# Extracting Low-Dimensional Control Variables for Movement Primitives

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

***Abstract*— Movement primitives (MPs) provide a powerful framework for data driven movement generation that has been successfully applied for learning from demonstrations and robot reinforcement learning. In robotics we often want to solve a multitude of different, but related tasks. As the parameters of the primitives are typically high dimensional, a common practice for the generalization of movement primitives to new tasks is to adapt only a small set of control variables, also called meta parameters, of the primitive. Yet, for most MP representations, the encoding of these control variables is pre-coded in the representation and can not be adapted to the considered tasks. In this paper, we want to learn the encoding of task-specific control variables also from data instead of relying on fixed meta-parameter representations. We use hierarchical Bayesian models (HBMs) to estimate a low dimensional latent variable model for probabilistic movement primitives (ProMPs), which is a recent movement primitive representation. We show on two real robot datasets that ProMPs based on HBMs outperform standard ProMPs in terms of generalization and learning from a small amount of data and also allows for an intuitive analysis of the movement. We also extend our HBM by a mixture model, such that we can model different movement types in the same dataset.**

## I. INTRODUCTION

Movement primitives (MPs) are a compact parametric description of a movement [17], [8], [9], [4]. They provide a powerful framework for data driven movement generation as they can be learned from demonstrations as well as by reinforcement learning. They can adapt to a new task by adapting a given set of meta-parameters [25], [11], [13]. For example, the final joint positions or the execution speed [8] of the movement can be adapted. Yet, for most movement primitive representations, the set of meta-parameters is pre-coded into the movement primitive representation and can not be adapted. However, for most tasks, a different encoding of the meta-parameters might be more appropriate than the pre-coded parameters of the primitive representation. We believe that this shortcoming has also hindered the application of movement primitives for more complex multi-task learning applications. In this paper we want to learn the encoding of the meta-parameters also from data. Our approach extracts a low-dimensional manifold in the MP parameter space. Each point on this manifold is described by a small set of control variables. Hence, our underlying assumption is that, while the parametrization of movements might be high-dimensional, useful parameter vectors for a given set of tasks typically share a lot of structure, i.e. they lie on a lower dimensional manifold. Each demonstration can now be characterized by the corresponding control variables that can be seen as a compact description of the task considered in this demonstration. For example, in a table tennis scenario, these control variables could specify the location of the hitting point or the desired return direction for the ball. Hence, our model can not only be applied for efficient generalization in multi-task learning with movement primitives but is also well suited for analyzing the movements of human demonstrators. We represent the latent manifold model by a hierarchical Bayesian model. The control variables for each demonstrations are treated as latent variables that are also inferred from the data. The model is extended by a mixture model such that we can learn the control variables of multiple types of movements. We will use Probabilistic Movement Primitives (ProMPs) as underlying movement primitive representation as they can be naturally integrated in the Hierarchical Bayesian Model (HBM) representation. When learning or analyzing movement data, we have to deal with several challenges, such as high-dimensionality, noise, missing data, partial observations, and the data can contain multiple modes that represent different types of movements. In order to deal with all these requirements, we apply a fully Bayesian approach where we integrate out all the estimated parameters of the model. In our experiments, we will illustrate the improved generalization properties of our approach compared to the standard ProMP approach in the case of a small amount of training data and show how demonstrations can be easily analyzed and characterized by the extracted latent control variables.

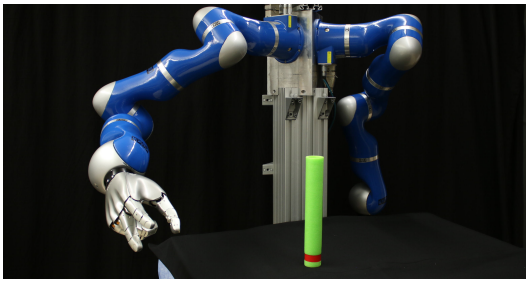

Fig. 1. The robot used in the experiments to learn trajectory distributions.

### A. Related Work

Movement primitives can be categorized into trajectory-based [8], [23], [17] and state-based representations [9]. In this paper we will focus on trajectory based approaches as they are more commonly used and easier to scale up to higher dimensions. A common trajectory-based approach are the dynamical movement primitives (DMPs). DMPs [8] are

[1]Intelligent Autonomous Systems Lab, Technische Universität Darmstadt, Hochschulstr. 10, 64289 Darmstadt, Germany {rueckert, mundo, paraschos, neumann}@ias.tu-darmstadt.de

[2]Robot Learning Group, Max-Planck Institute for Intelligent Systems, Tuebingen, Germany mail@jan-peters.net

represented by a parametrized dynamical system that is given by a linear point-attractor that is perturbed by a non-linear time dependent forcing function. The forcing function can be used to encode an arbitrary shape of the trajectory and the weights of the forcing function can be easily obtained from demonstrations by linear regression. One of the benefits of the DMP approach is that it specifies a small set of meta-parameters. These meta-parameters include the final position of the movement, which is given by the point attractor, the final velocities, the execution speed, or the amplitude of the movement [10], [20], [8]. In multi-task learning with DMPs [11], [7], [13], it is a common strategy to only adapt the meta-parameters due to the high dimensionality of the weights of the forcing function. While DMPs have several more benefits such as stability, and the ability to represent stroke based and rhythmic movements, DMPs also have several limitations, such as that they can not represent optimal behavior in stochastic systems and the adaptation of the trajectory due to the meta-parameters is based on heuristics. These issues have been addressed by the recently proposed Probabilistic Movement Primitives approach [17], [18]. ProMPs estimate a distribution of trajectories instead of encoding single trajectories. The main benefit of the probabilistic representation is that we can use probabilistic operators such as conditioning for adaptation and a product of distribution for co-activating primitives. A distribution over trajectories also contains information on which time points are relevant for the movement, e.g., time points with small variance in the Cartesian end-effector space could denote task relevant via-points or targets. However, in difference to DMPs, ProMPs are lacking meta-parameters that can be used to adapt the trajectories with a small amount of control variables. While it would be easy to pre-specify such control variables by conditioning the trajectory distribution for a fixed set of time points, such an approach would again require a lot of manual tuning and is lacking flexibility.

Our approach automatically extracts a small amount of control variables from a given set of demonstrations in the ProMP framework. We use a hierarchical Bayesian approach to model prior distributions, which is inspired by techniques from multi-task learning (MTL) [27], [15], [12], [24]. In MTL the underlying assumption is that multiple tasks (or trajectories) share a common structure, and, hence, with an increasing number of related tasks that have been already learned, the number of needed training samples for generalizing to a new task decreases [1]. This property is highly desired in robotics, where the data is often high dimensional and obtaining training samples is costly. Different approaches exist to model the shared information across tasks. They can be roughly separated into two different categories, i.e. methods where parameters of the model are close to each other in a geometric sense [6], [24] and approaches where the parameters of the model share a common structure [27], [26], [3], [15], [21], [19]. This structure can be a clustering assumption [26], a (Gaussian) prior for the parameters of all tasks [27], [15] or some advanced structure like the Kingman's coalescent [3], which is a continuous time, partitioned valued Markov process. Our approach is highly related to the Bayesian MTL approach presented in [19], where a prior distribution over parameters is learned. The prior distribution is assumed to have a low-dimensional, latent structure that is represented by a linear factor model. In order to represent several modes (or non-linearities) in the data, the model is extended to a mixture model of linear factor models. For both, the number of mixture components and the number of factors, a non-parametric Dirichlet prior has been used. All parameters of the model are integrated out by the use of a combination of sampling and variational inference. We will use a simplification of this model, assuming a fixed number of mixture components, without the Dirichlet priors, allowing a much more efficient algorithm without the need for expensive sampling methods. We extend the model of Passos et al. by an additional hyper-prior and show that this hyper-prior significantly increases the robustness of the Bayesian model.

## II. Probabilistic Movement Primitives

In this section we will give a brief overview on Probabilistic Movement Primitives (ProMPs) as they provide the foundation for our hierarchical Bayesian model. ProMPs represent a movement by a distribution $p(\boldsymbol{\tau})$ over trajectories $\boldsymbol{\tau} = \boldsymbol{y}_{1:T}$, where $\boldsymbol{y}_t$ specifies the joint positions (or any other quantities, such as a Cartesian coordinates of a ball) at time step $t$. ProMPs use a linear basis function model with $J$ basis functions to represent a single trajectory, i.e.

$$p(\boldsymbol{y}_t|\boldsymbol{w}) = \mathcal{N}\left(\boldsymbol{y}_t\middle|\boldsymbol{\Psi}_t\boldsymbol{w}, \beta^{-1}\boldsymbol{I}\right) \text{ and } p(\boldsymbol{\tau}) = \prod_{t=1}^{T} p\left(\boldsymbol{y}_t|\boldsymbol{w}\right),$$

where $\beta$ denotes the precision of the data. The weight vector $\boldsymbol{w}$ is a compact representation of the trajectory. The basis functions $\boldsymbol{\Psi}_t$ only depend on the time or, alternatively, on the phase of the movement. For a single Degree of Freedom (DoF), $\boldsymbol{\Psi}_t$ is just given by a vector of normalized Gaussian basis functions $\boldsymbol{\phi}_t$ with

$$\phi_{t,i} = \frac{\exp\left(-0.5(t-c_i)^2\right)}{\sum_{j=1}^{J}\exp\left(-0.5(t-c_j)^2\right)},$$

where $c_i$ denotes the center of the $i$th basis function (note that to enhance readability we skipped the bandwidth parameters in this notation).

For multi-dimensional systems with $D$ DoFs, the basis function matrix is represented by a block-diagonal matrix, i.e,

$$\boldsymbol{\Psi}_t = \begin{bmatrix} \boldsymbol{\phi}_t^T & \boldsymbol{0}^T & \dots & \boldsymbol{0}^T \\ \boldsymbol{0}^T & \boldsymbol{\phi}_t^T & \dots & \boldsymbol{0}^T \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0}^T & \boldsymbol{0}^T & \boldsymbol{0}^T & \boldsymbol{\phi}_t^T \end{bmatrix}.$$

Due to this encoding of the basis function matrix, the trajectories of all DoFs can still be represented as a single weight vector $\boldsymbol{w}^T = [\boldsymbol{w}_1^T, \boldsymbol{w}_2^T, \dots, \boldsymbol{w}_D^T]$ that is given by a concatenation of all weight vectors for each degree of freedom.

Still, a single weight vector $\boldsymbol{w}$ only represents a single trajectory $\boldsymbol{\tau}$. In order to represent a distribution over trajectories $p(\boldsymbol{\tau})$, we can estimate a distribution $p(\boldsymbol{w})$ over the weight vectors and, subsequently, integrate out the weight vectors. In the original ProMP approach, a multivariate Gaussian distribution is used to model the prior distribution

$$p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}_w, \boldsymbol{\Sigma}_w). \tag{1}$$

As such, the distribution over trajectories is also Gaussian and can be computed in closed form

$$\begin{aligned} p(\boldsymbol{\tau}) &= \int p(\boldsymbol{\tau}|\boldsymbol{w})p(\boldsymbol{w})d\boldsymbol{w}, \\ &= \int \mathcal{N}\left(\boldsymbol{y}_{1:T}\middle|\boldsymbol{\Psi}_{1:T}\,\boldsymbol{w}, \beta^{-1}\boldsymbol{I}\right)\mathcal{N}\left(\boldsymbol{w}|\boldsymbol{\mu}_w, \boldsymbol{\Sigma}_w\right)d\boldsymbol{w}, \\ &= \mathcal{N}\left(\boldsymbol{y}_{1:T}\middle|\boldsymbol{\Psi}_{1:T}\,\boldsymbol{w}, \boldsymbol{\Psi}_{1:T}\boldsymbol{\Sigma}_w\boldsymbol{\Psi}_{1:T}^T + \beta^{-1}\boldsymbol{I}\right), \end{aligned}$$

where $\boldsymbol{\Psi}_{1:T}$ is a $TD \times DJ$ matrix containing the basis function matrices for all time steps and $\boldsymbol{w}$ is a $DJ$ dimensional column vector.

### A. Learning from Demonstrations with ProMPs

A ProMP already defines a simple hierarchical Bayesian model in a similar fashion as a Bayesian linear regression model. The mean $\boldsymbol{\mu}_w$ and the covariance matrix $\boldsymbol{\Sigma}_w$ can be learned from data by maximum likelihood using the Expectation Maximization (EM) algorithm [5]. A simpler solution that works well in practice is to compute first the most likely estimate of $\boldsymbol{w}^{[i]}$ for each trajectory $\boldsymbol{\tau}^{[i]}$ independently, where the index $i$ denotes the $i$-th demonstration[1]. Subsequently, mean and covariance of $p(\boldsymbol{w})$ can be estimated by the sample mean and sample covariance of the $\boldsymbol{w}^{[i]}$'s. One advantage of the EM based approach in comparison to the more direct approach is that the EM algorithm can also be used for learning from incomplete data where, e.g., some segments of the trajectories might be missing due to occlusions in vision based recordings.

However, the training of ProMPs also suffers from a severe disadvantage. As the model has a lot of parameters due to the high-dimensional covariance matrix, ProMPs suffer from overfitting if we have little training data and noisy trajectories. The more sophisticated hierarchical Bayesian model for ProMPs introduced in this paper alleviates this problem.

### B. Predictions with ProMPs by Conditioning

ProMPs can also be used to predict the behavior of the demonstrator once we have seen an initial part of a new trajectory. Lets assume that we have observed a human demonstrator at $m = 1, 2, ..., M$ different time points[2] $t_1$ to $t_M$ at the positions $\boldsymbol{y}_{t_1}$ to $\boldsymbol{y}_{t_M}$. Let us further denote $\boldsymbol{\Psi_o}$ as the concatenation of the basis function matrices for these time points and $\boldsymbol{o}$ as concatenation of the $\boldsymbol{y}_{t_m}$ vectors. Given these observations, we can obtain a conditioned distribution $p(\boldsymbol{w}|\boldsymbol{o})$ over the weight vectors. This distribution is Gaussian with mean and variance

$$\begin{aligned} \boldsymbol{\mu}_{\boldsymbol{w}|\boldsymbol{o}} &= \boldsymbol{\mu}_{\boldsymbol{w}} + \\ &\quad \boldsymbol{\Sigma}_{\boldsymbol{w}}\boldsymbol{\Psi}_{\boldsymbol{o}}^T\left(\boldsymbol{\Sigma}_o + \boldsymbol{\Psi}_{\boldsymbol{o}}\boldsymbol{\Sigma}_{\boldsymbol{w}}\boldsymbol{\Psi}_{\boldsymbol{o}}^T\right)^{-1}\left(\boldsymbol{o} - \boldsymbol{\Psi}_o\boldsymbol{\mu}_{\boldsymbol{w}}\right), \end{aligned} \tag{2}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{w}|\boldsymbol{o}} = \boldsymbol{\Sigma}_{\boldsymbol{w}} - \boldsymbol{\Sigma}_{\boldsymbol{w}}\boldsymbol{\Psi}_{\boldsymbol{o}}^T\left(\boldsymbol{\Sigma}_o + \boldsymbol{\Psi}_{\boldsymbol{o}}\boldsymbol{\Sigma}_{\boldsymbol{w}}\boldsymbol{\Psi}_{\boldsymbol{o}}^T\right)^{-1}\boldsymbol{\Psi}_{\boldsymbol{o}}\boldsymbol{\Sigma}_{\boldsymbol{w}}. \tag{3}$$

The conditional distribution $p(\boldsymbol{w}|\boldsymbol{o})$ can be used to predict the behavior of the demonstrator for future time points $t > t_M$, i.e. we can determine the mean and covariance of $\boldsymbol{y}$ for future time points. Note that the same procedure can be applied for partial observations, where only a subset of the quantities in $\boldsymbol{y}_t$ is observed. The covariance matrix $\boldsymbol{\Sigma}_o$ can be used to control the importance of different dimensions.

## III. Extracting Control Variables with Hierarchical Priors

Our goal is to model non-linear prior distributions that can be modulated by low-dimensional latent control variables. We define a hierarchical prior on the weight vector $\boldsymbol{w}$ using mixture models

$$p(\boldsymbol{w}^{[i]}) = \sum_{k=1}^{K}\pi_k\mathcal{N}\left(\boldsymbol{w}^{[i]}\middle|\boldsymbol{b}_k + \boldsymbol{M}_k\boldsymbol{h}_k^{[i]}, \alpha^{-1}\boldsymbol{I}\right). \tag{4}$$

The vector $\boldsymbol{b}_k$ denotes an offset term and the projection matrix $\boldsymbol{M}_k$ defines the mapping from the low-dimensional control variables $\boldsymbol{h}_k^{[i]}$ to the weight vector $\boldsymbol{w}^{[i]}$ of trajectory $i$. The parameter $\alpha$ models the precision of the latent manifold priors and $\pi_k$ denotes the mixing coefficients. The different mixture components can model different movement types, e.g., forehand and backhand strokes in a table tennis game. Within a mixture component the latent control variable $\boldsymbol{h}_k^{[i]}$ models the adaptation of the movement to the current task.

All parameters of this prior distribution are unknown a priori and are learned from demonstrations. We follow a fully Bayesian approach, where we treat all parameters as random variables and introduce conjugate priors for these random variables. We derive variational update equations for all relevant distributions. We also demonstrate how predictions can be computed by conditioning with the hierarchical priors.

We will start our discussion for the most simple case, using only a single mixture component.

### A. Control Variables for a Single Movement Type

For a single mixture component the prior in Eq. (4) simplifies to

$$p(\boldsymbol{w}^{[i]}) = \mathcal{N}\left(\boldsymbol{w}^{[i]}\middle|\boldsymbol{b} + \boldsymbol{M}\boldsymbol{h}^{[i]}, \alpha^{-1}\boldsymbol{I}\right).$$

We introduce conjugate priors for the random variables, i.e. we use $p(\boldsymbol{b}) = \mathcal{N}(\boldsymbol{b}|\boldsymbol{0}, \boldsymbol{I})$ for the offset vector, $p(\boldsymbol{h}^{[i]}) = \mathcal{N}(\boldsymbol{h}^{[i]}|\boldsymbol{0}, \boldsymbol{I})$ for the control variables $\boldsymbol{h}^{[i]}$, $\alpha = \Gamma(\alpha|a_0, b_0)$ for the precision[3] $\alpha$, and $p(\boldsymbol{M}) = \prod_v \mathcal{N}(\boldsymbol{m}^{[v]}|\boldsymbol{0}, {\lambda^{[v]}}^{-1}\boldsymbol{I})$ for the projection matrix $\boldsymbol{M}$. Here, $\boldsymbol{m}^{[v]}$ denotes the $v$-th

[1] Given a trajectory $\boldsymbol{\tau}_i$, the corresponding weight vectors $\boldsymbol{w}^{[i]}$ can be estimated by a straight forward least squares estimate.

[2] Note that these time points do not need to be sampled in uniform time intervals.

[3] To make this prior non-informative we use $a_0 = 1e{-}5$ and $b_0 = 1e{-}5$.

column of the matrix $\boldsymbol{M} = [\boldsymbol{m}^{[1]}, \boldsymbol{m}^{[2]}, \ldots, \boldsymbol{m}^{[V]}]$, with $V$ denoting the dimensionality of the latent variable $\boldsymbol{h}^{[i]}$. The symbol $\Gamma$ denotes the Gamma distribution.

To enhance the numerical stability of the variational updates, we also add a gamma prior on the precision parameters of the projection matrix, i.e. $p(\lambda^{[v]}) = \Gamma(\lambda^{[v]}|c_0, d_0)$. The influence of this additional prior is evaluated in the experimental section.

As we use a variational inference approach [2], we assume a complete factorization of the variational posterior given by

$$q(\boldsymbol{\xi}) = q(\boldsymbol{b})q(\boldsymbol{M})q(\lambda_{1:V})q(\alpha)\prod_{i=1}^{L} q(\boldsymbol{w}^{[i]})q(\boldsymbol{h}^{[i]}),$$

where $\boldsymbol{\xi} = \{\boldsymbol{w}^{[1:L]}, \boldsymbol{h}^{[1:L]}, \boldsymbol{b}, \boldsymbol{M}, \lambda_{1:V}, \alpha\}$ and $L$ denotes the total number of demonstrations. The variational distributions for the weight vector $\boldsymbol{w}^{[i]}$, the latent variable $\boldsymbol{h}^{[i]}$, the offset vector $\boldsymbol{b}$, the $v$-th column of the projection matrix $\boldsymbol{M}$, are specified as $q(\boldsymbol{w}^{[i]}) := \mathcal{N}(\boldsymbol{w}^{[i]}|\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}}, \boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}})$, $q(\boldsymbol{h}^{[i]}) := \mathcal{N}(\boldsymbol{h}^{[i]}|\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}, \boldsymbol{\Sigma}_{\boldsymbol{h}^{[i]}})$, $q(\boldsymbol{b}) := \mathcal{N}(\boldsymbol{b}|\boldsymbol{\mu}_{\boldsymbol{b}}, \sigma_{\boldsymbol{b}}\boldsymbol{I})$, and $q(\boldsymbol{m}^{[v]}) := \mathcal{N}(\boldsymbol{m}^{[v]}|\boldsymbol{\mu}_{\boldsymbol{m}^{[v]}}, \sigma_{\boldsymbol{m}^{[v]}}\boldsymbol{I})$. The remaining definitions are listed in the appendix.

The most important variational update equations read

$$\begin{aligned}
\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} &= \boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}\left(\beta {\boldsymbol{\Psi}^{[i]}_{1:T}}^{T} \boldsymbol{y}^{[i]}_{1:T} + \bar{\alpha}\left(\boldsymbol{\mu}_{\boldsymbol{b}} + \bar{\boldsymbol{M}}\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}\right)\right),\\
\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}} &= \left(\beta {\boldsymbol{\Psi}^{[i]}_{1:T}}^{T} \boldsymbol{\Psi}^{[i]}_{1:T} + \bar{\alpha}\boldsymbol{I}\right)^{-1},\\
\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}} &= \bar{\alpha}\,\boldsymbol{\Sigma}_{\boldsymbol{h}^{[i]}}\,\bar{\boldsymbol{M}}^{T}\left(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}}\right),\\
\boldsymbol{\mu}_{\boldsymbol{b}} &= \sigma_{\boldsymbol{b}}\bar{\alpha}\boldsymbol{I}\left(\sum_{i=1}^{L}\left(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \bar{\boldsymbol{M}}\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}\right)\right),\\
\boldsymbol{\mu}_{\boldsymbol{m}^{[v]}} &= \sigma_{\boldsymbol{m}^{[v]}}\bar{\alpha}\boldsymbol{I}\left(\sum_{i=1}^{L}\mu_{h^{[v,i]}}\left(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}}\right)\right),
\end{aligned}$$

where $\bar{\boldsymbol{M}} = [\boldsymbol{\mu}_{\boldsymbol{m}^{[1]}}, \ldots, \boldsymbol{\mu}_{\boldsymbol{m}^{[V]}}]$. The inferred feature precision is denoted by $\bar{\alpha}$ and the scalar $\mu_{h^{[v,i]}}$ denotes the $v$-th element in the vector $\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}} = [\mu_{h^{[1,i]}}, \ldots, \mu_{h^{[V,i]}}]^T$.

Compared to the prior used in ProMPs in Eq. (1), the combination of the latent variable $\boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}$ and the projection matrix $\bar{\boldsymbol{M}}$ implements a more accurate model of the prior distribution. As we will demonstrate, this hierarchical prior model is less sensitive to overfitting in the case of noisy observations or incomplete data.

### *B. Predictions by Conditioning the Hierarchical Prior*

In the hierarchical prior model, predictions are performed by computing the conditioned distribution over the latent task variable $p(\boldsymbol{h}|\boldsymbol{o})$. This conditioned distribution can be simply determined by integrating out the weight vector $\boldsymbol{w}$

$$\begin{aligned}
p(\boldsymbol{h}|\boldsymbol{o}) &\propto p(\boldsymbol{o}|\boldsymbol{h})p(\boldsymbol{h}),\\
&= \int_{\boldsymbol{w}} p\left(\boldsymbol{o}|\boldsymbol{\Psi}_{\boldsymbol{o}}, \boldsymbol{w}\right) p\left(\boldsymbol{w}|\boldsymbol{h}\right) p(\boldsymbol{h}) d\boldsymbol{w},\\
&= \mathcal{N}\left(\boldsymbol{o}\middle|\boldsymbol{\Psi}_{\boldsymbol{o}}\left(\boldsymbol{\mu}_{\boldsymbol{b}} + \bar{\boldsymbol{M}}\boldsymbol{h}\right), \boldsymbol{\Sigma}_o + \bar{\alpha}^{-1}\boldsymbol{\Psi}_{\boldsymbol{o}}\boldsymbol{\Psi}_{\boldsymbol{o}}^T\right) p(\boldsymbol{h}),
\end{aligned}$$

where $p(\boldsymbol{h})$ is the Gaussian prior distribution for the latent variable. Now, we can condition on the control variable $\boldsymbol{h}$ on the demonstrations to obtain a Gaussian over $\boldsymbol{h}$ with mean and variance

$$\boldsymbol{\mu}_{\boldsymbol{h}|\boldsymbol{o}} = \bar{\boldsymbol{M}}^T \boldsymbol{\Psi}_{\boldsymbol{o}}^T \boldsymbol{A}^{-1}\left(\boldsymbol{o} - \boldsymbol{\Psi}_{\boldsymbol{o}}\boldsymbol{\mu}_{\boldsymbol{b}}\right), \tag{5}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{h}|\boldsymbol{o}} = \boldsymbol{I} - \bar{\boldsymbol{M}}^T \boldsymbol{\Psi}_{\boldsymbol{o}}^T \boldsymbol{A}^{-1} \boldsymbol{\Psi}_{\boldsymbol{o}} \bar{\boldsymbol{M}}, \tag{6}$$

where $\boldsymbol{A} = \boldsymbol{\Sigma}_o + \boldsymbol{\Psi}_{\boldsymbol{o}}\left(\bar{\alpha}^{-1}\boldsymbol{I} + \bar{\boldsymbol{M}}\bar{\boldsymbol{M}}^T\right)\boldsymbol{\Psi}_{\boldsymbol{o}}^T$.

Given the distribution over the inferred latent task variable the posterior over feature weights is given by

$$\boldsymbol{\mu}_{\boldsymbol{w}|\boldsymbol{o}} = \boldsymbol{\mu}_{\boldsymbol{b}} + \bar{\boldsymbol{M}}\boldsymbol{\mu}_{\boldsymbol{h}|\boldsymbol{o}}, \tag{7}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{w}|\boldsymbol{o}} = \bar{\alpha}^{-1}\boldsymbol{I} + \bar{\boldsymbol{M}}\boldsymbol{\Sigma}_{\boldsymbol{h}|\boldsymbol{o}}\bar{\boldsymbol{M}}^T. \tag{8}$$

It is illustrative to investigate the differences of the standard conditioning of the ProMPs in Eq. (2) and Eq. (3) to the conditioning with the hierarchical prior. The conditioning in the ProMP case requires a full-rank covariance matrix, which is hard to obtain given a small amount of training data. In contrast, the latent prior model only requires the projection matrix $\bar{\boldsymbol{M}}$ to perform the conditioning. Hence, the predictions of the latent prior model are less prone to overfitting and are, therefore, also applicable for a small amount of training data.

### *C. Extension to Multiple Movement Types ($K > 1$)*

The mixture distribution in Eq. (4) adds an additional multinomial variable per demonstration to our probabilistic model, i.e. $z_k^{[i]} \in \{0, 1\}$. We represent this multinomial variable as binary vector $\mathbf{z}^{[i]} = \{z_1^{[i]}, ..., z_K^{[i]}\}$.

To derive variational updates, we specify a multinomial hyper-prior for the mixing indices $p(\boldsymbol{Z}) = \prod_{i=1}^{L}\prod_{k=1}^{K}(\boldsymbol{\pi}_k)^{z_k^{[i]}}$.

The variational updates are the same as for the case with only a single component, with the difference that the trajectories are weighted by the responsibilities of the individual mixture components $\mu_{z_k^{[i]}}$, i.e.

$$\begin{aligned}
\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} =& \boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}\Bigg(\beta {\boldsymbol{\Psi}^{[i]}_{1:T}}^{T} \boldsymbol{y}^{[i]}_{1:T} +\\
&\sum_{k=1}^{K}\bar{\alpha}_k\mu_{z_k^{[i]}}\left(\boldsymbol{\mu}_{\boldsymbol{b}_k} + \bar{\boldsymbol{M}}_k\boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}}\right)\Bigg),\\
\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}} =& \left(\beta {\boldsymbol{\Psi}^{[i]}_{1:T}}^{T} \boldsymbol{\Psi}^{[i]}_{1:T} + \sum_{k=1}^{K}\bar{\alpha}_k\mu_{z_k^{[i]}}\boldsymbol{I}\right)^{-1}.
\end{aligned}$$

Computing predictions with the mixture model is also straight forward. For each component we compute the conditioned distribution on the latent control variables as in Eq. (5) and in Eq. (6) and the posterior over the feature weights using Eq. (7) and Eq. (8). Thereafter the posterior distributions are

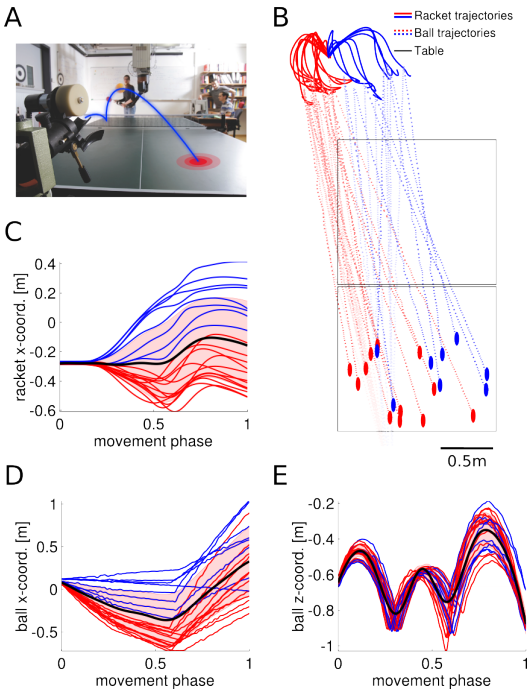


Fig. 2. (A-B) Trajectory prediction task in a table tennis setting using 20 end-effector and ball trajectories. (C-E) Learned distributions over trajectories for three dimensions (out of six) using ProMPs. The colors (red and blue) are only used to visualize differences in the movement directions.

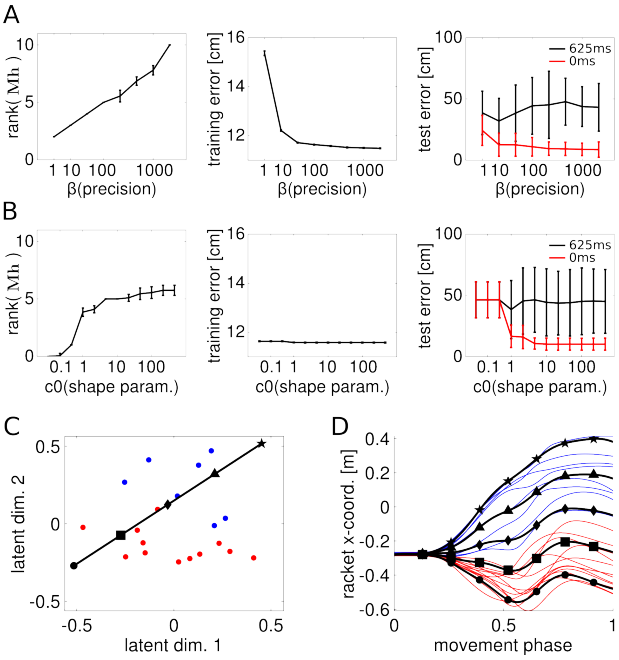


Fig. 3. (A) The data precision parameter $\beta$ can be used to adapt the model complexity while avoiding overfitting (shown in the 2nd and 3rd panel for two planning horizons until the ball impact). (B) The gamma prior on the precision parameters $\lambda$ to increase the numerical stability has little effect on the prediction performance (for $c_0 \geq 1$). (C) Investigation of the effect of the latent variables, where the first dimension of $\mathbf{h}$ describes the slope whereas the second dimension relates to the waviness (D).

weighted by the responsibilities of each mixture model

$$z^{[k]} = \frac{\pi_k \mathcal{N}\left(\boldsymbol{o}\middle|\boldsymbol{\mu}_{\boldsymbol{w}|\boldsymbol{o}}^{[k]}, \boldsymbol{\Sigma}_{\boldsymbol{w}|\boldsymbol{o}}^{[k]}\right)}{\sum_{j=1}^{K} \pi_j \mathcal{N}\left(\boldsymbol{o}\middle|\boldsymbol{\mu}_{\boldsymbol{w}|\boldsymbol{o}}^{[j]}, \boldsymbol{\Sigma}_{\boldsymbol{w}|\boldsymbol{o}}^{[j]}\right)},$$

$$\boldsymbol{\Sigma}_{\boldsymbol{w}|\boldsymbol{o}} = \sum_{k=1}^{K} z^{[k]} \boldsymbol{\Sigma}_{\boldsymbol{w}|\boldsymbol{o}}^{[k]},$$

$$\boldsymbol{\mu}_{\boldsymbol{w}|\boldsymbol{o}} = \sum_{k=1}^{K} z^{[k]} \boldsymbol{\mu}_{\boldsymbol{w}|\boldsymbol{o}}^{[k]}.$$

The remaining updates are listed in the appendix.

## IV. Results

We evaluate our method on two real robot tasks. In the first task the robot played a table tennis game and we recorded the Cartesian coordinates of a racket mounted at its end-effector and the Cartesian coordinates of the ball. A Barrett WAM anthropomorphic arm was used for this experiment [16]. The robot provides regular updates about its joint positions at a rate of 1KHz that are used by the forward kinematics to compute the Cartesian position of the racket. The ball is tracked by a high-speed, multi-camera vision system [14] that provides updates at a rate of 200Hz. The extracted dataset contains twenty ball and racket trajectories.

In the second task we placed an obstacle in front of a KUKA lightweight arm and demonstrated by kinesthetic teaching different ways to approach a desired target point in Cartesian space. During the demonstrations we avoided hitting the obstacle and we bypassed it either by moving to the left or to the right. The demonstrations are depicted in Fig. 5. For this experiment we recored the Cartesian position and orientation of the end-effector. The state vector $\boldsymbol{y}_t$ for this experiment is seven dimensional, three dimensions for the position and four for the quaternion based orientation.

### A. Summary of the investigated features

We compare the proposed model, denoted as Latent Manifold ProMPs (LMProMPs) in the figures, to the standard ProMP approach in the two robotic setups.

In the table tennis scenario we investigate the effect of noise and missing data on predicting the final ball impact location at the opponent's side of the table and we demonstrate how the learned latent variables can be used to semantically analyze the data.

Additionally, we demonstrate the beneficial properties of the mixture model in representing the bi-modal distribution required to successfully execute the KUKA reaching task. We use the learned mixture model to generate trajectories to new target locations, not encountered during training, and execute them on the real robot. We demonstrate that our proposed approach successfully avoids the obstacle, while the standard ProMPs average over the two modes and the generalization fails.

In both experiments we used linear regression to compute the feature weights $\boldsymbol{w}$ and we subsequently applied a principal component analysis. We initialized our model with the

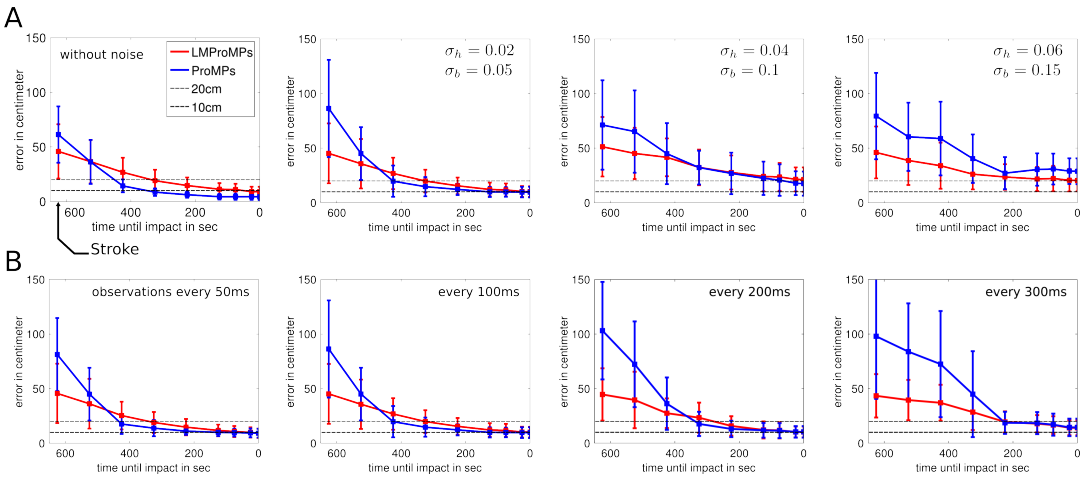


Fig. 4. The effect of noise (A) and missing data (B) on the prediction performance of ProMPs (blue lines) and LM-ProMPs (red lines). In (A), from left to right the amount of applied noise is increased. In (B) four different frame rates of observations ($\in \{50, 100, 200,$ and $300\}$ms) are investigated.

first ten principal components.

### B. The effect of noise and missing data

We use the table tennis setup to predict the final impact location of the ball at the opponent's court. We evaluate our prediction by computing the Euclidean distance in the x,y-plane to the true impact location. The dataset used for learning is shown in Fig. 2(A-B). It should be noted that the colors (red and blue) in Fig. 2 are only used for the visualization as no labels were used for modeling the data.

For a baseline comparison we trained the ProMPs on the same data. The learned distributions over trajectories for ProMPs are illustrated for three Cartesian coordinates in Fig. 2(C-E). We denote the mean of the trajectory distribution with a solid black line and the standard deviation by the shaded region.

In the collected dataset, the robot returns the ball within 550ms to 650ms in advance to the final ball impact. In our comparison, we analyze the prediction performance with respect to the time until the impact event, where we focus on the movement phase right after the stroke, $\approx 625$ms before the end. We used leave-one-out cross-validation to compute the test error.

A fast multi-camera vision setup, good lighting conditions, and access to the opponents sensor readings are amenities we can not always afford. Therefore, we simulate the effect of noisy and incomplete observations, and we evaluate their impact on the prediction performance. First, we add zero-mean Gaussian observation noise to the Cartesian coordinates of the racket and to the Cartesian coordinates of the ball. The standard deviation of the noise used in our evaluation is $\sigma_h \in 10^{-2}\{0, 2, 4, 6\}$ and $\sigma_b \in 10^{-2}\{0, 5, 10, 15\}$ for the racket and the ball, respectively. The results are illustrated in Fig. 4(A), where we show the advantage of the learned prior distribution using latent variables.

Additionally, we evaluate the effect of sparse observations using different sampling intervals, $\{50, 100, 200,$ and $, 300\}$ms. The proposed model is more robust with respect to sparse observations, whereas the standard ProMPs overfit to the training data, especially in the early phase of the movement. The performance comparison of the two approaches is illustrated in Fig. 4(B).

### C. Analyzing the model parameters

As opposed to most movement primitive approaches, our model has only one free parameter to choose that is the precision of the data denoted by $\beta$. For large $\beta$ values the number of contributing latent variables in the generative model is increased, and, at some point, the model will overfit to the training data. To analyze this effect, we approximate the complexity of the learned model by computing the rank of the linear feature weights denoted by $\boldsymbol{M}\boldsymbol{h}^{[i]}$ in Eq. (4).

For values of $\beta \in \{1, 10, 50, 100, 200, 500, 1000, 5000\}$ we compute the training and test error. The prediction performance is shown in Fig. 3(A). The lowest test error was achieved for $\beta = 10$ (for a prediction horizon of 625ms). Note that the test error will not converge to zero due to noise introduced with $\sigma_h = 0.02$ and $\sigma_b = 0.05$, and the sparse observations at 50ms intervals.

The numerical stability of the LMProMPs can be increased with the addition of a gamma prior on the $\lambda^{[v]}$ parameters, discussed in Subsection III-A. To investigate the influence of this regularization on the test error, we evaluated gamma priors with a constant mean ($c_0/d_0 = 100$) and increasing precision in the interval $c_0 \in [0.05, 500]$. For small values of $c_0$ the prior converges to a uniform distribution. For $c_0 \geq 1$ the variational updates were numerically stable and the gamma prior had only little influence on the test error, as shown in Fig. 3(B).

Finally, we semantically analyze the table tennis dataset to evaluate how the latent variable affect the learned prior distribution. We trained the model with 10-dimensional latent variables $\boldsymbol{h}^{[i]}$ in Eq. (4). The effect of the first two latent dimensions in the generative model is illustrated in Fig. 3(C-D). The two latent dimensions of the model affect the slope

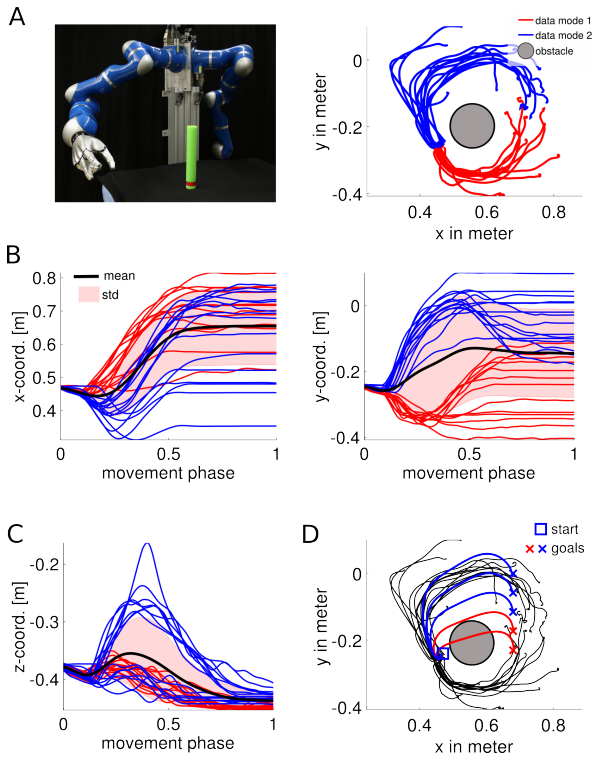


Fig. 5. (A) Experimental setting and two dimensions out of the 7-dimensional dataset (three end-effector coordinates and the four dimensional quaternions). The colors (red and blue) denote the movement direction to avoid the obstacle. (B-C) Learned distributions using ProMPs. The mean is denoted by the black line and the standard deviation by the shaded region. ProMPs cannot represent the bi-modal distribution in the 2nd panel in (B) and the conditioning on unseen targets might fail (D).

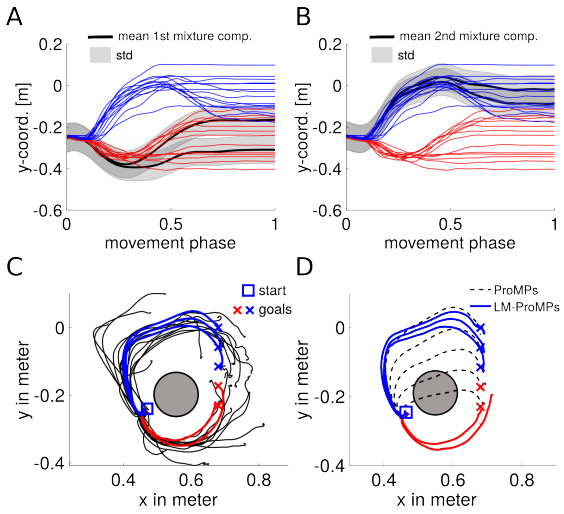


Fig. 6. Learned bi-modal distribution (the colors red and blue denote the modes) using the proposed mixture model with two mixture components (A-B). The latent variable is used to specialize on subregions within the distribution of the mixture component. This is illustrated for two dimensions of $\mathbf{h}$, where solid black lines denote the mean. (C) Conditioning result using LMProMPs. (D) Real robot results.

and the waviness of the x-coordinate of the racket trajectories shown in Fig. 3(D).

### D. Learning bi-modal trajectory distributions

To demonstrate that LMProMPs can model multi-modal distributions, we study demonstrations of a bi-modal target-reaching task. A KUKA lightweight arm was used to reach for different target locations on a table while avoiding an obstacle. We used kinesthetic teaching and we demonstrated two different ways to approach the target. The setup and demonstrations are shown in Fig. 5(A).

For a comparison, we trained ProMPs to learn from the demonstrations, which were unable to represent the two modes. As a result, generalization by conditioning to not encountered target locations may result in trajectories that pass through the obstacle. The learned distributions and example trajectories are shown in Fig. 5(B-C).

In contrast, the LMProMPs model is able to capture the two modes of the demonstrations, as shown in Fig. 6. We initialized the experiment with *K-means* clustering method using two components. The learned prior distribution and the influence of the first two dimensions of the latent variable are illustrated in Fig. 6(A-B). Each mixture component specializes on one mode of the data. Using the learned bi-modal prior distribution, our model is able to generate trajectories to new target locations that avoid the obstacle as shown in Fig. 6(C). The inferred trajectories are smooth and can be executed on the real robot using inverse kinematics to obtain a reference joint trajectory and inverse dynamics control to execute it. The resulting trajectories of the end-effector of the real robot are illustrated in Fig. 6(D).

## V. CONCLUSION

A desired feature of motor control approaches is to have a low number of control parameters that can be used to adapt learned skills to new or changing situations. In existing movement primitive approaches [17], [8], [9] these control parameters are predefined and can not adapt to the complexity of the tasks. In this paper we proposed a probabilistic movement primitive representation with hierarchical priors that learns these control parameters as well as distributions over trajectories from demonstrations. We demonstrated on two kinesthetic teaching datasets that the control variables can be used to generate new trajectories or to analyze the data. The model naturally extends to mixture models, where multi-modal distributions can be represented. In future work we will investigate non-parametric variants using, e.g., Dirichlet processes on more challenging simulated and real-robot tasks with a larger number of modes.


## ACKNOWLEDGMENT

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreements No. 270327 (CompLACS) and No. 600716 (CoDyCo). The authors would like to thank Guilherme Maeda, Rudolf Lioutikov and Katharina Muelling for their assistance in collecting data used in this manuscript.

## APPENDIX
## REMAINING VARATIONAL UPDATES

The remaining variational distributions for the parameter precision and the precision of the projection matrix read $q(\bar{\alpha}) = \Gamma(\bar{\alpha}|\bar{a},\bar{b})$ and $q(\bar{\lambda}^{[1:V]}) = \prod_{v=1}^{V}\Gamma(\bar{\lambda}^{[v]}|\bar{c},\bar{d})$. In the following we list the remaining update equations.

$$\boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}} = \bar{\alpha}_k \mu_{z_k^{[i]}} \boldsymbol{\Sigma}_{\boldsymbol{h}_k^{[i]}} \bar{\boldsymbol{M}}_k^T \left(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}_k}\right),$$

$$\boldsymbol{\Sigma}_{\boldsymbol{h}_k^{[i]}} = \left(\bar{\alpha}_k \mu_{z_k^{[i]}} \left(\bar{\boldsymbol{M}}_k^T \bar{\boldsymbol{M}}_k + d\boldsymbol{\Sigma}_{\boldsymbol{M}_k^{[V]}}\right) + \boldsymbol{I}\right)^{-1},$$

$$\boldsymbol{\mu}_{\boldsymbol{b}_k} = \sigma_{\boldsymbol{b}_k} \bar{\alpha}_k \boldsymbol{I} \left(\sum_{i=1}^{L} \mu_{z_k^{[i]}} \left(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \bar{\boldsymbol{M}}_k \boldsymbol{\mu}_{\boldsymbol{h}_k^{[i]}}\right)\right),$$

$$\sigma_{\boldsymbol{b}_k} = \left(1 + \sum_{i=1}^{L} \bar{\alpha}_k \mu_{z_k^{[i]}}\right)^{-1},$$

$$\boldsymbol{\mu}_{\boldsymbol{m}_k^{[v]}} = \sigma_{\boldsymbol{m}_k^{[v]}} \bar{\alpha}_k \boldsymbol{I} \left(\sum_{i=1}^{L} \mu_{z_k^{[i]}} \mu_{h_k^{[v,i]}} \left(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}_k}\right)\right),$$

$$\sigma_{\boldsymbol{m}_k^{[v]}} = \bar{\alpha}_k \boldsymbol{I} \left(\sum_{i=1}^{L} \mu_{z_k^{[i]}} \left(\mu_{h_k^{[v,i]}}\right)^2 + \left(\sigma_{h_k^{[v,i]}}\right)^{-1}\right),$$

$$\bar{c}_k^{[v]} = c_0 + d/2,$$

$$\bar{d}_k^{[v]} = d_0 + 1/2 \left(\boldsymbol{\mu}_{\boldsymbol{m}_k^{[v]}}^T \boldsymbol{\mu}_{\boldsymbol{m}_k^{[v]}} + d\,\sigma_{\boldsymbol{m}_k^{[v]}}\right),$$

$$\bar{\lambda}_k^{[v]} = \bar{c}_k^{[v]} / \bar{d}_k^{[v]},$$

$$\bar{a}_k = a_0 + d/2 \sum_{i=1}^{L} \mu_{z_k^{[i]}},$$

$$\bar{b}_k = b_0 + 1/2 \sum_{i=1}^{L} \mu_{z_k^{[i]}} \Big( \boldsymbol{C} + tr[\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}] + d\sigma_{\boldsymbol{b}} + \boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}^T d\boldsymbol{\Sigma}_{\boldsymbol{M}^{[V]}} \boldsymbol{\mu}_{\boldsymbol{h}^{[i]}} + tr[\boldsymbol{Q}] \Big),$$

$$\bar{\alpha}_k = \bar{a}_k / \bar{b}_k,$$

$$\mu_{z_k^{[i]}} = \rho_k^{[i]} \left(\sum_{j=1}^{K} \rho_j^{[i]}\right)^{-1},$$

$$\rho_k^{[i]} = \exp\Big( \log \pi_k + d/2[\Psi(\bar{a}_k) - \log \bar{b}_k] - \bar{\alpha}_k/2\, \boldsymbol{C} + tr[\boldsymbol{\Sigma}_{\boldsymbol{w}^{[i]}}] + d\left(\sigma_{\boldsymbol{b}} + \boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}^T \boldsymbol{\Sigma}_{\boldsymbol{M}^{[V]}} \boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}\right) + tr[\boldsymbol{Q}] \Big),$$

$$\pi_k = 1/L \sum_{i=1}^{L} \mu_{z_k^{[i]}},$$

where $d = D \cdot J$,

$$\boldsymbol{C} = \left(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}} - \bar{\boldsymbol{M}} \boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}\right)^T \left(\boldsymbol{\mu}_{\boldsymbol{w}^{[i]}} - \boldsymbol{\mu}_{\boldsymbol{b}} - \bar{\boldsymbol{M}} \boldsymbol{\mu}_{\boldsymbol{h}^{[i]}}\right),$$

$$\boldsymbol{\Sigma}_{\boldsymbol{M}_k^{[V]}} = \begin{bmatrix} \sigma_{\boldsymbol{m}_k^{[1]}} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_{\boldsymbol{m}_k^{[V]}} \end{bmatrix},$$

and

$$\boldsymbol{Q} = \left(\bar{\boldsymbol{M}}^T \bar{\boldsymbol{M}} + d\boldsymbol{\Sigma}_{\boldsymbol{M}^{[V]}}\right) \boldsymbol{\Sigma}_{\boldsymbol{h}^{[i]}}.$$

## REFERENCES


[1] Jonathan Baxter. A model of inductive bias learning. *Journal of Artificial Intelligence Research*, 12:149–198, 2000.

[2] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, New York, USA, 2006.

[3] Hal Daume. Bayesian multitask learning with latent hierarchies. In *Proc. Conf. on Uncertainty in Artificial Intelligence*, 2009.

[4] Andrea d'Avella, Philippe Saltiel, and Emilio Bizzi. Combinations of Muscle Synergies in the Construction of a Natural Motor Behavior. *Nature*, 6(3):300–308, March 2003.

[5] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[6] Theodoros Evgeniou and Massimiliano Pontil. Regularized multi-task learning. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2004.

[7] Denis Forte, Andrej Gams, Jun Morimoto, and Aleš Ude. On-line motion synthesis and adaptation using a trajectory database. *Robotics and Autonomous Systems*, 60(10):1327–1339, 2012.

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

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

[10] J. Kober, E. Oztop, and J. Peters. Reinforcement Learning to adjust Robot Movements to New Situations. In *Proceedings of the Robotics: Science and Systems Conference (RSS)*, 2010.

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

[12] Abhishek Kumar and Hal Daume. Learning task grouping and overlap in multi-task learning. In *Proceedings of the 29th international conference on Machine Learning*, 2012.

[13] A. Kupcsik, M. P. Deisenroth, J. Peters, and G. Neumann. Data-Efficient Contextual Policy Search for Robot Movement Skills. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2013.

[14] C.H. Lampert and J. Peters. Real-time detection of colored objects in multiple camera streams with off-the-shelf hardware components. *Journal of Real-Time Image Processing*, 2012.

[15] Alessandro Lazaric and Mohammad Ghavamzadeh. Bayesian multi-task reinforcement learning. In *ICML '10 Proceedings of the 27th international conference on Machine Learning*, 2010.

[16] K. Mülling, J. Kober, and J. Peters. A Biomimetic Approach to Robot Table Tennis. *Adaptive Behavior Journal*, (5), 2011.

[17] A. Paraschos, C. Daniel, J. Peters, and G Neumann. Probabilistic movement primitives. In *Advances in Neural Information Processing Systems (NIPS), Cambridge, MA: MIT Press.*, 2013.

[18] A. Paraschos, G Neumann, and J. Peters. A probabilistic approach to robot trajectory generation. In *Proceedings of the International Conference on Humanoid Robots (HUMANOIDS)*, 2013.

[19] Alexandre Passos, Piyush Rai, Jacques Wainer, and Hal Daume. Flexible modeling of latent task structures in multitask learning. In *In Proceedings of International Conference on Machine Learning*, 2012.

[20] P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal. Learning and Generalization of Motor Skills by Learning from Demonstration. In *International Conference on Robotics and Automation (ICRA)*, 2009.

[21] P. Rai and H. Daume. Infinite predictor subspace models for multitask learning. In *Int. Conf. on Artificial Intelligence and Statistics*, 2010.

[22] 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.

[23] Paul Ruvolo and Eric Eaton. Online multi-task learning via sparse dictionary optimization. In *Twenty-Eighth AAAI Conference on Artificial Intelligence (AAAI-14)*, July 2014.

[24] A. Ude, A. Gams, T. Asfour, and J. Morimoto. Task-Specific Generalization of Discrete and Periodic Dynamic Movement Primitives. *Trans. Rob.*, (5), October 2010.

[25] Ya Xue, Xuejun Liao, Lawrence Carin, and Balaji Krishnapuram. Multi-task learning for classification with dirichlet process priors. *Journal of Machine Learning Research*, 8:2007, 2007.

[26] Kai Yu, Voker Tresp, and Anton Schwaighofer. Learning gaussian processes from multiple tasks. In *ICML '05 Proceedings of the 22nd international conference on Machine learning*, pages 1012 – 1019, 2005.