While trajectory distributions support all of our desired properties for a MP representation, they would be of little use if we cannot use them to control a robot. We will first discuss how trajectory distributions can be learned and manipulated and subsequently show how to obtain a feedback control policy that follows the given trajectory distribution. Finally, we present several illustrative comparisons to the predominantly used DMP approach and we evaluate our approach on simulated and real robot scenarios. We investigate the use of ProMPs in a simulated table tennis application and we demonstrate how an anthropomorphic robot can play the game 'Astrojax'.

II. Probabilistic Trajectory Representation

In this section, we present our probabilistic representation of MPs. After explaining the benefits of trajectory distributions and how they can be modelled, we explain how we can perform new probabilistic operations on our MP representation, i.e., conditioning, combination, and blending of MPs.

A. Representation of a Single Trajectory

Our movement representation is based on a distribution $p(\boldsymbol{\tau}|\boldsymbol{\theta})$ over trajectories, where $\boldsymbol{\theta}$ represents the parameter vector of this distribution. Similar to the DMPs approach, we will model the MPs for each joint independently. A single trajectory $\boldsymbol{\tau} = \{q_t\}_{t=1...T}$ is represented by a sequence of time points, where $\boldsymbol{q}_t$ represents the joint angle at time point t. As we are seeking a compact trajectory representation, we use a linear basis-function model

$$\boldsymbol{y}_t = \begin{bmatrix} q_t \\ \dot{q}_t \end{bmatrix} = \begin{bmatrix} \boldsymbol{\psi}_t^T \\ \dot{\boldsymbol{\psi}}_t^T \end{bmatrix} \boldsymbol{w} + \begin{bmatrix} \epsilon_q \\ \epsilon_{\dot{q}} \end{bmatrix}, \tag{1}$$

for representing a single trajectory. The weight vector $\boldsymbol{w}$ compactly represents the trajectory $\boldsymbol{\tau}$. The vector $\boldsymbol{\psi}_t$ represents the basis functions at time point t and $\epsilon_q, \epsilon_{\dot{q}}$ zero-mean Gaussian i.i.d observation noise. The probability of observing a trajectory given the weight vector $\boldsymbol{w}$ is given by $p(\boldsymbol{\tau}|\boldsymbol{w}) = \prod_{t=1}^{T} p(\boldsymbol{y}_t|\boldsymbol{w})$, where

$$p(\boldsymbol{y}_t|\boldsymbol{w}) = \mathcal{N}\left(\boldsymbol{y} \middle| \boldsymbol{\Psi}_t^T \boldsymbol{w}, \boldsymbol{\Sigma}_y\right), \tag{2}$$

with $\Psi_t = [\boldsymbol{\psi}_t, \dot{\boldsymbol{\psi}}_t]$ and $\boldsymbol{\Sigma}_y$ contains the variances σ_q^2 for the position and velocities $\sigma_{\dot{q}}^2$ on its diagonal.

B. Representation of the Trajectory Distribution

As the weight vector $\boldsymbol{w}$ represents a single trajectory, we can abstract a distribution over trajectories as a distribution $p(\boldsymbol{w}|\boldsymbol{\theta})$, over the weight vector $\boldsymbol{w}$, which is parametrized by $\boldsymbol{\theta}$. The probability of observing a trajectory becomes

$$p(\boldsymbol{\tau}|\boldsymbol{\theta}) = \int p(\boldsymbol{\tau}|\boldsymbol{w}) p(\boldsymbol{w}|\boldsymbol{\theta}) d\boldsymbol{w}, \tag{3}$$

where we integrate out $\boldsymbol{w}$. The parameters $\boldsymbol{\theta}$ can be obtained by maximum likelihood estimates, e.g., using the expectation maximization (EM) algorithm [12]. We model the distribution over the weights as a Gaussian $p(\boldsymbol{w};\boldsymbol{\theta}) \sim \mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}_{\boldsymbol{w}}, \boldsymbol{\Sigma}_{\boldsymbol{w}})$, where $\boldsymbol{\theta}$ is given by the mean $\boldsymbol{\mu}_w$ and covariance matrix $\boldsymbol{\Sigma}_w$. Hence, the trajectory distribution $p(\boldsymbol{\tau}|\boldsymbol{w})$ is also given by a Gaussian. The mean $\boldsymbol{\mu}_t$ and covariance $\boldsymbol{\Sigma}_t$ at time point t are given by

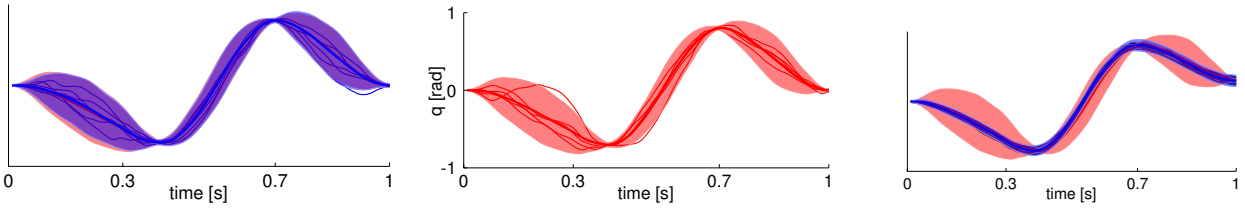
$$\boldsymbol{\mu}_t = \begin{bmatrix} \mu_{t,q} \\ \mu_{t,\dot{q}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\psi}_t^T \\ \dot{\boldsymbol{\psi}}_t^T \end{bmatrix} \boldsymbol{\mu}_{\boldsymbol{w}}, \tag{4}$$

$$\boldsymbol{\Sigma}_t = \begin{bmatrix} \sigma_{t,qq}^2 & \sigma_{t,q\dot{q}}^2 \\ \sigma_{t,q\dot{q}}^2 & \sigma_{t,\dot{q}\dot{q}}^2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{\psi}_t^T \boldsymbol{\Sigma}_{\boldsymbol{w}} \boldsymbol{\psi}_t & \boldsymbol{\psi}_t^T \boldsymbol{\Sigma}_{\boldsymbol{w}} \dot{\boldsymbol{\psi}}_t \\ \dot{\boldsymbol{\psi}}_t^T \boldsymbol{\Sigma}_{\boldsymbol{w}} \boldsymbol{\psi}_t & \dot{\boldsymbol{\psi}}_t^T \boldsymbol{\Sigma}_{\boldsymbol{w}} \dot{\boldsymbol{\psi}}_t \end{bmatrix} \tag{5}$$

and are obtained by integrating out the weights $\boldsymbol{w}$. The concept of trajectory distributions is also illustrated in Figure 2.

C. Phase Signal

A movement primitive typically needs to be executed at different speeds, e.g., when playing table tennis, we need to execute the strokes at different speeds to modulate the strength of the shot. Therefore, we need to temporally scale the movement primitive when playing table tennis, as we need to execute the strokes at different speeds in order to modulate the strength of the hit. As demonstrations from human tutors do not have generally the same durations, temporal scaling also allows learning from multiple demonstrations. We decouple Equations (4) and (5) from time t by introducing a phase variable z_t which is a monotonic



(a) Trajectory Distribution (b) Demonstrations (c) Single Trajectory

Fig. 2. (a) Trajectory distribution learned by the ProMP approach. Trajectories created by the stochastic feedback controller of the ProMP are plotted in blue. The ProMP can exactly reproduce the demonstrated trajectory distribution (shown in red below the blue shaded area). (b) Five out of 20 demonstrations (red lines) created by an stochastic optimal control algorithm for a via-point task. The resulting trajectories show a lot of variability due to the noise in the system. The red shaded area denotes two times the standard deviation of the demonstration. (c) Trajectory distribution created by following a DMP twenty times. The DMP is trained on the mean trajectory of the same demonstrations. While the DMP can follow the mean of the demonstrations, it can not adapt its variance. As a consequence, the accuracy at the via-points is worse as for the ProMPs, while the control actions are higher in non-relevant areas of the trajectory.

function of time. The speed profile $\dot{z}_t$of the phase determines the execution speed of the of the movement. We arbitrarily define the phase to be $z_0 = 0$ at the beginning and $z_T = 1$ at the end of the movement. The basis functions now depend on the phase instead of the time, such that $\boldsymbol{\psi}_t = \boldsymbol{\psi}(z_t)$ and the corresponding derivative of the basis becomes $\dot{\boldsymbol{\psi}}_t = \boldsymbol{\psi}'(z_t)\dot{z}_t$.

D. Basis Functions

We want to use our representation for stroke based and periodic movements. As for the DMPs we use normalized Gaussian basis functions ψ_i for stroke based movements, where

$$\boldsymbol{\psi}_i(z_t) = \frac{\phi_i(z)}{\sum_i \phi_i(z)}, \;\; \phi_i(z) = \exp\left(-\frac{(z_t - c_i)^2}{2h}\right). \quad (6)$$

The center of the i^{th} basis is denoted by c_i, and the basis width by h. We distribute the centers uniformly in the phase space, i.e., $c_i \in [0, 1]$. For encoding periodic movements, we use normalized Von-Mises basis functions,

$$\boldsymbol{\psi}_i(z_t) = \frac{\phi_i(z)}{\sum_i \phi_i(z)}, \phi_i(z) = \exp\left(h\cos(2\pi(z_t - c_i))\right), \quad (7)$$

which are periodic in the phase.

E. Generalizing by Conditioning

Generalization of MPs is a necessary requirement for a variety of applications. For example, in table tennis the visual information of the ball position is used to modify the position of the hitting point. We support generalization in the ProMPs framework by conditioning the learned distribution $p(\boldsymbol{w})$ over the weights w. Let us denote q_t^* and $\dot{q}_t^*$ as the desired angle and angular velocity respectively at time point t. We perform conditioning by

$$p\left(\boldsymbol{w}|[q_t^*, \dot{q}_t^*]^T\right) \propto p\left([q_t^*, \dot{q}_t^*]^T|\boldsymbol{w}\right) p(\boldsymbol{w}|\boldsymbol{\theta}) = \mathcal{N}\left(\begin{bmatrix} q_t^* \\ \dot{q}_t^* \end{bmatrix} \middle| \begin{bmatrix} \boldsymbol{\psi}_t^T \boldsymbol{w} \\ \dot{\boldsymbol{\psi}}_t^T \boldsymbol{w} \end{bmatrix}, \begin{bmatrix} \sigma_q^* & 0 \\ 0 & \sigma_{\dot{q}}^* \end{bmatrix}\right) \mathcal{N}\left(\boldsymbol{w}|\boldsymbol{\mu}_{\boldsymbol{w}}, \boldsymbol{\Sigma}_{\boldsymbol{w}}\right), \quad (8)$$

where σ_q^* denotes the desired accuracy on the angle q_t^* and $\sigma_{\dot{q}}^*$ on the velocity. The new probability distribution over the weights $p(\boldsymbol{w}|q_t^*, \dot{q}_t^*) = \mathcal{N}(\boldsymbol{w}|\mu_w^{\text{new}}, \Sigma_w^{\text{new}})$ can be computed analytically,

$$\boldsymbol{\mu}_{\boldsymbol{w}}^{[\text{new}]} = \boldsymbol{\mu}_{\boldsymbol{w}} + \boldsymbol{\Sigma}_{\boldsymbol{w}} \begin{bmatrix} \boldsymbol{\psi}_t^T \\ \dot{\boldsymbol{\psi}}_t^T \end{bmatrix} \boldsymbol{\Sigma}' \left(\begin{bmatrix} q_t^* \\ \dot{q}_t^* \end{bmatrix} - \begin{bmatrix} \boldsymbol{\psi}_t^T \\ \dot{\boldsymbol{\psi}}_t^T \end{bmatrix}^T \boldsymbol{\mu}_{\boldsymbol{w}} \right), \quad (9)$$

$$\boldsymbol{\Sigma}_{\boldsymbol{w}}^{[\text{new}]} = \boldsymbol{\Sigma}_{\boldsymbol{w}} - \boldsymbol{\Sigma}_{\boldsymbol{w}} \begin{bmatrix} \boldsymbol{\psi}_t^T \\ \dot{\boldsymbol{\psi}}_t^T \end{bmatrix} \boldsymbol{\Sigma}' \begin{bmatrix} \boldsymbol{\psi}_t^T \\ \dot{\boldsymbol{\psi}}_t^T \end{bmatrix}^T \boldsymbol{\Sigma}_{\boldsymbol{w}}, \quad (10)$$

where $\boldsymbol{\Sigma}'$ is given by

$$\boldsymbol{\Sigma}' = \left(\begin{bmatrix} \sigma_q^* & 0 \\ 0 & \sigma_{\dot{q}}^* \end{bmatrix} + \begin{bmatrix} \boldsymbol{\psi}_t^T \\ \dot{\boldsymbol{\psi}}_t^T \end{bmatrix}^T \boldsymbol{\Sigma}_{\boldsymbol{w}} \begin{bmatrix} \boldsymbol{\psi}_t^T \\ \dot{\boldsymbol{\psi}}_t^T \end{bmatrix} \right)^{-1}, \quad (11)$$

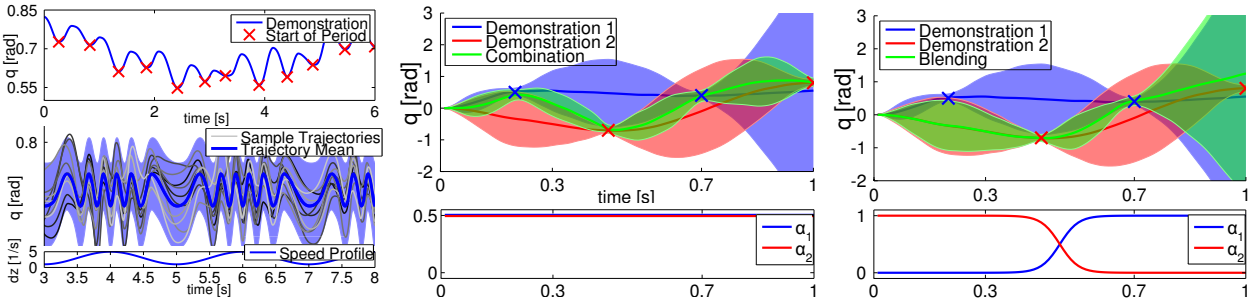
and $\boldsymbol{\mu}_{\boldsymbol{w}}^{[\text{new}]}$ denote the new mean, $\boldsymbol{\Sigma}_{\boldsymbol{w}}^{[\text{new}]}$ the new covariance of the weight vector $\boldsymbol{w}$. In Figure 4 we compare our approach to DMPs for different goal positions. We observe that the DMPs approach is able to reach the target positions and velocities, but it fails to preserve the shape of the trajectory obtained by the demonstrations. As a consequence, in tasks in which the timing of the primitive is important, the use of DMPs leads to inferior performance.

F. Combination and Blending

While humans are able to seemingly blend and combine movements, MPs approaches rarely have this capability. Our probabilistic representations allows us to combine and blend a set of different primitives into a single one. We co-activate multiple primitives simultaneously by taking the product of distributions,

$$p_{new}(\boldsymbol{\tau}) \propto \prod_i p_i(\boldsymbol{\tau})^{\alpha^{[i]}}, \quad (12)$$

where primitive i is activated with the factor $\alpha^{[i]}$. The concurrent co-activation results in the 'intersection' of all distributions, which could be seen as the overlapping region of all distributions. For example, if two trajectory distributions are used to solve two different tasks but have an overlapping solution in the joint space, the product of both distributions will result in a primitive that solves a combination of the two tasks. Additionally, we may not only want to combine primitives, but rather transition smoothly



(a) Temporal Modulation (b) Combination (c) Blending

Fig. 3. (a) (top) Demonstration of a rhythmic movement obtained via kinesthetic teach-in. The beginning of each period is indicated by a red cross. (middle) Learned movements of the ProMP approach, where we already adapted the speed profile of the phase variable z. The blue line and blue shaded area shows the mean and standard deviation of the reproduction. The lines in grey show the generated trajectories from the ProMPs. While the trajectory distribution is periodic, the single trajectories show the demonstrated variability. (bottom) The speed profile of the phase variable. (b) Combination (green) of two movement primitives (red and blue). Each of the original primitives had to reach two different via-points (indicated by 'x'). In the combination, both primitives are active, and hence, all four via-points are reached by the green primitive. (c) Blending (green) of movement primitives. In the beginning, the red movement primitive is active. At $t = 0.5s$, we smoothly switch the activation to the blue primitive and the red primitive is ignored.

form one primitive to another, also denoted as blending between two movements. Continuous blending is achieved by

$$p_{new}(\boldsymbol{\tau}) \propto \prod_t \prod_i p_i(\boldsymbol{y}_t)^{\alpha_t^{[i]}}, \tag{13}$$

where the activations $\alpha_t^{[i]}$ are time-varying. The new distribution is a product of Gaussian distributions and can be computed analytically. The updated mean and covariance are given as

$$\boldsymbol{\Sigma}_t^{[\text{new}]} = \left(\sum_i \left(\boldsymbol{\Sigma}_t^{[i]} / \alpha_t^{[i]} \right)^{-1} \right)^{-1}, \tag{14}$$

$$\boldsymbol{\mu}_t^{[\text{new}]} = \left(\boldsymbol{\Sigma}_t^{[\text{new}]} \right)^{-1} \left(\sum_i \left(\boldsymbol{\Sigma}_t^{[i]} / \alpha_t^{[i]} \right)^{-1} \boldsymbol{\mu}_t^{[i]} \right). \tag{15}$$

The mean $\boldsymbol{\mu}_t$ and covariance matrix $\boldsymbol{\Sigma}_t$ are used to determine the gains of the stochastic feedback controller that we will derive in Section III.

III. Derivation of the Corresponding Controller

To use the resulting trajectory distributions, we need a control law. We derive a stochastic feedback controller that exactly reproduces the desired trajectory distribution $p(\boldsymbol{\tau}|\boldsymbol{\theta})$. To do so, we use a model-based approach in which an approximate forward model is used for predicting the distribution at the next time step. Given the desired distribution and the model, the control signals at each time step are obtained in closed form.

A. Controller Architecture

We model the underlying system by a second order linear dynamical system as it enables us to derive a closed form solution of the feedback controller. Such a linear model is also used to represent the robot's dynamics in DMPs approach. It is a valid model given the assumption that the inverse dynamics model is sufficiently precise for controlling the robot [9], [13]. The state of the linear system is given by the joint angle q_t and velocity $\dot{q}_t$ and the control output $u_t = \ddot{q}_t$ by the desired acceleration. For a given duration δt between two subsequent time-steps, the next state of the dynamical system is given by

$$q_{t+\delta\text{t}} = q_t + \dot{q}_t \, \delta\text{t}, \tag{16}$$

$$\dot{q}_{t+\delta\text{t}} = \dot{q}_t + u_t \, \delta\text{t}. \tag{17}$$

We generate the controller output by a feedback control law,

$$u_t = P_t q_t + D_t \dot{q}_t + k_t + \epsilon_u, \tag{18}$$

where P_t and D_t denote the time-varying position and velocity gains, k_t denotes the feed-forward component and the controller noise $\epsilon_u \sim \mathcal{N}(0, \sigma_u/\delta\text{t})$ is assumed to be zero mean i.i.d. Gaussian. To allow a subsequent transition to continuous time systems, we use a control noise that behaves like a Wiener process [14]. Its variance grows linearly with the time step δt. We substitute Equation (18) in (17) to obtain a new transition probability for the velocity,

$$\dot{q}_{t+\delta\text{t}} = P_t \, \delta\text{t} \, q_t + (1 + D_t \, \delta\text{t}) \, \dot{q}_t + (k_t + \epsilon_t) \, \delta\text{t}. \tag{19}$$

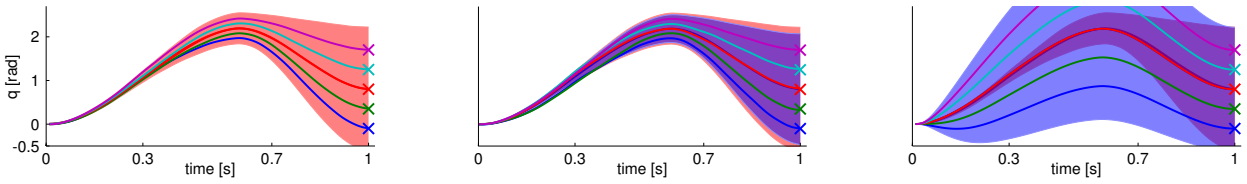
We use our model to compute the distribution of the state of the system at the next time step $t + \delta$t, analytically. From Equations (16) and (19), given the distribution $p([q_t, \dot{q}_t]^T)$ of the current time step, we obtain

$$\begin{aligned} p\left(\begin{bmatrix} q_{t+\delta\text{t}} \\ \dot{q}_{t+\delta\text{t}} \end{bmatrix} \right) &= \int p\left(\begin{bmatrix} q_{t+\delta\text{t}} \\ \dot{q}_{t+\delta\text{t}} \end{bmatrix} \middle| \begin{bmatrix} q_t \\ \dot{q}_t \end{bmatrix} \right) p\left(\begin{bmatrix} q_{t+\delta\text{t}} \\ \dot{q}_{t+\delta\text{t}} \end{bmatrix} \right) \mathrm{d}q_t \, \mathrm{d}\dot{q}_t \\ &= \int \mathcal{N}\left(\begin{bmatrix} q_{t+\delta\text{t}} \\ \dot{q}_{t+\delta\text{t}} \end{bmatrix} \middle| \begin{bmatrix} m_1\,(q_t, \dot{q}_t) \\ m_2\,(q_t, \dot{q}_t) \end{bmatrix}, \begin{bmatrix} 0 & 0 \\ 0 & \sigma_u \end{bmatrix} \right) \\ &\quad \mathcal{N}\left(\begin{bmatrix} q_t \\ \dot{q}_t \end{bmatrix} \middle| \begin{bmatrix} \mu_{t,q} \\ \mu_{t,\dot{q}} \end{bmatrix}, \begin{bmatrix} \Sigma_{t,qq} & \Sigma_{t,q\dot{q}} \\ \Sigma_{t,q\dot{q}}^T & \Sigma_{t,\dot{q}\dot{q}} \end{bmatrix} \right) \mathrm{d}q_t \, \mathrm{d}\dot{q}_t, \end{aligned} \tag{20}$$

where

$$m_1\,(q_t, \dot{q}_t) = q_t + \dot{q}_t \, \delta\text{t}, \tag{21}$$

$$m_2\,(q_t, \dot{q}_t) = P_t \, \delta\text{t} \, q_t + (1 + D_t \, \delta\text{t}\,) \, \dot{q}_t + k_t \, \delta\text{t}. \tag{22}$$



(a) Optimal Control (b) ProMPs (c) DMPs

Fig. 4. (a) Demonstrations created by an optimal control algorithm for different target *positions*. (b) Reproduced generalization to the target states by ProMPs. The ProMPs can reached the target states while keeping the demonstrated shape of the trajectory. (c) Generalization with the DMPs by adapting the goal attractor. While the DMPs can reach the target states, the similarity with the demonstrations is lost.

The integral from Equation (20) can be computed analytically and yields a Gaussian distribution with mean

$$\begin{bmatrix}\mu_{t+\delta\mathrm{t},q}\\ \mu_{t+\delta\mathrm{t},\dot{q}}\end{bmatrix} = \begin{bmatrix}\mu_{t,q} + \mu_{t,\dot{q}}\,\delta\mathrm{t}\\ P_t\,\delta\mathrm{t}\,\mu_{t,q} + (1 + D_t\,\delta\mathrm{t})\,\mu_{t,\dot{q}} + k_t\,\delta\mathrm{t}\end{bmatrix}, \quad (23)$$

and the elements of the covariance matrix

$$\sigma^2_{t+\delta\mathrm{t},q} = \sigma_{t,q} + 2\sigma_{t,q\dot{q}}\,\delta\mathrm{t} + O\left(\delta\mathrm{t}^2\right), \quad (24)$$

$$\begin{aligned}\sigma^2_{t+\delta\mathrm{t},q\dot{q}} &= P_t\,\delta\mathrm{t}\,\sigma_{t,q} + (1 + D_t\,\delta\mathrm{t})\,\sigma_{t,q\dot{q}}\\ &\quad + \sigma_{t,\dot{q}}\,\delta\mathrm{t} + O\left(\delta\mathrm{t}^2\right)\\ &= \sigma_{t+\delta\mathrm{t},\dot{q}q}, \end{aligned} \quad (25)$$

$$\begin{aligned}\sigma^2_{t+\delta\mathrm{t},\dot{q}} &= P_t\,\delta\mathrm{t}\,\left(\sigma_{t,\dot{q}} + \sigma_{t,q\dot{q}}\right) + \sigma_{t,\dot{q}} +\\ &\quad D_t\,\delta\mathrm{t}\,\left(\sigma_{t,\dot{q}} + \sigma_{t,q\dot{q}}\right) + \sigma_u^2\,\delta\mathrm{t} + O\left(\delta\mathrm{t}^2\right), \end{aligned} \quad (26)$$

where $O(\delta\mathrm{t}^2)$ contains the second order terms in $\delta\mathrm{t}$. The stochastic control law that we specified in Equation (18) is fully determined at each control step by the feed-forward component k_t, the feedback gains P_t and D_t, and the noise variance σ_u. In this paper, we assume that the noise variance σ_u^2 of the controller is given. In the following sections, we show the analytical computations for all of these terms.

B. Derivation of the Feedback Gains

We derive our controller parameters by matching the predicted distribution from our model, see Equation (20), with the desired distribution at $t + \delta\mathrm{t}$. We observe that both distributions are Gaussian, thus it is sufficient to match the first two moments of both distributions. We begin our derivations by computing the feedback gains. We equalize Equations (25) and (26) and we rearrange the terms to formulate difference of $\sigma^2_{t+\delta\mathrm{t},i} - \sigma^2_{t,i}$ for $i \in \{q, q\dot{q}\}$. We transit to a continuous time formulation by dividing by $\delta\mathrm{t}$ and taking the limit $\delta\mathrm{t} \to 0$, which results in

$$\dot{\sigma}^2_{t,q\dot{q}} = P_t\,\sigma^2_{t,q} + D_t\,\sigma^2_{t,q\dot{q}} + \sigma^2_{t,\dot{q}}, \quad (27)$$

$$\dot{\sigma}^2_{t,\dot{q}} = P_t\left(\sigma^2_{t,\dot{q}} + \sigma^2_{t,q\dot{q}}\right) + D_t\left(\sigma^2_{t,\dot{q}} + \sigma^2_{t,q\dot{q}}\right) + \sigma_u^2 \quad (28)$$

the derivatives of $\sigma^2_{t,q\dot{q}}$ and $\sigma^2_{t,\dot{q}}$, where all the second order terms $O(\delta\mathrm{t}^2)$ disappear. All the right-hand side terms of Equation (28), except from the gains P_t and D_t, can be computed analytically from Equation (5). Additionally, the left-hand side terms can also be computed analytically

$$\dot{\sigma}_{t,q\dot{q}} = \left(\boldsymbol{\psi}_t^T\boldsymbol{\sigma_w}\dot{\boldsymbol{\psi}}_t\right)' = \dot{\boldsymbol{\psi}}_t^T\boldsymbol{\sigma_w}\dot{\boldsymbol{\psi}}_t + \boldsymbol{\psi}_t^T\boldsymbol{\sigma_w}\ddot{\boldsymbol{\psi}}_t, \quad (29)$$

$$\dot{\sigma}_{t,\dot{q}} = \left(\dot{\boldsymbol{\psi}}_t^T\boldsymbol{\sigma_w}\dot{\boldsymbol{\psi}}_t\right)' = 2\dot{\boldsymbol{\psi}}_t^T\boldsymbol{\sigma_w}\ddot{\boldsymbol{\psi}}_t, \quad (30)$$

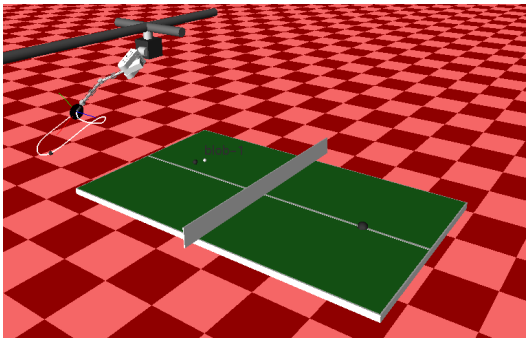


Fig. 5. The table tennis setup. On the left, the BioRob arm is mounted on linear axis, the ball position denoted by 'blob-1' and the ball prediction. On the opponent's side is the robot's target for this simulation. In our experiments, we use 15 different combinations of ball initial positions and robot's targets covering most of the table.

in a similar fashion. Therefore, we solve the equation system defined by Equations (25) and (26) on P_t, D_t by substitution, and we obtain

$$P_t = \frac{\dot{\sigma}^2_{t,q\dot{q}} - \sigma^2_{t,\dot{q}} - D_t\sigma^2_{t,q\dot{q}}}{\sigma^2_{t,q}} \quad (31)$$

$$D_t = \frac{\dot{\sigma}^2_{t,\dot{q}} - \sigma_u^2 - \left(\dot{\sigma}^2_{t,q\dot{q}} - \sigma^2_{t,\dot{q}}\right)\left(\sigma^2_{t,\dot{q}} + \sigma^2_{t,q\dot{q}}\right)}{\sigma^2_{t,q}\left(1 - \sigma^2_{t,q\dot{q}}\sigma^2_{t,q}\right)\left(\sigma^2_{t,\dot{q}} + \sigma^2_{t,q\dot{q}}\right)} \quad (32)$$

where all of the terms can be computed analytically.

C. Derivation of the Feed-Forward Controls

Similarly, we obtain the closed-from solution for the feed-forward control signal k_t. We equalize the mean of the velocity of the predictive distribution from Equation (23) and the desired distribution, from Equation (4) at the next time step. Rearranging, dividing by $\delta\mathrm{t}$, and taking the limit of $\delta\mathrm{t} \to 0$ yields

$$\dot{\mu}_{t,\dot{q}} = P_t\mu_{t,q} + D_t\mu_{t,\dot{q}} + k_t, \quad (33)$$

which we solve for the feed-forward control signal

$$k_t = \dot{\mu}_{t,\dot{q}} - P_t\mu_{t,q} - D_t\mu_{t,\dot{q}} \quad (34)$$

$$= \ddot{\boldsymbol{\psi}}_t^T\boldsymbol{\mu}_w - P_t\boldsymbol{\psi}_t^T\boldsymbol{\mu}_w - D_t\dot{\boldsymbol{\psi}}_t^T\boldsymbol{\mu}_w, \quad (35)$$

where all terms can be computed analytically.

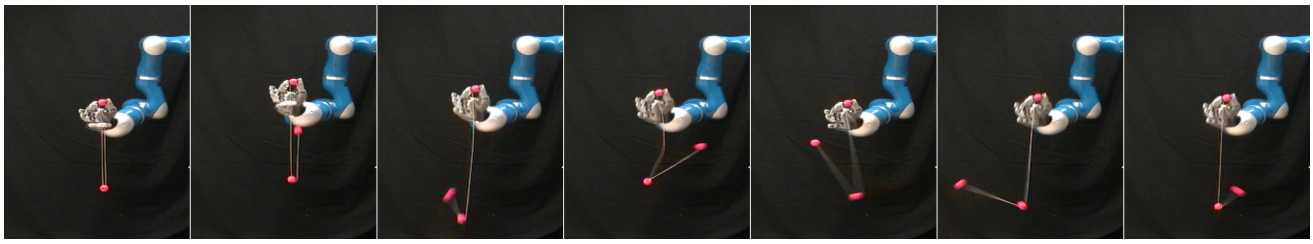


Fig. 7. The KUKA light-weight arm using the toy 'Astrojax'. The robot holds on of the balls in his fingers and starts with releasing the ball connected to the other end of the string. He subsequently reproduces the demonstrated rhythmic movement showing the same human-like variability in its movement pattern.

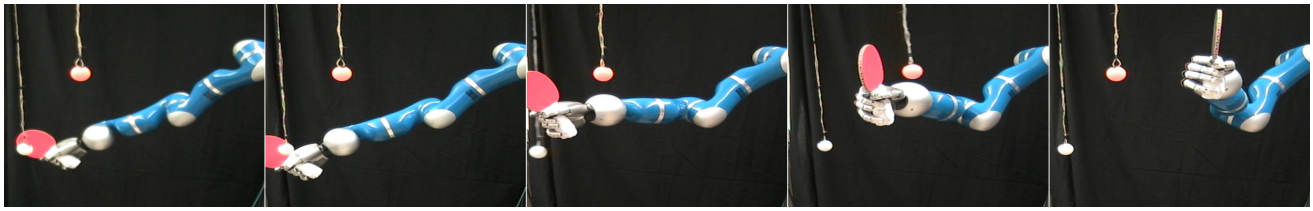


Fig. 8. The robot learned from demonstration how to hit one of two targets (indicated by the white and orange table tennis balls) while avoiding the other. The figure shows the combination of both movement primitives. By activating both primitives, the robot hit the white ball and subsequently the orange ball.

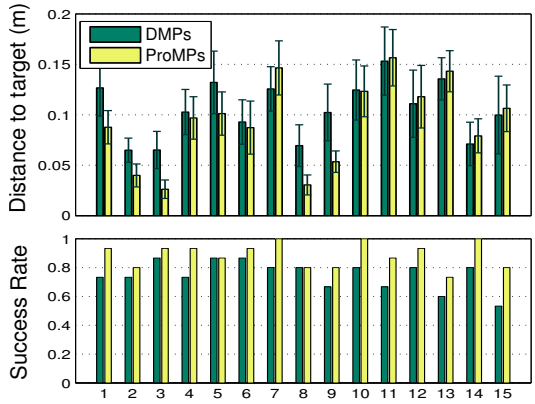


Fig. 6. (top) The distance between the ball position when landed on the opponents field and the actual targeted point in meters, for the DMP and the ProMP approaches. We tested 15 different combinations of ball initial states and robot's targets, and we average the data over 20 samples. The bars denote the mean error and the error-bars one standard deviation. (bottom) The success rate for each combination. If the distance between the landed position and the target position is less than 0.4 meters is counted as a success. The performance of ProMPs is superior in all the experiments leading generally to smaller errors with an increased success rate.

IV. Experimental Evaluation

We evaluated our approach on one realistically simulated task and on two real-robot tasks. For the simulated task, we use a 6-DoF BioRob arm, mounted on 2-DoF linear axis, while for the real-robot tasks we use a 7-DoF KUKA anthropomorphic, light-weight arm, which is equipped with a 5-finger, 15 DoF hand made by DLR. In both cases, we use an inverse dynamics model to control the robot, where the ProMPs directly controlled the desired acceleration of the joints. The robot is illustrated in Figure 1. For the simulated experiment, we used the ProMPs for playing table tennis. The trajectories used for training were from an analytical player [15]. In the first real-robot experiment, we demonstrated rhythmic trajectories for the toy 'Astrojax' [16] by kinesthetic teach-in. In the second experiment, we generated demonstrations for hitting a ball with a table tennis racket. We generated two different types of movements for hitting the ball at different positions. We were able to combine both movements such that the robot could hit the two balls in sequence with a single primitive.

A. Table Tennis

In this experiment, we use a physically realistic model of a table tennis setup. We use a BioRob 5-DoF arm [17] mounted on two linear axis, with an additional shoulder joint, depicted at Figure 5. We control the robot with inverse dynamics control, however, the readings of the current positions have a lag of one time step to simulate the system realistically. As a result, the desired and actual trajectories do not match exactly, thus, making the robot more sensitive to jerky movements.

At the start of each episode, the ball is set to different pre-specified positions and velocities, targeting the robot. Then, the ball is simulated with second-order Euler integration. The robot has to return the ball to a specific target area at the opponents field. For our experiment, we gathered trajectories for 15 different combinations of initial ball states and robot targets, generated from an analytical player. We trained the ProMPs approach with the whole data-set and created one primitive. In our experiment, the ball state is set at the beginning and the ProMP is conditioned to the hitting position of the analytical player. A delay before the start of the execution of the primitive is provided by the simulation.

In order to make the task more realistic, we assume that the ball state is estimated, instead of being directly observed, with zero-mean i.i.d. Gaussian noise. The estimation of the

ball increases the task difficulty significantly as it also affects the delay before the execution of the primitive. We display our results on Figure 6, and we compare our approach to DMPs. We train the DMP with the one demonstration, and we modify the goal position and velocity according to the built-in generalization approach. The DMP approach has inferior performance when executed in a noisy environment, as the generalization capabilities of the DMP destroy the shape of the demonstrated trajectory, as shown in Figures 4.

B. Playing Astrojax

'Astrojax' is a toy consisting of three balls on a string. Two balls are fixed at either end of the string and one ball is free to slide along the string. Roughly, 'Astrojax' is a game between 'YoYo' and juggling. A wide variety of tricks can be performed on it. We demonstrated a rhythmic movement to the robot which created a 'basic orbit' pattern. We subsequently used the ProMPs to learn the movement where we used 30 Von-Mises basis functions for each joint. The robot could reproduce the behavior and recreated the same pattern, which is illustrated in Figure 7. As the demonstrations exhibited a lot of variability, the robot produced periodic movements which showed the same type of variability. The demonstrated and the learned movements for one DoF is shown in Figure 3(a). In contrast, the DMP approach would repeat always exactly the same movement, rendering the behavior of the robot less human-like.

C. Ball Hitting Combination Task

In this task, the robot had to hit different targets with a table tennis racket. We initially demonstrated two distinct movements for hitting the ball at two different locations. The distance of the target locations to the robot was chosen differently, such that, the time of contact with the first target was before the time of contact with the second target. For each target location, we demonstrated five trajectories. After learning the primitives, the robot could hit each of the target locations independently, with the corresponding primitive. We subsequently activated both primitives simultaneously to get the combination of both primitives. The robot hit both targets with one stroke. The resulting movement of the robot is illustrated in Figure 8.

V. Conclusion

Movement primitives are a promising approach for learning, modulating, and re-using movements in a modular control architecture. To effectively take advantage of such a control architecture, the MPs need to support simultaneous activation, match the quality of the encoded behavior from the demonstrations, be able to adapt to different desired target positions, and efficiently learn by imitation. In this paper, we introduced a probabilistic approach to movement primitives, which meets all these requirements. We parametrize the desired trajectory distribution of the primitive. The trajectory distribution can be obtained from demonstrations and simultaneously defines a feedback controller, which is used for movement execution. Our probabilistic formulation allows for new operations on movement primitives, including conditioning and combination of primitives. The resulting method works well on three benchmark tasks with simulated and real robots. Future work will focus on using the ProMPs in a modular control architecture and improving upon imitation learning by reinforcement learning.

Acknowledgements

The authors want to thank for the support of the European Union projects # FP7-ICT-270327 (Complacs) and # FP7-ICT-2011-9 (CoDyCo).

References

[1] A. J. Ijspeert and S. Schaal, "Learning Attractor Landscapes for Learning Motor Primitives," in *Advances in Neural Information Processing Systems 15*, ser. (NIPS). Cambridge, MA: MIT Press, 2003.

[2] S. M. Khansari-Zadeh and A. Billard, "Learning Stable Non-Linear Dynamical Systems with Gaussian Mixture Models," *IEEE Transaction on Robotics*, 2011.

[3] J. Kober, K. Mülling, O. Kroemer, C. Lampert, B. Schölkopf, and J. Peters, "Movement Templates for Learning of Hitting and Batting," in *International Conference on Robotics and Automation (ICRA)*, 2010.

[4] J. Kober and J. Peters, "Policy Search for Motor Primitives in Robotics," *Machine Learning*, pp. 1–33, 2010.

[5] A. Ude, A. Gams, T. Asfour, and J. Morimoto, "Task-Specific Generalization of Discrete and Periodic Dynamic Movement Primitives," *Trans. Rob.*, no. 5, Oct. 2010.

[6] P. Kormushev, S. Calinon, and D. Caldwell, "Robot Motor Skill Coordination with EM-based Reinforcement Learning," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010.

[7] J. Nakanishi, J. Morimoto, G. Endo, G. Cheng, S. Schaal, and M. Kawato, "Learning from demonstration and adaptation of biped locomotion," *Robotics and Autonomous Systems*, 2004.

[8] L. Rozo, S. Calinon, D. G. Caldwell, P. Jimenez, and C. Torras, "Learning Collaborative Impedance-Based Robot Behaviors," in *AAAI Conference on Artificial Intelligence*, 2013.

[9] L. Sciavicco and B. Siciliano, *Modelling and Control of Robot Manipulators*, 2nd ed. Springer, 2005.

[10] J. Peters and S. Schaal, "Reinforcement Learning of Motor Skills with Policy Gradients," *Neural Networks*, no. 4, pp. 682–97, 2008.

[11] E. Todorov and M. Jordan, "Optimal Feedback Control as a Theory of Motor Coordination," *Nature Neuroscience*, vol. 5, pp. 1226–1235, 2002.

[12] A. Lazaric and M. Ghavamzadeh, "Bayesian Multi-Task Reinforcement Learning," in *Proceedings of the 27th International Conference on Machine Learning (ICML)*, 2010.

[13] J. Peters, M. Mistry, F. E. Udwadia, J. Nakanishi, and S. Schaal, "A Unifying Methodology for Robot Control with Redundant DOFs," *Autonomous Robots*, no. 1, pp. 1–12, 2008.

[14] H. Stark and J. W. Woods, *Probability and Random Processes with Applications to Signal Processing (3rd Edition)*, 3rd ed., Aug. 2001.

[15] K. Mülling, J. Kober, and J. Peters, "A Biomimetic Approach to Robot Table Tennis," Max-Planck-Gesellschaft. Piscataway, NJ, USA: IEEE, 10 2010, pp. 1921–1926.

[16] Wikipedia, "Astrojax," 2013, [Online; accessed 1-Feb-2013]. [Online]. Available: http://en.wikipedia.org/wiki/Astrojax

[17] S. Klug, T. Lens, O. von Stryk, B. Möhl, and A. Karguth, "Biologically inspired robot manipulator for new applications in automation engineering," in *Proceedings of Robotik 2008*, ser. VDI-Berichte, no. 2012. Munich, Germany: VDI Wissensforum GmbH, June 11-12 2008.