

Probabilistic movement primitives for coordination of multiple human–robot collaborative tasks

Guilherme J. Maeda¹ · Gerhard Neumann¹ · Marco Ewerton¹ ·
Rudolf Lioutikov¹ · Oliver Kroemer² · Jan Peters^{1,3}

Received: 29 March 2015 / Accepted: 10 February 2016
© Springer Science+Business Media New York 2016

Abstract This paper proposes an interaction learning method for collaborative and assistive robots based on movement primitives. The method allows for both action recognition and human–robot movement coordination. It uses imitation learning to construct a mixture model of human–robot interaction primitives. This probabilistic model allows the assistive trajectory of the robot to be inferred from human observations. The method is scalable in relation to the number of tasks and can learn nonlinear correlations between the trajectories that describe the human–robot interaction. We evaluated the method experimentally with a lightweight

robot arm in a variety of assistive scenarios, including the coordinated handover of a bottle to a human, and the collaborative assembly of a toolbox. Potential applications of the method are personal caregiver robots, control of intelligent prosthetic devices, and robot coworkers in factories.

Keywords Movement primitives · Physical human–robot interaction · Imitation learning · Mixture model · Action recognition · Trajectory generation

This is one of several papers published in *Autonomous Robots* comprising the “Special Issue on Assistive and Rehabilitation Robotics”.

Electronic supplementary material The online version of this article (doi:[10.1007/s10514-016-9556-2](https://doi.org/10.1007/s10514-016-9556-2)) contains supplementary material, which is available to authorized users.

✉ Guilherme J. Maeda
maeda@ias.tu-darmstadt.de

Gerhard Neumann
neumann@ias.tu-darmstadt.de

Marco Ewerton
ewerton@ias.tu-darmstadt.de

Rudolf Lioutikov
lioutikov@ias.tu-darmstadt.de

Oliver Kroemer
okroemer@usc.edu

Jan Peters
peters@ias.tu-darmstadt.de; jan.peters@tuebingen.mpg.de

¹ Technische Universitaet Darmstadt, Darmstadt, Germany

² University of Southern California, California, USA

³ Max Planck Institute for Intelligent Systems, Tuebingen, Germany

1 Introduction

Assistive and collaborative robots must have the ability to physically interact with the human, safely and synergistically. Amongst the several challenges posed by physical human–robot interaction, this paper focuses on the problems of recognizing the human action and generating the respective movement of the robot assistant. An example is shown in Fig. 1a where a robot must decide if it should hand over a screwdriver or hold part of the assembly to help the human coworker. Once the human action is recognized, the adaptation of the corresponding robot skill must be taken into account. This is illustrated in Fig. 1b where an assistive robot must adapt its trajectory to hand over a bottle at the location of the human hand.

Pre-programming an assistive robot at home or a collaborative robot in a factory for all possible tasks that a human may eventually need help with is, however, unfeasible. The robot must be easy to program, without requiring the need of a dedicated expert. For this reason, we advocate *interaction learning* as a data-driven approach based on the application of imitation learning (Schaal 1999) in the context of human–robot interaction.

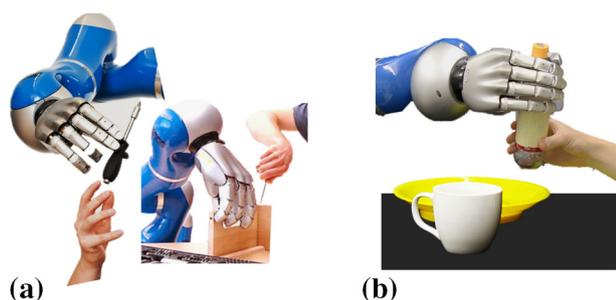


Fig. 1 Algorithms for collaborative and assistive robots must address both action recognition and movement coordination. **a** A robot coworker must recognize the intention of the human to decide if it should hand over the screwdriver or hold the box. **b** An assistive robot must coordinate the location of the handover of a bottle in respect to the location of the hand of the human

Complex activities in human–robot interaction involve multiple interaction patterns. Take a manufacturing scenario as an example. While a massive amount of training data can be obtained by observing how coworkers assemble a product several times throughout the day, this data will contain a variety of movements pertaining to different tasks (handover of parts, holding parts together, screwing, etc.). Thus, to be useful, our method must be capable of *learning from unlabeled data*, in an unsupervised fashion.

In physical human–robot interaction, even tasks as simple as the act of passing an object from the robot to a human can be quite challenging. Humans tend to present variability in their movement, both temporally and spatially, and a robot assistant must be able to adapt to such variations. Therefore, the robot must be capable of generalizing among different partners and their uncertainties. Variability and uncertainty in the interaction incites the use of a *probabilistic framework* to encode movements.

The main contribution of this paper is the introduction of a probabilistic framework for interaction learning with movement primitives. In a single framework, our proposed method allows the assistive movement of the robot to be inferred from the recognition of different human actions. In essence, the proposed method uses imitation learning to construct a mixture model of human–robot interaction primitives. Since we use Probabilistic Movement Primitives (Paraschos et al. 2013) the method will be hereinafter referred to as Interaction ProMPs. This probabilistic model provides a prior that can be used for both recognizing the human intent and for generating the corresponding commands for a robot assistant/collaborator.

This paper consolidates the theoretical framework of a mixture of Interaction ProMPs and validates the method in assistive and collaborative tasks. In less detail, parts of this paper have previously appeared in conference proceedings (Maeda et al. 2014; Ewerton et al. 2015) where preliminary versions of our algorithm have been described. The remain-

der of this paper is organized as follows, Sect. 2 describes related work, Sect. 3 describes the proposed method and compares our method with the previous framework of interaction primitives based on dynamical movement primitives (DMPs). Section 4 provides extensive experiments and Sect. 5 discusses the limitations and extensions of the method, followed by conclusions.

2 Related work on action recognition and interaction control

For the purposes of this paper, related work is categorized in three main areas: action recognition, control for interaction, and approaches that combine both the classification of action and the generation of robot movements.

2.1 Action and intention recognition

Hidden Markov Models (HMMs) have been widely used for analyses of interactions between multiple persons with early adoption by the computer vision community (Oliver et al. 2000). More recently, HMMs have been proposed in several works related to human–robot interaction. Lee et al. (2010) used a hierarchical HMM to learn and represent robot behaviors. In their approach, a high-level HMM identifies the current state of the interaction and triggers low-level HMMs which correspond to the robot’s motor primitives. To ensure that the robot adapts to the movement of the human partner, virtual springs were attached between markers on the human body and corresponding positions on the robot. Tanaka et al. (2012) proposed a Markov model to predict the positions of a worker in an assembly line. The space in which the worker moves is discretized into different regions and a Gaussian mixture model was used to relate positions to procedures. Using this information a robot could then deliver tools and parts to a human worker along the assembly line. More recently, a path-map HMM approach was used to model interactions in cooperative tasks in which a backbone of shared hidden states correlates the actions of the interacting agents (Ben Amor et al. 2013).

Besides HMMs, other probabilistic graphical models have also been used to address interaction tasks. Koppula and Saxena (2013) used an augmented Conditional Random Field where the additional nodes and edges of the graph are used to capture sub-activities, human poses, object affordances and object locations over time. Inference on the graphical model, allows a robot to anticipate the human activity. In a separate step, a pre-programmed robot response is selected accordingly such that it can help the human in achieving the recognized task. In contrast to our work, the output of our method is a movement primitive that is intrinsically correlated to the human action and can be used to directly control

the robot. Wang et al. (2013) proposed an intention-driven dynamics model to encode human intentions as latent states in a graphical model. Intentions can be modeled as discrete variables, such as action labels, or continuous variables, such as an object's final position. The transitions between latent states and the mapping from latent states to observations were modeled via Gaussian Processes.

As evidenced by these works, graphical models are powerful tools for classifying interactions. The construction of these models, however, often requires a substantial amount of training data. In particular in human–robot scenarios with many degrees-of-freedom, the acquisition of sufficiently large and general data sets is one of the main drawbacks of the previously cited methods.

2.2 Continuous representation and control of interaction

Besides discrete action recognition, physical human–robot interaction poses the problem of continuous movement control. The dynamics of the interaction need to be specified in a way that allows for robust reproduction of the collaborative task under different external disturbances; a common approach being based on direct force sensing or force emulation. Rozo et al. (2013) proposed a framework for haptic collaboration between a human and a robotic manipulator. Given a set of kinesthetic demonstrations, their method learns a mapping between measured forces and the impedance parameters used for actuating the robot, e.g., the stiffness of virtual springs governing the collaborative task. In another force-based approach, Lawitzky et al. (2012) proposed learning physical assistance in a collaborative transportation task. In the early learning phase, the robot uses the measured force values to follow the human guidance during the task. Recorded force and motion patterns are then used to learn a Hidden Markov Model (HMM) which can predict the human's next action, and over time the robot learns to take over a more active role in the interaction. Kulvicius et al. (2013) also address a transportation task where the two agents are modeled as two point particles coupled by a spring. The forces applied by the other agent tell the robot how to adapt its own trajectory.

Our work differs significantly from the cited works in the sense that our method does not use or emulate force signals, but instead, learns the correlation between the movement of multiple agents. Our method replaces the problem of planning trajectories with the problem of inference, while decreasing force sensing requirements. Also, by correlating movements, we can address general multi-agent interactions where forces are not necessarily the most natural representation, for example, the simple gesture of asking and receiving an object.

In the work of Yamane et al. (2013), a database of human-to-human handovers comprised of several trajectories is constructed offline and queried during the human–robot interaction. In their method, the query to the database is the observation of the human (the one passing the object) and the output is the corresponding trajectory of the robot partner. To make the search fast enough for online adaptation, a hierarchical composition of movements is proposed in the form of a binary tree. Our work relates closely in application when considering only a single interaction primitive. The use of a probabilistic model and ProMPs, however, allows for the efficient encoding of such handover trajectories as normal distributions, while the search on a hierarchical tree is replaced by a much simpler procedure of conditioning the referred distributions in closed form.

Continuous representation of robot movements with Gaussian Mixture Models (GMMs) have shown promise when compared to other regression and movement primitive methods (Calinon et al. 2010, 2012). Although this paper also uses a mixture of Gaussians, the purposes of GMMs are different. In the body of work of Calinon et al., GMMs are used to encode variations along multiple demonstrated trajectories of a *single task*. The flexibility provided by multiple components are exploited by a stiffness controller which adapts the robot behavior as a function of the local uncertainty over the distribution of demonstrated trajectories. The present paper focus on the action recognition and trajectory generation problem in a *multiple task* scenario where each mixture component encodes the distribution of whole trajectories, as opposed to parts of it. The feedback controller of the robot is assumed given. The parameters of each component allows our method to recognize the action and to condition the movement of one agent given the observation of another. A common feature of both methods is that the inference step can be seen as a full-state estimator, in which any input/output combination is possible.

2.3 Hybrid approaches for action recognition and robot control

While very successful for classifying actions, the use of graphical models and HMMs for motion generation is not straightforward. For example, the use of a HMM with discrete states, although very successful in action classification, introduces artifacts into the motion generation (Ben Amor et al. 2013). As a consequence, the generation of trajectories for the continuous dynamic control of the robot is usually addressed by a different level of representation (e.g. a lower-level HMM (Lee et al. 2010) or movement primitives). Bonilla and Asada (2014) present a hybrid design for a robot to be used on the shoulder. In their work, Petri Nets accounts for discrete control transitions while at the motion

level, Partial Least Squares Regression has been used to find the best action of the robot at future time steps.

Here, we propose a framework based solely on a continuous movement representation that is used to both recognize actions and to generate trajectories in the form of movement primitives. We leverage on DMP-based interaction primitives (Ben Amor et al. 2014) and ProMPs (Paraschos et al. 2013). Despite few similarities between the interaction models, Sect. 3.4 will discuss the advantages of using ProMPs instead of DMPs in detail.

Compared to our work, a similar human prediction step was presented by Mainprice and Berenson (2013). In the cited work, the intrinsic correlation of the movements of different agents are not exploited while here, the inference of the robot trajectory and the recognition of the human action are parametrically correlated. As a result, in the work of Mainprice & Berenson, an independent motion planning procedure—using STOMP (Kalakrishnan et al. 2011)—had to be used specifically to generate the robot trajectories once the intent of the human was recognized.

3 Probabilistic movement primitives for human–robot interaction

In this section, the basic concepts of Probabilistic Movement Primitives for a single degree-of-freedom (DoF) are introduced. Next, the multi-DoF case in the context of human–robot interaction, namely Interaction ProMPs, will follow naturally. Finally, a multi-modal algorithm to compute a mixture of interaction primitives is presented.

3.1 Probabilistic movement primitives

Probabilistic movement primitives represent a distribution over trajectories that are correlated spatially and temporally. Let us generically refer to each joint or Cartesian state of a human or a robot as a DoF. At time step t , each DoF is represented by its position q_t and velocity \dot{q}_t .

For a single DoF we denote $\mathbf{y}_t = [q_t \ \dot{q}_t]^T$ and a trajectory of length T as a sequence $\mathbf{y}_{1:T}$. Assuming a smooth trajectory, a parameterization of $\mathbf{y}_{1:T}$ in a lower dimensional weight space can be achieved by linear regression on N Gaussian basis functions, here denoted as $\boldsymbol{\psi}$. Thus,

$$\mathbf{y}_t = \begin{bmatrix} q_t \\ \dot{q}_t \end{bmatrix} = \begin{bmatrix} \boldsymbol{\psi}_t^T \\ \dot{\boldsymbol{\psi}}_t^T \end{bmatrix} \mathbf{w} + \boldsymbol{\epsilon}_y, \quad (1)$$

and

$$p(\mathbf{y}_t | \mathbf{w}) = \mathcal{N}(\mathbf{y}_t | \boldsymbol{\Psi}_t^T \mathbf{w}, \boldsymbol{\Sigma}_y), \quad (2)$$

where $\boldsymbol{\Psi}_t = [\boldsymbol{\psi}_t, \dot{\boldsymbol{\psi}}_t]$ is a $N \times 2$ dimensional time-dependent basis matrix and $\boldsymbol{\epsilon}_y \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_y)$ is zero-mean i.i.d. Gaussian noise. The probability of observing the whole trajectory is then

$$p(\mathbf{y}_{1:T} | \mathbf{w}) = \prod_1^T \mathcal{N}(\mathbf{y}_t | \boldsymbol{\Psi}_t \mathbf{w}, \boldsymbol{\Sigma}_y). \quad (3)$$

Similar to DMPs, the speed of the execution of the movement is decoupled from the speed of the original trajectory by using an artificial “clock”, known as the phase variable z . Effectively, the phase variable replaces the time in order to control the location of the basis functions with $\boldsymbol{\psi}(z)$ and is commonly used to synchronize multiple DoFs of the same robot or DoFs of multiple agents. Under the assumption that dynamic constraints (such as torque limits) are satisfied, the phase can be used to impose arbitrary velocity profiles on the trajectory. For simplicity, in this article we will assume the phase of the model is identical to the timing of the demonstration such that $z_t = t$ and $\boldsymbol{\psi}_t = \boldsymbol{\psi}(z_t)$. Note however that any monotonically increasing function can be used (Paraschos et al. 2013).

In general, ProMPs are learned from multiple demonstrations. The diversity of demonstrations is supposed to reveal the variance of the task, the uncertainty of the execution, as well as to introduce exploration noise when required. Multiple demonstrations are assumed to have the same temporal phase. When this is not the case, trajectories must be time-aligned. Since the time-alignment of training data can be seen as a separate problem, we describe in the Appendix a particular algorithm for time-alignment that was used during experiments.

Assume M trajectories are obtained via demonstrations; their parameterization leading to a set of weight vectors $\mathbf{W} = \{\mathbf{w}_1, \dots, \mathbf{w}_m, \dots, \mathbf{w}_M\}$. Define a learning parameter $\boldsymbol{\theta}$ to govern the distribution of \mathbf{W} such that $\mathbf{w} \sim p(\mathbf{w}; \boldsymbol{\theta})$. A distribution of trajectories is obtained by integrating out \mathbf{w} ,

$$p(\mathbf{y}_{1:T}; \boldsymbol{\theta}) = \int p(\mathbf{y}_{1:T} | \mathbf{w}) p(\mathbf{w}; \boldsymbol{\theta}) d\mathbf{w}. \quad (4)$$

We model $p(\mathbf{w})$ as a Gaussian with mean $\boldsymbol{\mu} \in \mathbb{R}^N$ and covariance $\boldsymbol{\Sigma} \in \mathbb{R}^{N \times N}$, that is $\boldsymbol{\theta} = \{\boldsymbol{\mu}, \boldsymbol{\Sigma}\}$, computed from the training set \mathbf{W} . The fidelity with which the distribution of trajectories in (4) captures the true nature of a task clearly depends on how $\boldsymbol{\theta}$ controls the distribution of weights. The assumption of a normal distribution will lead to limitations to be discussed and addressed by a mixture of models in Sect. 3.3.

One of the basic operations of ProMPs is to compute a posterior probability of the weights $p(\mathbf{w}; \boldsymbol{\theta}^+)$ conditioned on a, possibly sparse and asynchronous, sequence of observations $\mathbf{y}_{t':t}^*$. Conditioning can be achieved off-line and in closed-form with

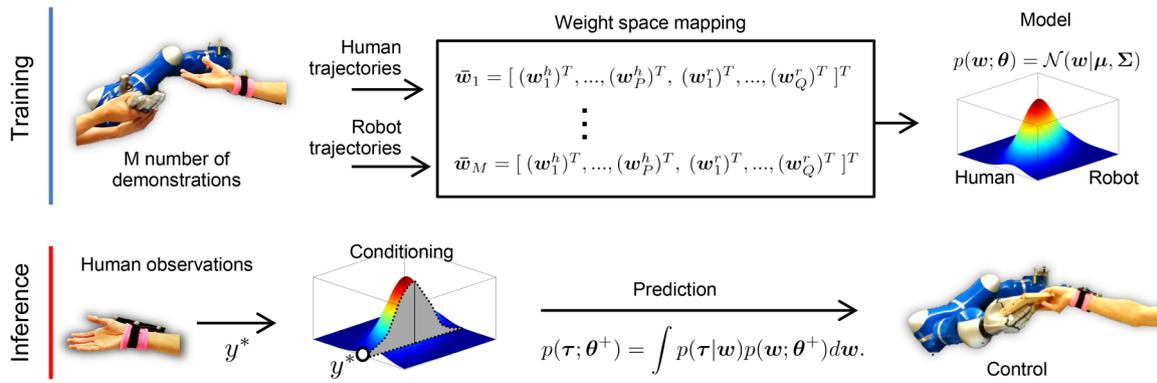


Fig. 2 Illustration of the workflow of Interaction ProMP where the distribution of human–robot parameterized trajectories is abstracted to a single bivariate Gaussian. The conditioning step is shown as the slicing

of the distribution at the observation of the human. In the real case, the distribution is multivariate and correlates all the weights of all demonstrations

$$\begin{aligned} \mu^+ &= \mu + K(y_{t:t'}^* - H_{t:t'}^T \mu), \\ \Sigma^+ &= \Sigma - K(H_{t:t'}^T \Sigma), \end{aligned} \tag{5}$$

$$\bar{w} = [(w_1^H)^T, \dots, (w_P^H)^T, (w_1^R)^T, \dots, (w_Q^R)^T]^T. \tag{8}$$

where $K = \Sigma H_{t:t'} (\Sigma_y^* + H_{t:t'}^T \Sigma H_{t:t'})^{-1}$ and Σ_y^* is the measurement noise. The upper-script $(\cdot)^+$ represents the values after the update. The observation matrix $H_{t:t'}^T$ is obtained by concatenating the bases at the corresponding time steps of the observation $H_{t:t'} = [\Psi_t, \dots, \Psi_{t'}]$. Recursive implementation of the conditioning for on-line applications is achieved by conditioning one observation each time, and by using the posterior parameters $\theta^+ = \{\mu^+, \Sigma^+\}$ as the prior of the next observation. In this case $H_t = \Psi_t$.

Similarly to the single DoF case, we compute a normal distribution from a set of M demonstrations $\{\bar{w}_1, \dots, \bar{w}_M\}$ with $\mu \in \mathbb{R}^{(P+Q)N}$ and $\Sigma \in \mathbb{R}^{(P+Q)N \times (P+Q)N}$. In essence, an Interaction ProMP provides a model that correlates the weights that parameterize the trajectories of a human and a robot when executing a task in collaboration. The Interaction ProMP is conditioned on the observations of the human and the robot is controlled based on the posterior distribution over robot trajectories.

3.2 Interaction ProMPs: correlating human and robot movements with probabilistic movement primitives

The conditioning (5) applies with the difference that only the P measurements of the human are provided while the Q DoFs of the robot are inferred. During conditioning, the observation matrix H_t can be represented in observed and unobserved partitions

In the single DoF case the learning parameter θ was used to capture the correlation among the weights of different demonstrations. The key aspect of the Interaction ProMP is to capture the correlation of multiple DoFs of multiple agents.

At each instant t , let us define the state vector as a concatenation of the P DoFs provided by the human, followed by the Q DoFs provided by the robot

$$y_t = [y_{1,t}^H, \dots, y_{P,t}^H, y_{1,t}^R, \dots, y_{Q,t}^R]^T, \tag{6}$$

$$H_t^T = \begin{bmatrix} (\Psi_t^H)^T & \dots & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \ddots & \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \dots & (\Psi_t^H)^T & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \dots & \mathbf{0} & \mathbf{0}^R & \dots & \mathbf{0} \\ \mathbf{0} & \ddots & \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0}^R \end{bmatrix}, \tag{9}$$

where the upper scripts $(\cdot)^H$ and $(\cdot)^R$ refer to the human and robot DoFs, respectively. Similar to the single DoF case, all DoF's trajectories are parameterized as weights,

where each zero entry is of $2 \times N$ dimension, and the human (observed) and robot (estimated) partitions have P and Q number of entries in the diagonal, respectively. When only positions of the human are provided, the feature vector becomes $\Psi_t^H = [\psi_t, \mathbf{0}]$ and velocities are estimated.

$$p(y_t | \bar{w}) = \mathcal{N}(y_t | H_t^T \bar{w}, \Sigma_y), \tag{7}$$

The trajectory distributions that predict the human movement and provides the corresponding reference robot trajectories are obtained by integrating out the weights \bar{w}

where $H_t^T = \text{diag}(\Psi_t^T, \dots, \Psi_t^T)$ has $P+Q$ entries (one entry for each DoF). Each human–robot collaborative demonstration now provides $P+Q$ training trajectories, from which a multi-DoF weight vector \bar{w} is regressed as

$$p(y_{1:T}; \theta^+) = \int p(y_{1:T} | \bar{w}) p(\bar{w}; \theta^+) d\bar{w}. \tag{10}$$

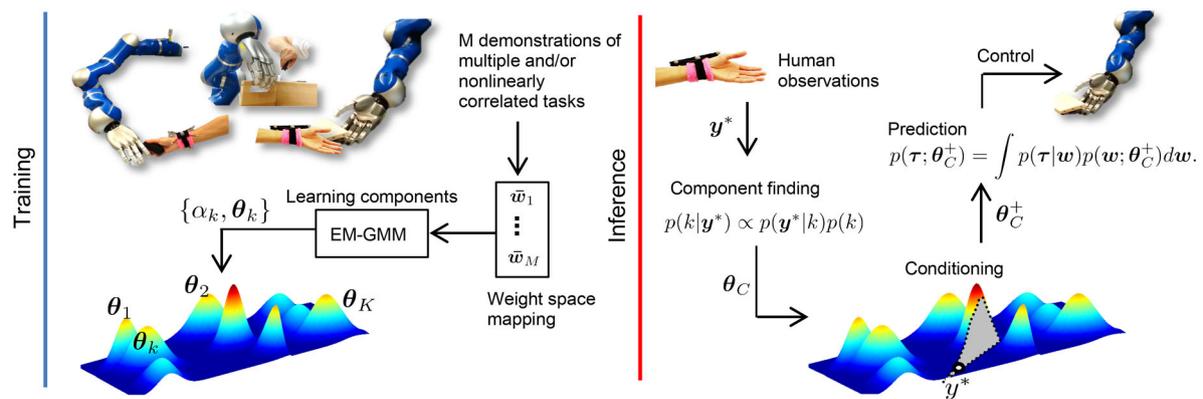


Fig. 3 The workflow of a mixture of Interaction ProMPs where distributions are abstracted as a mixture of bivariate Gaussians. *Left* During the training phase, multiple pairs of human–robot trajectories representing different tasks are collected and mapped into the low-dimensional weight space. Parameters that govern the normal distribution of the

weights are then found by maximizing the likelihood with EM algorithm. *Right* During execution the most probable component is found, which is then conditioned on the current human observation to provide the trajectory for the robot partner

Figure 2 summarizes the workflow of the Interaction ProMP. During the training phase, imitation learning is used to learn the parameter θ that governs the distribution of the weights. In the figure, the distribution is abstracted as a bivariate Gaussian where each of the two dimensions are given by the distribution over the weights of the human and robot trajectories. During the inference phase, the assistive trajectory that the robot must execute is predicted by integrating out the weights of the posterior distribution $p(\bar{\mathbf{w}}; \theta^+)$. The operation of conditioning is illustrated by the slicing of the prior, at the current observation of the human \mathbf{y}^* (the subscript $(\cdot)_t$ will be omitted whenever the specific time instant of the observation is not relevant).

3.3 Mixture of interaction ProMPs

So far, we have developed Interaction ProMPs as a single Gaussian distribution of trajectory weights that correlate human–robot movements. The single distribution has, however, two limitations. First, it assumes that the parameterized trajectories of different agents correlate linearly. While this assumption may hold locally, a single model can not always guarantee global representation. Second, as a single Interaction ProMP only represents one task, multiple tasks require multiple Interaction ProMPs.

These limitations motivate the introduction of a mixture of Interaction ProMPs. The mixture provides means to combine several Interaction ProMPs to represent multi-modal interactions. The algorithm addresses, in the same manner, (1) the case where a nonlinearly correlated task can be approximated by local Interaction ProMPs and (2) multiple tasks, where each task is represented by one or more Interaction ProMP(s).

A mixture of Interaction ProMPs is achieved by learning a Gaussian Mixture Model in the weight space of the

training data using the Expectation-Maximization algorithm (Bishop 2006). The interaction model maintains a set of K learning parameters $\theta_{1:K} = \{\theta_1, \dots, \theta_K\}$. The Interaction ProMP method described in Sect. 3.2 is a particular case of the mixture of Interaction ProMPs for $K = 1$.

Each mixture component is associated with a probability distribution $p(\bar{\mathbf{w}}; \alpha_k, \theta_k)$, where $k \in \{1, \dots, K\}$, $\alpha_k = p(k)$, and $\theta_k = \{\mu_k, \Sigma_k\}$. Here, α_k , μ_k , and Σ_k are the prior probability, the mean and the covariance matrix of the k th mixture component, respectively. The mixture model can be formalized as

$$p(\bar{\mathbf{w}}) = \sum_{k=1}^K p(k)p(\bar{\mathbf{w}}|k) = \sum_{k=1}^K \alpha_k \mathcal{N}(\bar{\mathbf{w}}|\mu_k, \Sigma_k). \quad (11)$$

Figure 3 illustrates the workflow of the method. During the training, the EM algorithm receives unlabeled weights of several demonstrations and locally improves the estimate of the parameters $\{\alpha_k, \theta_k\}$. During inference, the method finds the most probable mixture component based on the current human observation. Given the mixture component, the procedure of conditioning and predicting trajectories is identical to the single Interaction ProMP in Sect. 3.2 and can be achieved with the use of Eqs. (5) and (9). Details of the mixture model are given next.

3.3.1 Learning the mixture model

For a K number of Gaussian mixture components, the method iterates over the Expectation step and the Maximization step until convergence of the probability distribution over the weights. The expectation step computes the responsibilities r_{ik} , where r_{ik} is the probability of cluster k given weight vector $\bar{\mathbf{w}}_i$.

$$r_{ik} = p(k|\bar{\mathbf{w}}_i) = \frac{\mathcal{N}(\bar{\mathbf{w}}_i|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\alpha_k}{\sum_{l=1}^K \alpha_l \mathcal{N}(\bar{\mathbf{w}}_i|\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)}. \quad (12)$$

The maximization step updates the parameters α_k , $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ of each cluster k , using

$$n_k = \sum_{i=1}^n r_{ik}, \quad \alpha_k = \frac{n_k}{n}, \quad (13)$$

$$\boldsymbol{\mu}_k = \frac{\sum_{i=1}^n r_{ik} \bar{\mathbf{w}}_i}{n_k}, \quad (14)$$

$$\boldsymbol{\Sigma}_k = \frac{1}{n_k} \left(\sum_{i=1}^n r_{ik} (\bar{\mathbf{w}}_i - \boldsymbol{\mu}_k)(\bar{\mathbf{w}}_i - \boldsymbol{\mu}_k)^T \right). \quad (15)$$

In our implementation, since the training is done off-line, the number of K components of the mixture is computed with leave-one-out cross-validation. More sophisticated methods for addressing the number of clusters are left for future work. The parameters $\alpha_{1:K}$, $\boldsymbol{\mu}_{1:K}$ and $\boldsymbol{\Sigma}_{1:K}$ are initialized with k-means clustering.

3.3.2 Inference of the assistant's trajectory

Before the conditioning can be executed the most probable component of the mixture given the human observation \mathbf{y}^* must be found. Given a sparse sequence $\mathbf{y}_{t:t'}^*$ and a prior probability $p(k)$, the probability of each component $p(k|\mathbf{y}_{t:t'}^*)$ can be computed with Bayes' theorem

$$p(k|\mathbf{y}_{t:t'}^*) \propto p(\mathbf{y}_{t:t'}^*|k)p(k). \quad (16)$$

Since each component k is governed by $\boldsymbol{\theta}_k$, the likelihood in (16) can be computed with

$$p(\mathbf{y}_{t:t'}^*; \boldsymbol{\theta}_k) = \int p(\mathbf{y}_{t:t'}^*|\mathbf{H}_{t:t'}^T \bar{\mathbf{w}}, \boldsymbol{\Sigma}_y^*) p(\bar{\mathbf{w}}; \boldsymbol{\theta}_k) d\bar{\mathbf{w}} \quad (17)$$

$$= \mathcal{N}(\mathbf{y}_{t:t'}^*|\mathbf{H}_{t:t'}^T \boldsymbol{\mu}, \mathbf{H}_{t:t'}^T \boldsymbol{\Sigma} \mathbf{H}_{t:t'} + \boldsymbol{\Sigma}_y^*). \quad (18)$$

The trajectory is inferred by conditioning the distribution of weights of the most probable mixture component on $\mathbf{y}_{t:t'}^*$ and integrating out the weights

$$p(\mathbf{y}_{1:T}; \boldsymbol{\theta}_C^+) = \int p(\mathbf{y}_{1:T}|\bar{\mathbf{w}}) p(\bar{\mathbf{w}}; \boldsymbol{\theta}_C^+) d\bar{\mathbf{w}}, \quad (19)$$

$$\text{where } C = \arg \max_k p(k|\mathbf{y}_{t:t'}^*). \quad (20)$$

Note that (19) computes the probability of observing the sequence $\mathbf{y}_{t:t'}^*$, rather than each measurement in isolation. Therefore, even if several trajectories of different components largely overlap in space, it is still possible to identify the correct component with high certainty as the *order* at which those measurements are made are also taken into

account. (The interested reader also is referred to (Maeda et al. 2014), where action recognition experiments were conducted in more detail).

3.4 Comparison with a previous interaction primitive approach

Interaction primitives have been previously proposed in combination with dynamical movement primitives (Ben Amor et al. 2014). Similar to the ProMP case, in the framework of Interaction DMPs, a distribution of weights $p(\mathbf{w})$ is learned from several demonstrations of a task. The principal difference is that in the latter, the weights are mapped to the forcing function f_t as opposed to the positions q_t . That is,

$$f_t = \boldsymbol{\psi}_t^T \mathbf{w}, \quad (21)$$

where $\boldsymbol{\psi}_t$ are the normalized Gaussian basis functions. For details on DMPs please refer to the work of Ijspeert et al. (2013) and references therein.

For each DoF, the forcing function adds a nonlinear acceleration term to the movement which complements a linear and stable spring-damper system

$$\ddot{q} = [\alpha_y(\beta_y(g - q) - \dot{q}/\tau) + f_t]\tau^2, \quad (22)$$

where g is the goal attractor, α_y , β_y are user-defined parameters that characterize the spring-damper behavior and τ controls the speed of execution.

Referring back to (5), the conditioning with DMPs is based on the observation of accelerations (or forces) and velocities, that is $\mathbf{y}_t^* = f(\ddot{q}, \dot{q}, \dots, t)^*$. However, in a typical interaction task, the observations of a human partner might arrive at irregular periods of time, for example, when the measurement signal is prone to interruption (a typical case is occlusion in motion capture systems). In such cases, the application of DMPs can become very restrictive and even impractical given that acceleration are often computed from position's second derivatives. The advantage of the Interaction ProMP is that conditioning is made directly on the observed positions of the human. In fact, conditioning can be made even with a single position measurement of the human.

Figure 4 illustrates a simple simulated case where both observed human and controlled robot have a single DoF. The training data was created by sketching pairs of trajectories manually using a computer mouse from which the initial distribution of trajectories of the Interaction ProMP are obtained (blue patches). As shown by the posterior distribution, Interaction ProMPs allow for the prediction of trajectories after observing only four sparse measurements. The same experiment can not be executed with Interaction DMPs as the second order derivative provided by only four

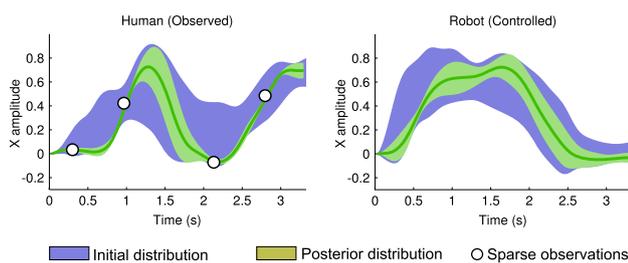


Fig. 4 Simulated case where a human provides only four sparse and asynchronous observations, from which the trajectory of the robot assistant must be inferred. This scenario is tractable with Interaction ProMPs but impractical for Interaction DMPs due to a lack of data for the computation of the DMP forcing function. The patches represent the $\pm 2\sigma$ deviations from the mean

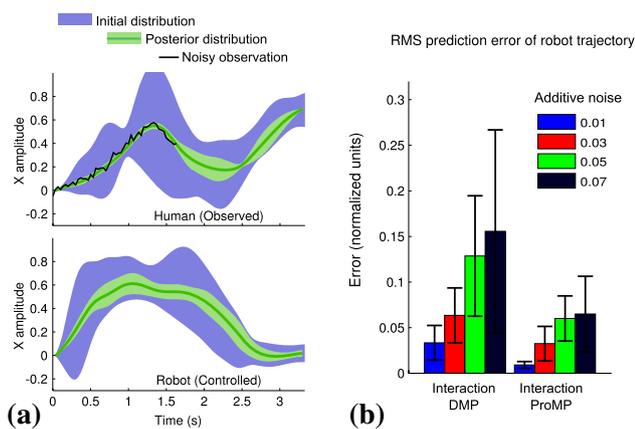


Fig. 5 Interaction ProMPs are more robust to noisy measurements when compared to Interaction DMPs. **a** An example where prediction of the robot movement is obtained from a partial, noisy observation of the human. **b** Root-mean-square prediction error of the movement of robot given increasing levels of noise of the human observation for both Interaction ProMP and DMP

observations introduces inaccuracies on the representation of the true force.

The derivative-free conditioning of the ProMP makes the method also robust to noisy measurements. In Fig. 5a, the Interaction ProMP is conditioned on a constant, synchronous stream of noisy position measurements. For the particular case shown in the figure, the true trajectory is corrupted with an additive Gaussian noise with $\sigma = 0.04$. We quantified the sensitivity to noise in terms of the RMS prediction error of the robot trajectory for both Interaction DMPs and Interaction ProMPs. Both methods were given the same observations, corrupted with increasing levels of additive Gaussian noise. The results are shown in Fig. 5b where each bar represents the average and standard deviation over 10 trials. Human observations were made up to 40% of the total trajectory. Prediction errors with ProMPs are less than half of that obtained with DMPs. It is evident that Interaction ProMPs are much less sensitive to noise than the DMP counterpart.

4 Experiments with interaction ProMPs

Several human–robot experiments were conducted with a 7-DoF KUKA lightweight arm equipped with a 5-finger hand. In the first experiment we quantified the accuracy of the inference achievable by the setup. This assessment allowed us to implement Interaction ProMPs to show its applicability to adapt the shape of the trajectories during an assistive task where the robot hands over a bottle to a human. Next, we show how a mixture of Interaction ProMPs can be used to help a human assemble a box with a robot coworker and also compare it with a baseline lookup table using the same training data. Last, we expose the benefits of the mixture of Interaction ProMPs for encoding a task where the parameters of the trajectory have a multi-modal distribution.

4.1 Interaction ProMPs in the handover context

A handover is comprised of a complex series of combined physical and social interaction steps. As previously investigated (Strabala et al. 2013), these steps range from (1) the social-cognitive cues that establish the connection between the giver and the taker, (2) the coordination of the location and the resulting trajectory as a function of preferences and socially acceptable movements (Sisbot and Alami 2012), and (3) the final physical transfer that comprises interaction forces and compliances (Kupcsik et al. 2015).

Although some of the scenarios here presented ultimately lead to the handover of objects, handovers are not the only application of Interaction ProMPs and our method does not intend to be a self-contained solution to the whole handover problem. Essentially, the Interaction ProMP is a method that represents, recognizes and infers appropriate robot motion commands based on pairs of demonstrated trajectories, thus open to any application that can be framed in this manner (see the case in Sect. 5.3).

While the following experiments are minimalist scenarios designed to evaluate Interaction ProMPs, in a complete framework for handovers the method is particularly suited to address the location coordination problem of the physical channel (as opposed to the social-cognitive channel (Strabala et al. 2013)). Moreover, as an imitation learning method, Interaction ProMPs implicitly encode user preferences from demonstrations which seems to be more suited for human interaction than pure motion planning approaches (Cakmak et al. 2011).

4.2 Evaluation of human–robot coordinated trajectory inference

We evaluated the accuracy of the inference of the Interaction ProMP in our robot setup. The experiment consisted in collecting training data by measuring the XYZ trajectory

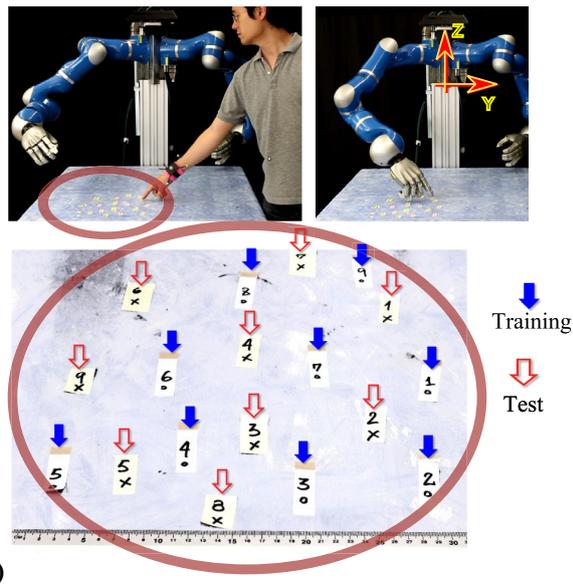


Fig. 6 **a** The experiment proposed to quantify the accuracy of the trajectory inference. An Interaction ProMP is used to correlate the joint trajectories of the robot with the observed trajectory of the human such that both reach the same position on the table. The zoomed view shows the training and test positions. **b** Robot error in reaching at each of the pre-defined test positions

coordinates of the wrist of a human via motion capture when reaching several pre-defined training positions on a table located in front of the robot (human positions were measured in relation to the world reference frame located on the torso of the robot). The robot was also demonstrated how to reach the same positions via kinesthetic teaching. The several pairs of training trajectories were used to create an Interaction ProMP, which could then be conditioned on test positions specified by the human. As shown in Fig. 6a, 9 different reaching positions were used to collect training data, sparsely covering an approximate circular area of diameter 30 cm.

The remaining positions, also shown in Fig. 6, were used for testing. During the inference phase, the human reached at the test positions to provide y^* . These observations were then used to condition the Interaction ProMP to make the robot reach the same positions. Figure 6b shows the distance error on the plane of the table between the position reached by the robot and the true expected position for the test cases.

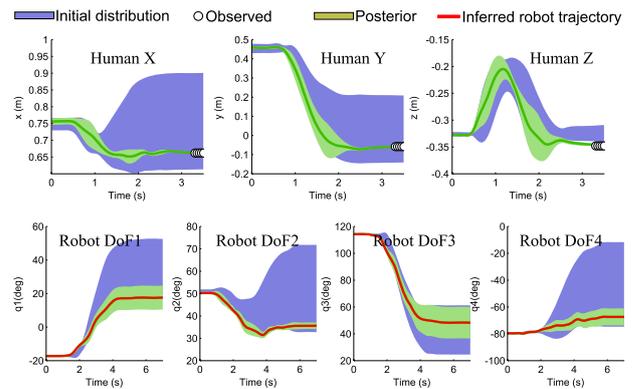


Fig. 7 One of the test cases during the reaching task. *Upper row* shows trajectory distributions of the tracked wrist of the human. The posterior distribution of trajectories collapses at the end, where the observations were made. *Bottom row* shows the first four joints of the 7-DoF robot arm. The mean of the posterior distribution was used as a reference trajectory for a standard tracking controller in joint space

The maximum error was of 3 cm, or 10% in relation to the total area covered by the demonstrations.

Figure 7 shows one example of the inference of the trajectory distribution for one of the test positions. The first row shows the observed coordinates of the wrist. The second row shows the first four joints of the robot, starting from the shoulder. Since we are only interested on the pointed position, the interaction primitive was conditioned on the final measurements of the wrist trajectory (indicated by the collapse in variance at the end of the human movement). The most probable trajectories, given by the mean of the posterior distribution, were used as reference signals for the robot’s tracking controller.

We propose a variant of the reaching experiment to illustrate a practical case of an assistive robot that helps a human reaching for a bottle that is out of his/her reach. We trained an Interaction ProMP similar to the previous case by correlating different positions of a bottle handover. Due to the practicalities of the experiment, we hand-coded extra point-to-point and open-close finger commands before and after the execution of the primitives to account for the grasping and releasing of the bottle. Figure 8a shows as a sequence of snapshots where the robot grasps the bottle from the other side of the table and delivers at the location conditioned by the current position of the human hand. Figure 8b shows several locations of successful hand-overs. As shown by the last picture, due to the inference in the robot’s joint space, it is possible to use the full workspace of the robot, including postures that pass through singularities and would otherwise require special attention when using inverse kinematics.

4.3 A multi-task robot coworker

To demonstrate the use of a mixture of Interaction ProMPs, we set an experiment where the robot plays the role of a

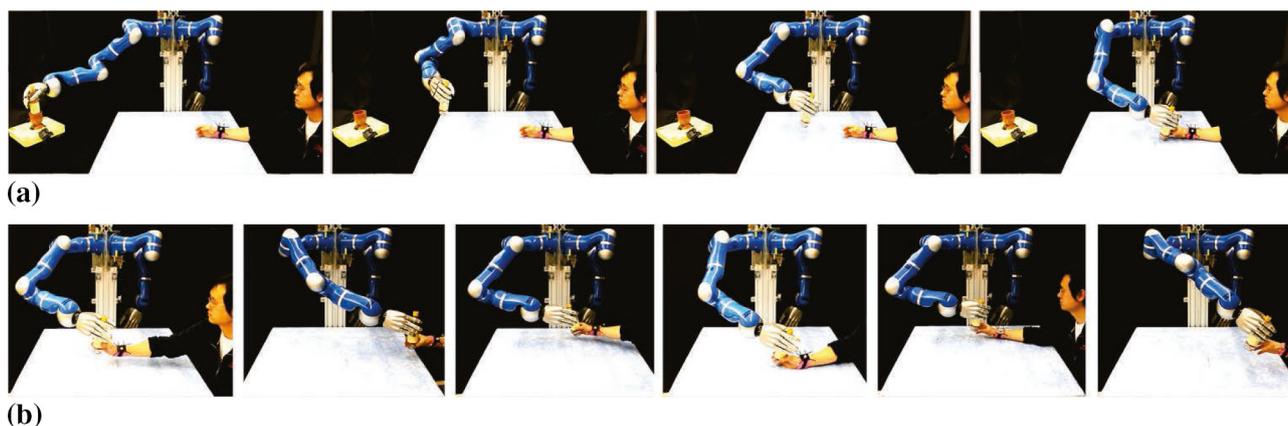


Fig. 8 The robot assists a human by handing over a bottle out of the human reach. **a** A sequence of snapshots where the robot gives the bottle to a human. The location of the handover is conditioned by the location of the human wrist. **b** Different locations of hand-overs provides evi-

dence that the inferred joint trajectories of the robot have been properly correlated with the Cartesian trajectories of the human. The *last picture* shows that correlation in joint space allows for controlling the robot regardless of singularities

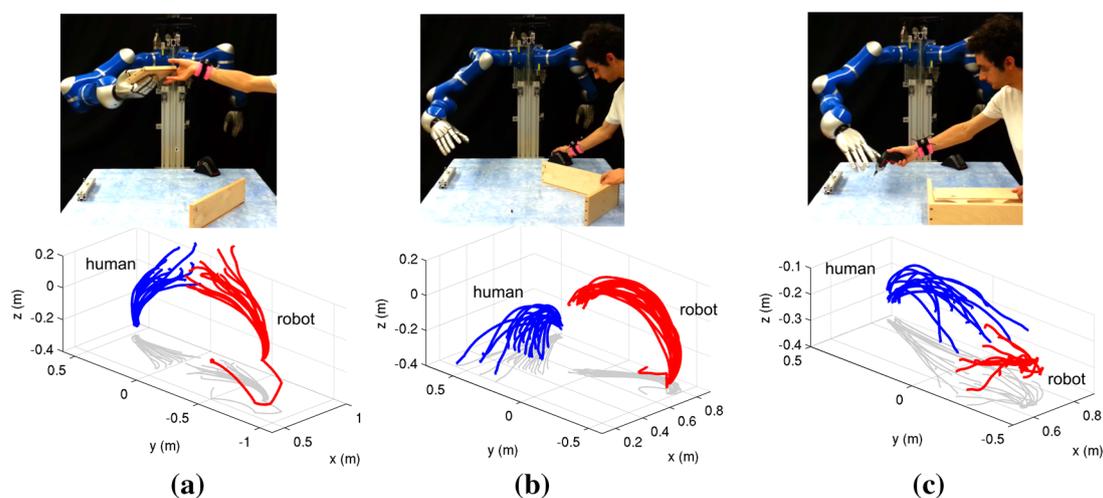


Fig. 9 Demonstrations of the three different interactions and their respective trajectories. All trajectories are fed to the mixture of Interaction ProMPs in batch, without labels

coworker that helps a human assembling a box. The assembly consists of three different collaborative interactions. As shown in the pictures of Fig. 9, in one of them, the human extends his/her hand to receive a plate. The robot fetches a plate from a stand and gives it to the human. In a second interaction the human fetches the screwdriver and the robot grasps and gives a screw to the human as a pre-emptive collaborator would do. The third type of interaction consists of the robot receiving a screwdriver such that the human coworker can have both hands free (the same primitive representing this interaction is also used to give the screwdriver back to the human). Each form of interaction is encoded by its own set of one or more Interaction ProMP component(s).

Each interaction of plate handover, screw handover and holding the screwdriver was demonstrated 15, 20, and 13 times, respectively. As described in Sect. 3.3, all training

data were fed to the algorithm resulting in 48 human–robot pairs of unlabeled demonstrations shown in the upper row of Fig. 10 as the gray curves. The lower row of the same figure shows, as blue patches, the prior distribution of trajectories learned from a mixture model with three components. As an example of the conditioning, the posterior is represented by its mean and by the region inside \pm two standard deviations as a green patch.

To define the appropriate number of mixture components, the RMS prediction error was quantified as a function of the number of clusters (Fig. 11), obtained by leave-one-out cross-validation (LOOCV) over the whole set of demonstrations. As one would expect, since the unlabeled training data contains three distinct interaction patterns, the improvement is clearly visible up to three mixture components. Since no significant improvement was obtained afterwards, the GMM

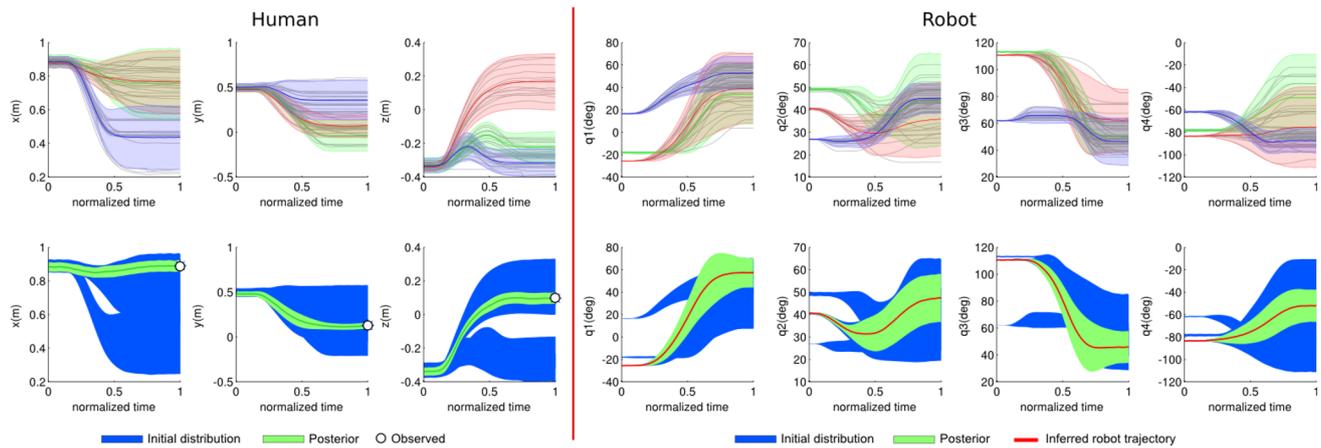


Fig. 10 Upper row Mixture components represented by their mean trajectories and the region inside two standard deviations ($\mu \pm 2\sigma$). Lower row Posterior probability distribution after conditioning on human observations

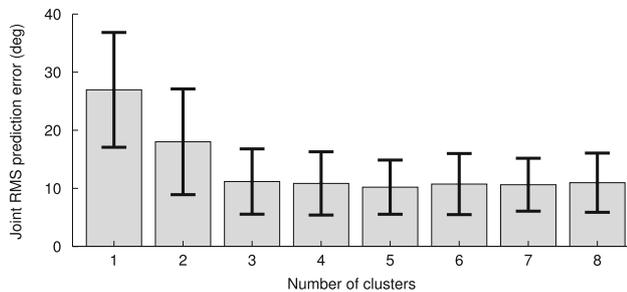


Fig. 11 Root-mean-square error of the joint trajectories (averaged over all tests) using a leave-one-out cross-validation as a function of the number of clusters (mixture components). The plateau after three clusters seem to be consistent with the training data since it consists of three distinct interaction patterns



Fig. 12 Handover of a plate. Conditioning on three different positions of the wrist (using motion capture) of a human coworker

with three mixture components was selected for controlling the robot.

We assembled the toolbox, consisting of seven parts and 12 screws, two times. The experiments demanded more than 40 executions of the mixture of interaction primitives. The successful rate of the correct mixture component was of 100%. As an example, Fig. 12 shows the robot executing the plate handover at three different positions based on the location of the wrist marker. Note that the postures of the arm are very different, although they are all captured by the same mixture component.

4.4 Comparison with a database approach

Using the same training data of the previous experiment we compared our method with a simple database approach. Motion databases are conceptually simple and have multiple applications in robotics (Yamane et al. 2011). Particularly in handover problems, databases have shown suitable due to the degree of similarity amongst different handover trajectories (Yamane et al. 2013) and the limited size of databases built from demonstrations.

The procedure consisted in querying the database as a lookup table with the human observation and returning the robot trajectory that is paired to the closest query. To simplify the analysis and to compare both methods with the same observation, we used the final position of the human marker to query both the database, and to condition the Interaction ProMP. We also limited the study to the trajectories corresponding to the plate handover component. As the test runs off-line, the issues of retrieving trajectories based on partial observations in real time are not considered. In the work of Yamane et al. 2013, efficient online queries led to a complex hierarchical database construction with a sliding window search.

Since the Interaction ProMP and the database return the joint trajectories of the robot, we implemented a high-fidelity, forward kinematic simulator of the robot and hand in V-REP (Rohmer et al. 2013) as shown in Fig. 13a. We then computed the positioning error of the robot hand in relation to the ground-truth test data, from which the human query is obtained. We compared the handover accuracy as a function of the number of demonstrations in the training set.

The procedure consisted in running LOOCV on each of the 15 demonstrations of plate handovers. For each test we then start with a set of 14 demonstrations, from which the

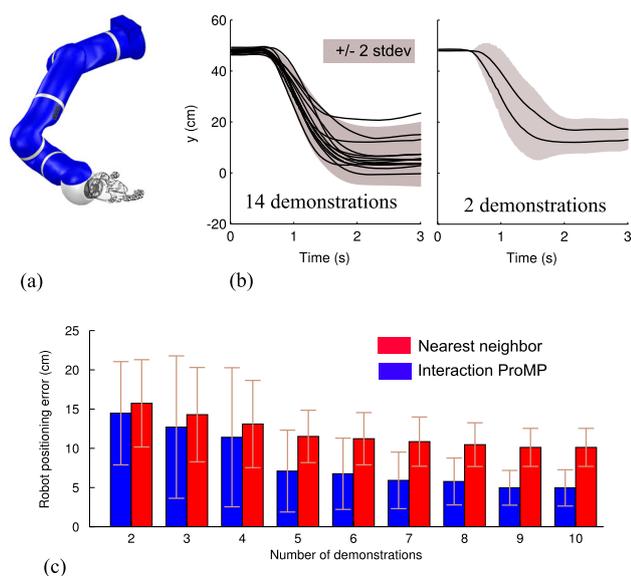


Fig. 13 **a** A high-fidelity kinematic model of the arm and hand used for forward kinematics. **b** Examples of two Interaction ProMPs built with different number of demonstrations. **c** With 10 demonstrations the error of the prediction of the nearest neighbor is two times larger than the Interaction ProMP using the same training dataset

look-up table and the Interaction ProMP were created and queried. We then randomly select one of the demonstrations and removed it from the training set, and the look-up table and the Interaction ProMP were reconstructed and queried again. This procedure was repeated until only two demonstrations were left. The effect of removing data can be seen in Fig. 13b for the Y direction of the human movement.

The bars in Fig. 13c show the average positioning error with \pm one standard deviation. Note that both methods perform roughly equally up to a training set of four demonstrations. The error of the Interaction ProMP gradually decreases to approximately 5 cm when ten demonstrations are used. In the case of nearest neighbor, this accuracy reached a plateau of about 10 cm under the same number of demonstrations. The nearest neighbor suffers from the sampling problem as the error is only decreased when new training is added closer to any of the other already existing data in relation to the query point. An advantage of ProMPs is not only to allow for inference with a small set of demonstrations but also, differently from a database approach, the computational load is not affected by the size of the training set.

4.5 Human–robot tasks with nonlinear correlations

We propose a scenario to expose the issue of the Interaction ProMP when representing a task where the parameterized trajectories of the human and the robot have a clear nonlinear correlation. In the same vein as the first reaching experiment,



Fig. 14 Experimental setup of a toy problem to illustrate the mixture of interaction primitives for nonlinear human–robot correlations. The robot was driven by kinesthetic teaching to reach its finger at the positions specified by the human. Certain reaching positions could be achieved by either moving the arm to the right or to the left of the pole on the table. Other positions, such as the ones indicated by the circle, could only be achieved by one type of demonstration

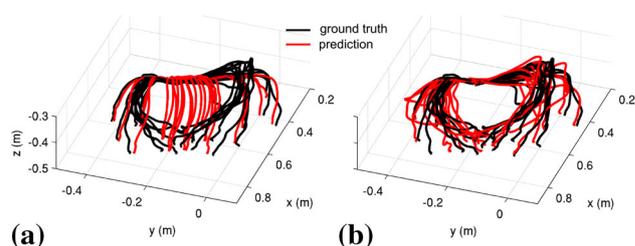


Fig. 15 Results of the predictions of the robot trajectories in Cartesian space. Both subplots show the same ground truth trajectories (in black). Predictions are generated by leave-one-out cross-validation on the whole data set comprised of 28 demonstrations. **a** Prediction using the conventional Interaction ProMPs with a single Gaussian. **b** Prediction using the proposed method with a mixture of Gaussians for the case of 8 components

we placed a pole in front of the robot as shown in Fig. 14 (also, the human uses a wand to specify the reaching positions to improve the accuracy of the pointing). Thus, the robot could only achieve certain positions, specified by the human, by moving either to the right or to the left of the pole. This scenario forced the robot to assume quite different configurations, depending on which side of the pole its arm was moving around.

As shown by the two pictures in Fig. 14, for certain positions, both right and left demonstrations could be provided for the same position specified by the human. For other positions, as the ones indicated by the circle, only one demonstration was possible. The demonstrations, totaling 28 pairs of human–robot trajectories, resulted in a multi-modal distribution of right and left trajectory patterns moving around the pole.

As shown in Fig. 15a, the modeling of the whole distribution of trajectory weights with a single Gaussian could not provide the richness required to capture the multi-modal training data. Several trajectories generated by a single component Interaction ProMP crossed over the middle of the

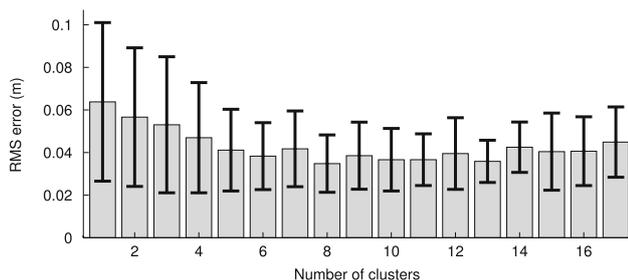


Fig. 16 RMS error when predicting test trajectories as a function of the number of Gaussian components

demonstrated trajectories, which represents the mean of the single Gaussian distribution. Figure 15b shows the predictions using a mixture of Interaction ProMPs. The GMM assumption that the parameters are only locally linear correlated seemed to represent the data much more accurately. As shown in Fig. 16, this improvement is quantified in terms of the RMS error of the prediction of the trajectory in relation to the ground truth using leave-one-out cross-validation over the whole data set. The same figure also shows that there is a sharp decrease in the RMS error up to six clusters, especially when taking into account the variance among the 28 tests. Beyond seven clusters no improvement is observed and the prediction error fluctuates around 4 cm.

A video of the experiments can be found in the following link:

<https://youtu.be/7w86NzGM7Rw>

and also available as Supplementary Materials of this article.

5 Interaction ProMPs: limitations, extensions and other uses

This section discusses the limitations and extensions of Interaction ProMPs in regards to the speed of the interaction, trajectory optimization for obstacle avoidance, and extensions of the framework to different applications.

5.1 Inference on partial trajectories and sensitivity to speed variation

In the experiments of Sect. 4 the conditioning of the interaction model was made at the end of the human trajectory due to two reasons of practical nature. First, this procedure suited our tasks as the robot aimed at approaching the final position of the human wrist. Second, recall from Sect. 3.1 that multiple demonstrations are assumed to be time-aligned. While time-alignment simplifies the modeling during the training phase, it induces the assumption that the human has no variation in

the speed of the movement, which is not realistic. However, by conditioning only at the end of the human trajectory, as it was done during the experiments in Sect. 4, variations in the speed become irrelevant.

In certain cases, it may be desired that the robot infers the trajectory while the human is moving, which brings the need to infer the handover location from partial observations of the human trajectory. Since the human is expected to have variability in speed, we test the robustness of our method by posing the following question. Given that an observation was made at instant t , what is the effect on the task recognition and spatial coordination accuracy when erroneously conditioning the model at $t + \epsilon$, where ϵ accounts for variations of the speed of the human movement?

The procedure consisted in running leave-one-out cross validation (LOOCV) where Interaction ProMPs were conditioned at ten randomly drawn observations within the first 25% of the test trajectory, providing an intermittent and partially observed test data (for example, to simulate an unreliable streaming of positions or eventual marker occlusion). We then generated additional tests by increasing and decreasing the speed of the human by temporally scaling the temporal axis at the rates of 0.75, 1, 1.25, and 1.5.¹ Three cases are illustrated in Fig. 17a where the same Interaction ProMP of the plate handover is conditioned on the random samples temporally misaligned due to changes in speed.

Given these test data, the probability of each of the three components (plate handover, holding tool, screw handover) and the final Cartesian positioning error of the robot end-effector were quantified. The Cartesian error was computed with

$$e = \sqrt{(x_{gt} - x_{pr})^2 + (y_{gt} - y_{pr})^2 + (z_{gt} - z_{pr})^2}, \quad (23)$$

where $(\cdot)_{gt}$ is the ground truth from the test data, and $(\cdot)_{pr}$ is the predicted value from the conditioned model. The sensitivity on positioning is shown in Fig. 17b as the average over the whole data set of plate handovers from the LOOCV runs. The error bar represents \pm one standard deviation. The bar indicated as “EndTraj” is the method used during experiments where conditioning was made only at the end of the human observed position; leading to a mean error of 4.37 cm. “EndTraj” is different from the bar labeled with “1”, where the trajectory of the robot was predicted from the first 25% of the human observation, thus leading to an increased error of 7.39 cm.

As expected, the prediction was worse when temporal misalignments of the observations due to speed variations occurred. The method seems to be more sensitive to misalignments of faster movements—trajectory durations sped

¹ For example, a rate of 1.25 acts as a surrogate for a human that moves 25% slower than the time-aligned interaction model.

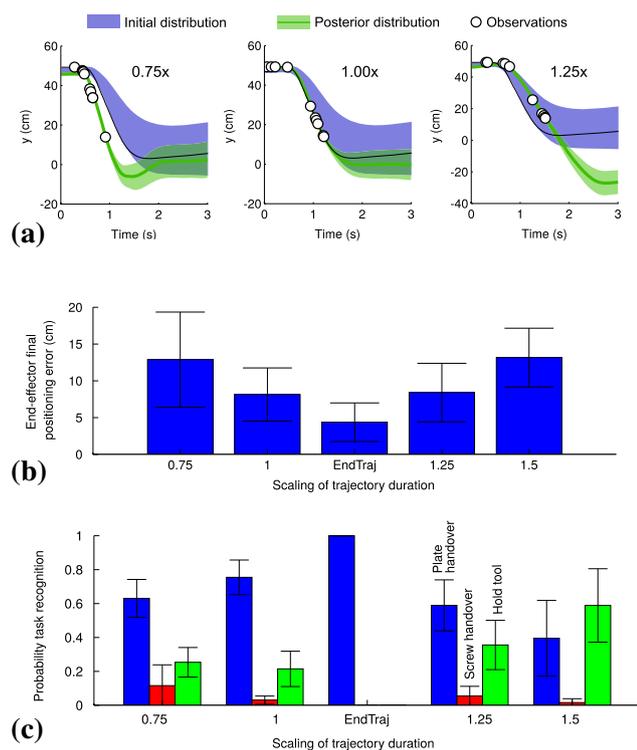


Fig. 17 The robustness of the method in relation to partial sparse data measured at different speeds. **a** Three cases of randomly sampled observations perturbed by scaling the original duration by factors of 0.75, 1, and 1.25. Only the first 25 % of the trajectory is observed. **b** The average error of the final robot end-effector position as a function of changes in the speed. The bar labeled “EndTraj” represents the case where the interaction primitive is conditioned at the final human position. **c** Partial observations and unpredicted changes in speed also affect the probability of task recognition

up by a factor of 0.75 have roughly the same error as the trajectories slowed down by a factor of 1.5. The effect on task recognition was, however, the opposite; as shown in Fig. 17c, observations slowed by a factor of 1.5 were most of the time recognized as the holding tool task.

In conclusion, speed variations of $\pm 25\%$ increased the positioning error up to 13 cm (for the case of a trajectory duration scaled by 0.75) while the probability of correct task recognition was of 60 % (we disregard the 1.5 scaling as tasks were not properly recognized). Ultimately, this sensitivity is a function of the differences among the trajectories of the multiple tasks and on the variability of the speeds in the human movement. A human partner with a consistent timing will perform better coordination with the robot than a human that has large temporal variability in his/her movement.

Principled approaches to address arbitrary variations in human speed and partial observations must consider the estimation of speed explicitly. Speed estimation is, however, a challenging problem where a number of approaches have been proposed. They can vary from the explicit encoding of the time as part of the state-space (Calinon and Billard 2009;

Calinon et al. 2012), to controllers that are reactive to temporal variations such as in (Kim et al. 2010) and (Englert and Toussaint 2014). These methods, however, operate directly on trajectories of a single robot agent. A method for phase, rather than speed, estimation with movement primitives for human–robot interaction was proposed in (Ben Amor et al. 2014) and relies on using DTW iteratively to find the lowest time alignment cost. Each cost is computed from an exhaustive search on different sizes of segments of the trajectory. Such form of brute force search is not suitable considering the online requirement of the problem.

We have been working on extensions of ProMPs and Interaction ProMPs for inference of the human phase, from which the speed can be recovered. We have shown that the reaction time of the robot could be decreased by 25 % with phase inference, which allows the conditioning of the model with partially observed trajectories of the human (Maeda et al. 2015). Preliminary work on multiple nonlinear velocity profiles within the same trajectory has been recently addressed (Ewerton et al. 2015).

5.2 Interaction ProMPs and trajectory optimization

A distinct feature of Interaction ProMPs in relation to path planning and trajectory optimization is to generate trajectories with an inference procedure where the trajectory profiles are a function of informative (demonstrated) priors. However, trajectory generation based on inference has the drawback of being limited to the demonstrated set, not generalizing beyond it. This section exploits Interaction ProMPs with trajectory optimization to generalize demonstrations—for example, for obstacle avoidance—illustrated with a 2D problem.

Consider the demonstrations shown in Fig. 18a where a letter “a” was drawn with a mouse multiple times on a computer screen. The patches represent \pm two standard deviations around the mean. The X direction of the writing is used to represent the movements of a single-DoF observed agent, and the Y direction to represent the inferred trajectory of the robot. Figure 18b shows the case where the Interaction ProMP was conditioned on the final position of the observed agent. The predicted robot trajectory collides with an obstacle which was not present during the demonstrations. As the obstacle is within the prior distribution, Interaction ProMPs can quickly return a collision-free trajectory simply by conditioning the robot on a via-point defined on a safe distance in relation to the obstacle. The resulting distribution is shown in Fig. 18c.

Figure 18d shows the case where the obstacle crosses the distribution of demonstrated trajectories. In this case, obstacle avoidance was solved with trajectory optimization, here using PI^2 (Theodorou et al. 2010), on the robot’s ProMP with a cost that penalizes collisions and deviations from the

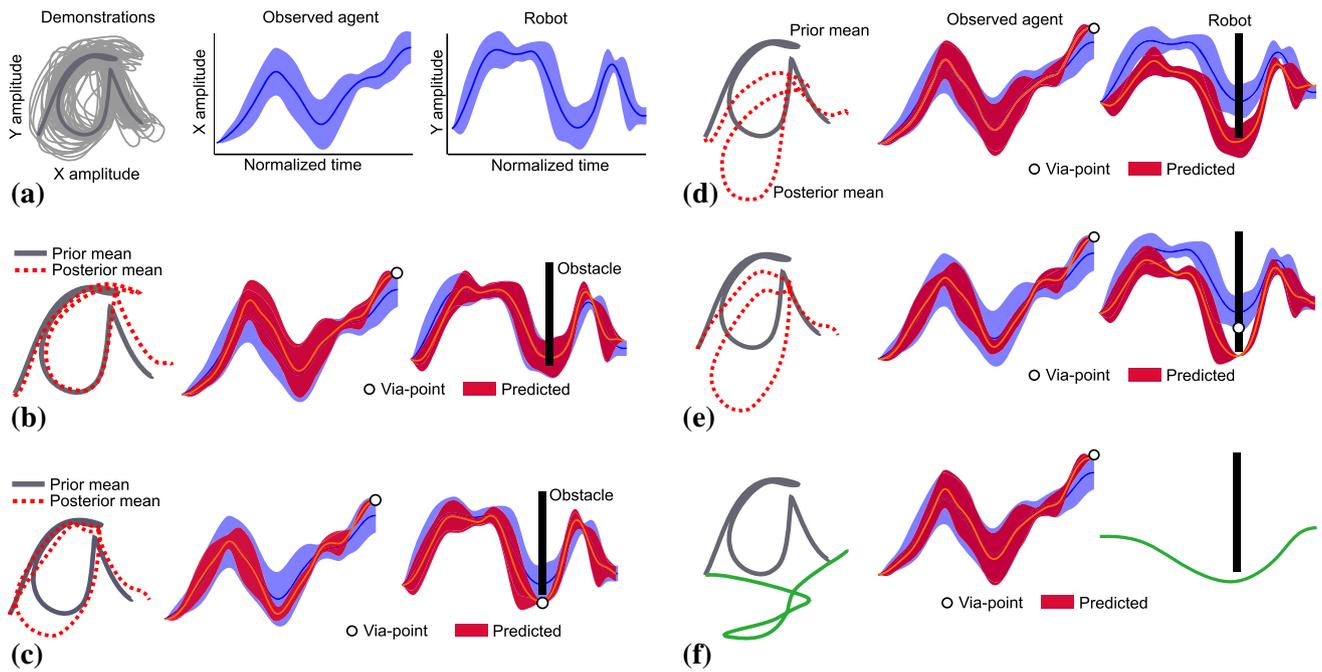


Fig. 18 An illustrative 2D toy-problem to investigate trajectory optimization with Interaction ProMPs. The axes were removed for clarity. **a** Demonstration data set obtained from multiple drawings of a letter “a”. **b** Interaction ProMP conditioned only on the observed agent.

prior mean. The initial guess was given by the mean of the prior distribution. We refer to this method simply as “Trajectory Optimization”. One possible application of Interaction ProMPs for obstacle avoidance is to use the posterior as an initial guess for the optimizer, conditioned within the distribution limits but as far as possible from the obstacle. Not only conditioning is computationally inexpensive when compared to numerical optimization but it returns a posterior that is feasible and resembles the demonstrations. The result of such combination is shown in Fig. 18e. Although both letters in (d) and (e) have similar profiles, as shown in Fig. 19, the benefit of the combination is evident when comparing the obstacle avoidance cost as a function of the number of iterations for the cases with and without Interaction ProMPs. An informative initialization provided by Interaction ProMPs reduced the initial optimization cost by half, and required only six iterations rather than 13 to converge to the same cost.

While trajectory optimization and recent methods for motion planning based on trajectory optimization (Ratliff et al. 2009; Kalakrishnan et al. 2011) have been widely accepted in robotics, their focus is to optimally achieve a goal while satisfying task constraints such as obstacle avoidance and joint/torque limits. Different from imitation learning based methods, trajectory optimization does not exploit a prior of human demonstrations and does not address action recognition. Figure 18f illustrates a pure trajectory optimization solution that generates a collision free trajectory for the

c Interaction ProMP conditioned on a via-point to avoid obstacles. **d** Pure trajectory optimization initialized by the prior mean. **e** Trajectory optimization initialized by the conditioned interaction primitive. **f** Pure trajectory optimization without informative priors

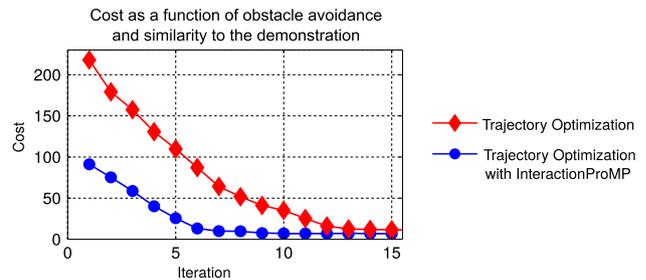


Fig. 19 Using Interaction ProMPs in conjunction with trajectory optimization. In comparison to conventional trajectory optimization, the combined method starts with a solution that has half of the cost, and converges much faster to the local optimum

robot without an informative prior. The robot goal position was inferred from the observed agent and the optimizer was given a straight line as initial guess. Clearly, no resemblance with the letter “a” should be expected.

In the real toolbox assembly the equivalent problem can be expected when using simple trajectory optimizers for handovers. While a collision-free trajectory may be found, it is not clear how to use trajectory optimization to exploit demonstrated movements other than using hand-coded costs. Demonstrations are particularly important for human–robot interaction as they encode predictable or legible movements for interaction which are not easy to quantify, and therefore to optimize. For the same reason it is not possible to com-

pletely replace an interaction primitive by a controller that directly tracks the markers of the human: feedback control does not provide the flexibility and complexity of the trajectories that can be encoded in a primitive learned from human demonstrations. Finally, Interaction ProMPs allow for the encoding and recognition of human gestures, such as retrieving of pointed objects (in the spirit of Fig. 6), which are not easily accomplished with trajectory optimization methods.

These initial evaluations shows that a combination of trajectory optimization and Interaction ProMPs is synergistic. Our method provides action recognition capabilities and informative initial guesses for optimizations. On the other hand, trajectory optimization allows for the extrapolation of the conditioned model beyond the demonstrated set, and also to adapt trajectories to satisfy constraints such as obstacle avoidance and joint limits.

5.3 Interaction ProMPs for coordination of position and motor commands

While this paper motivates Interaction ProMPs for human–robot collaboration and assistance, this Section illustrates a different application of the algorithm. The generality of the method is an important difference between the Interaction ProMP and methods specific for handovers—see [Strabala et al. 2013](#) for a review on the topic. Note that the methodology presented in Sect. 3 makes no assumptions on the nature of the trajectories of the correlated agents; ultimately, the goal of an Interaction ProMP is to infer robot motions given external observations, which does not necessarily have to be provided by a human.

Here we present the use of Interaction ProMPs to encode the correlation between torque commands with joint positions of a single agent. This problem is illustrated with a simulated planar robot with three revolute joints shown in Fig. 20a. The robot moves on the plane of gravity with linear joint PD controllers set to a low value. The robot movement is compliant but tracking accuracy is low. This is a typical case where a model-based inverse dynamics or a computed torque controller is required to provide compensation commands. We assume such a model is not available.

As shown in Fig. 20a we assume that the ideal torques to perfectly track the straight trajectories that reach the positions labeled as “rA”, “rB”, “rC”, and “rD” are given. Given that a new position “rE” must be reached, the inverse dynamics problem is that of generating the corresponding torque commands. We use a single task Interaction ProMPs in the same manner as described in Sect. 3.2 where the state vector (6) is

$$\mathbf{y}_t = [\mathbf{q}_{1,t}, \mathbf{q}_{2,t}, \mathbf{q}_{3,t}, \mathbf{u}_{1,t}, \mathbf{u}_{2,t}, \mathbf{u}_{3,t}]^T, \quad (24)$$

such that correlations between joint and torque commands are captured by the model. In this case, the Interaction ProMP

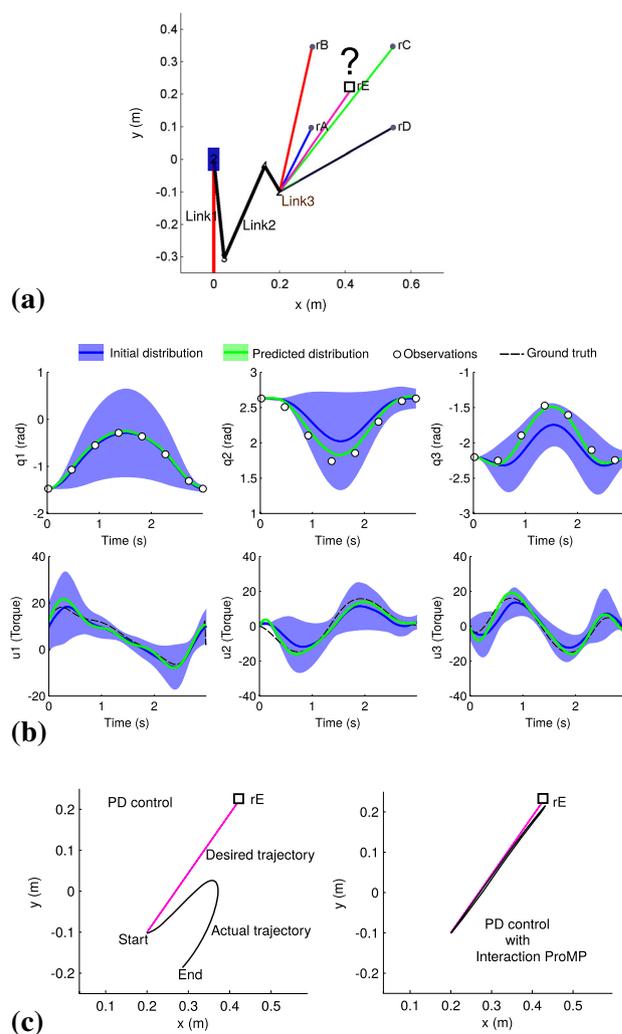


Fig. 20 **a** The reaching task to evaluate Interaction ProMPs as a local inverse dynamics model. **b** Distribution over joint trajectories and torque commands conditioned on eight desired joint states. **c** The Cartesian trajectory of the end-effector. The average tracking error using the Interaction ProMP as a feedforward input decreased the Cartesian tracking error from 0.153 to 0.008 m

is effectively a local inverse dynamics of the robot for the reaching task.

As shown in Fig. 20b, we use eight states along the trajectory to reach the position “rE” as observations. The conditioned Interaction ProMP model returns the torque commands which are used in conjunction with the joint PD controllers. Since observations are ideal and noiseless the conditioned model has approximately zero uncertainty. Figure 20c shows the tracking result of the robot when reaching the position “rE” with and without the Interaction ProMP prediction. Due to gravity, the PD controller can barely track the trajectory, rendering a root-mean-square error in Cartesian space of 0.153 m. The controller with Interaction ProMPs can achieve a much better tracking with an error of 0.008 meters despite the fact that this position was not part of the training data. This

application of Interaction ProMP can greatly speed-up policy search methods by replacing uninformed initial guesses with a prior model of the previously seen solutions.

6 Conclusions

This paper presented an algorithm for collaborative and assistive robots whose movements must be coordinated with the movements of a human partner given multiple and unlabeled tasks. This goal was achieved with a mixture of Interaction Probabilistic Movement Primitives to learn a model of nonlinearly correlated human–robot parameterized trajectories. To address multiple tasks, a mixture of Interaction ProMPs as a GMM was proposed. A procedure to compute the most probable component was presented, rendering a method that both recognizes the task and also coordinates the movement of the robot according to the observed human action.

We provided a variety of experiments to evaluate the accuracy of the inference, and also to validate its application for controlling a robot assistant. We compared the main differences between DMPs and ProMPs for interaction and advocated the latter for applications where measurements are noisy and/or prone to interruption. We also showed that Interaction ProMPs make better use of the available demonstrations when compared to lookup table approaches.

In the last part of this article we discussed limitations, alternative uses and extensions of the Interaction ProMP framework. The problem of the sensitivity of the method in regards to speed variability, the combined use of the framework with trajectory optimization for obstacle avoidance, and the generality of the method for applications that go beyond human–robot interaction.

Acknowledgments The research leading to these results has received funding from the European Community’s Seventh Framework Programmes (FP7-ICT-2013-10) under Grant agreement #610878 (3rdHand) and from the European Union’s Horizon 2020 research and innovation programme under grant agreement #645582 (RoMaNS) and from the Project BIMROB of the Forum für interdisziplinäre Forschung (FiF) of the TU Darmstadt. The authors would like to acknowledge Heni Ben Amor for the invaluable ideas and discussions that contributed to this paper.

Appendix: Time-alignment of multiple demonstrations

One issue of imitation learning is that multiple demonstration trajectories provided by humans are usually, sometimes severely, warped in time. To compute the distribution of ProMP weights the demonstrated trajectories must be first aligned in relation to a common “clock”. In the context of

movement primitives, this clock is often referred to as the phase variable. In this paper, all human and robot trajectories collected during the experiments presented in Sect. 4 were aligned by using the method briefly presented in (Maeda et al. 2014) and will be described in detail here.

The method consists of minimizing the cost J defined as the cumulative absolute difference between the demonstrated trajectory to be time-aligned \mathbf{y}_w and a trajectory taken as a phase reference \mathbf{y}_r ,

$$J = \sum_{k=0}^K |\mathbf{y}_r(k) - \mathbf{y}_w(\mathbf{t}_w^{j+1}(k))|, \quad (25)$$

where both trajectories are resampled to have the same number of K steps. The vector \mathbf{t}_w^{j+1} is the unwarped time, which is the solution of the iterative update

$$\mathbf{t}_w^{j+1} = v_0^j + \mathbf{G}^j \mathbf{t}_w^j, \quad (26)$$

where $\mathbf{G} = \text{diag}(g(1), \dots, g(K))$ and j is the iteration number of the optimization step.

We propose $g(k)$ as a smooth and continuous warping function parameterized by N weights

$$g(k) = \psi_k^T \mathbf{v}_{1:N}, \quad (27)$$

where ψ_k is the Gaussian basis function at time step k . The vector of parameters $\mathbf{v}^j = [v_0, v_1, \dots, v_N]$ is optimized by gradient descent to decrease the cost J defined in (25). The extra parameter v_0 is used to shift the time which is useful when the reference and warped trajectories are, in fact, identical but start at different instants. The optimization is usually initialized with $v_0^j = 0$ and $\mathbf{t}_w^j = \mathbf{t}_r$.

Dynamic Time Warping (DTW) (Sakoe and Chiba 1978) is a method widely used for solving time-alignment problems. An extension of DTW for the case where the time-alignment must be made on-line given only partial observations of \mathbf{y}_w was presented in (Ben Amor et al. 2014). An issue intrinsic to DTW-based algorithms, however, is that several adjacent time steps of the trajectory to be aligned may be attributed to a single time step of the reference trajectory, and vice-versa. For trajectories provided by a dynamical system, this issue leads to discontinuities in the solution and unnatural movements. An extreme example of this problem is shown in Fig. 21(a) where it is observed that parts of the warped trajectory were lost after the DTW alignment.

A heuristic referred to as the slope constraint was proposed in (Sakoe and Chiba 1978) to alleviate this problem by forcing the same index to not be repeated more than a certain threshold. The slope constraint, however, does not completely solve the discontinuity problem and the tuning of the slope constraint is task dependent. By construction, our proposed method enforces that the warping function \mathbf{g} is

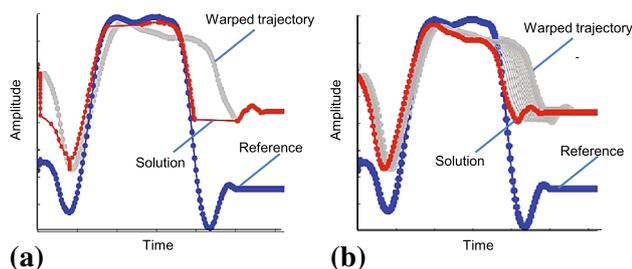


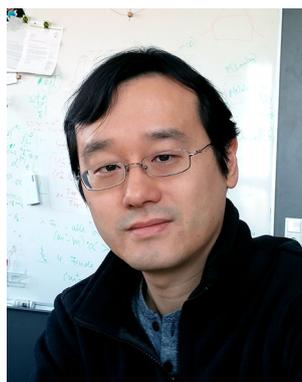
Fig. 21 Comparison of time-alignment given by classical Dynamic Time Warping without slope constraint (a) and the proposed method using gradient descent (b). In the latter, the thin gray curves represent intermediate solutions of the local optimization while converging

both continuous and smooth. The use of a smooth function not only avoids the tuning of slope constraint but also preserves the overall shape of the trajectory. Figure 21(b) shows the solution of our method for the same input data used in Fig. 21(a).

References

- Ben Amor, H., Neumann, G., Kamthe, S., Kroemer, O., & Peters, J. (2014). Interaction primitives for human–robot cooperation tasks. In: *Proceedings of the IEEE international conference on robotics and automation (ICRA)*.
- Ben Amor, H., Vogt, D., Ewerton, M., Berger, E., Jung, B., & Peters, J. (2013). Learning responsive robot behavior by imitation. In: *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 3257–3264).
- Bishop, C. (2006). *Pattern recognition and machine learning* (Vol. 4(4)). New York: Springer.
- Bonilla, B. L., & Asada, H. H. (2014). A robot on the shoulder: Coordinated human-wearable robot control using coloured petri nets and partial least squares predictions. In: *Proceedings of the IEEE international conference on robotics and automation (ICRA)* (pp. 119–125).
- Cakmak, M., Srinivasa, S. S., Lee, M. K., Forlizzi, J., & Kiesler, S. (2011). Human preferences for robothuman hand-over configurations. In: *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 1986–1993).
- Calinon, S., & Billard, A. (2009). Statistical learning by imitation of competing constraints in joint space and task space. *Advanced Robotics*, 23(15), 2059–2076.
- Calinon, S., D’halluin, F., Sauser, E. L., Caldwell, D. G., & Billard, A. G. (2010). Learning and reproduction of gestures by imitation. *IEEE Robotics & Automation Magazine*, 17(2), 44–54.
- Calinon, S., Li, Z., Alizadeh, T., Tsagarakis, N. G., & Caldwell, D. G. (2012). Statistical dynamical systems for skills acquisition in humanoids. In: *Proceedings of the international conference on humanoid robots (HUMANOIDS)* (pp. 323–329).
- Englert, P., & Toussaint, M. (2014). Reactive phase and task space adaptation for robust motion execution. In: *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 109–116).
- Ewerton, M., Maeda, G., Peters, J., & Neumann, G. (2015). Learning motor skills from partially observed movements executed at different speeds. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 456–463).
- Ewerton, M., Neumann, G., Lioutikov, R., Ben Amor, H., Peters, J., & Maeda, G. (2015). Learning multiple collaborative tasks with a mixture of interaction primitives. In: *Accepted: Proceedings of the IEEE international conference on robotics and automation (ICRA)*.
- Ijspeert, A. J., Nakanishi, J., Hoffmann, H., Pastor, P., & Schaal, S. (2013). Dynamical movement primitives: Learning attractor models for motor behaviors. *Neural Computation*, 25(2), 328–373.
- Kalakrishnan, M., Chitta, S., Theodorou, E., Pastor, P., & Schaal, S. (2011). STOMP: Stochastic trajectory optimization for motion planning. In: *Proceedings of the IEEE international conference on robotics and automation (ICRA)* (pp. 4569–4574).
- Kim, S., Gribovskaya, E., & Billard, A. (2010). Learning motion dynamics to catch a moving object. In: *Proceedings of the IEEE-RAS international conference on humanoid robots (HUMANOIDS)* (pp. 106–111).
- Koppula, H. S., & Saxena, A. (2013). Anticipating human activities using object affordances for reactive robotic response. In: *Robotics: Science and systems*.
- Kulvicius, T., Biehl, M., Aein, M. J., Tamosiunaite, M., & Wörgötter, F. (2013). Interaction learning for dynamic movement primitives used in cooperative robotic tasks. *Robotics and Autonomous Systems*, 61(12), 1450–1459.
- Kupcsik, A., Hsu, D., & Lee, S. (2015). Learning dynamic robot-to-human object handover from human feedback. In: *International symposium on robotics research (ISRR)*.
- Lawitzky, M., Medina, J., Lee, D., & Hirche, S. (2012, Oct). Feedback motion planning and learning from demonstration in physical robotic assistance: differences and synergies. In: *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 3646–3652).
- Lee, D., Ott, C., & Nakamura, Y. (2010). Mimetic communication model with compliant physical contact in human–humanoid interaction. *The International Journal of Robotics Research*, 29(13), 1684–1704.
- Maeda, G., Ewerton, M., Lioutikov, R., Ben Amor, H., Peters, J., & Neumann, G. (2014). Learning interaction for collaborative tasks with probabilistic movement primitives. In: *Proceedings of the international conference on humanoid robots (HUMANOIDS)* (pp. 527–534).
- Maeda, G., Neumann, G., Ewerton, L. R., M., & Peters, J. (2015). A probabilistic framework for semiautonomous robots based on interaction primitives with phase estimation. In: *Proceedings of the international symposium of robotics research (ISRR)*.
- Mainprice, J., & Berenson, D. (2013). Human–robot collaborative manipulation planning using early prediction of human motion. In: *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 299–306).
- Oliver, N., Rosario, B., & Pentland, A. (2000). A Bayesian computer vision system for modeling human interactions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8), 831–843. doi:10.1109/34.868684.
- Paraschos, A., Daniel, C., Peters, J., & Neumann, G. (2013). Probabilistic movement primitives. In: *Advances in neural information processing systems (NIPS)* (pp. 2616–2624).
- Ratliff, N., Zucker, M., Bagnell, J. A., & Srinivasa, S. (2009). Chomp: Gradient optimization techniques for efficient motion planning. In: *Proceedings of the IEEE international conference on robotics and automation (ICRA)* (pp. 489–494).
- Rohmer, E., Freese, M., Singh, S. P. (2013). V-rep: A versatile and scalable robot simulation framework. In: *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems (IROS)*.

- Rozo, L., Calinon, S., Caldwell, D. G., Jimenez, P., & Torras, C. (2013). Learning collaborative impedance-based robot behaviors. In: *AAAI conference on artificial intelligence*. Bellevue, Washington, USA
- Sakoe, H., & Chiba, S. (1978). Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 26(1), 43–49.
- Schaal, S. (1999). Is imitation learning the route to humanoid robots? *Trends in Cognitive Sciences*, 3(6), 233–242.
- Sisbot, E. A., & Alami, R. (2012). A human-aware manipulation planner. *IEEE Transactions on Robotics*, 28(5), 1045–1057.
- Strabala, K. W., Lee, M. K., Dragan, A. D., Forlizzi, J. L., Srinivasa, S., Cakmak, M., et al. (2013). Towards seamless human–robot handovers. *Journal of Human–Robot Interaction*, 2(1), 112–132.
- Tanaka, Y., Kinugawa, J., Sugahara, Y., & Kosuge, K. (2012). Motion planning with worker’s trajectory prediction for assembly task partner robot. In: *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 1525–1532).
- Theodorou, E., Buchli, J., & Schaal, S. (2010). Reinforcement learning of motor skills in high dimensions: A path integral approach. In: *Proceedings of the IEEE international conference on robotics and automation (ICRA)* (pp. 2397–2403).
- Wang, Z., Muelling, K., Deisenroth, M. P., Ben Amor, H., Vogt, D., Schoelkopf, B., et al. (2013). Probabilistic movement modeling for intention inference in human–robot interaction. *The International Journal of Robotics Research*, 7, 841–858.
- Yamane, K., Revfi, M., & Asfour, T. (2013). Synthesizing object receiving motions of humanoid robots with human motion database. In: *Proceedings of the IEEE international conference on robotics and automation (ICRA)* (pp. 1629–1636).
- Yamane, K., Yamaguchi, Y., & Nakamura, Y. (2011). Human motion database with a binary tree and node transition graphs. *Autonomous Robots*, 30(1), 87–98.



Guilherme J. Maeda received his B.Sc. in Mechanical Engineering from the University of Sao Paulo, Brazil (2003) and the M.Sc. in Control Engineering from the Tokyo Institute of Technology, Japan (2007). He received his Ph.D. from the Australian Centre for Field Robotics, University of Sydney, Australia in 2013. In the same year he joined the Intelligent Autonomous Systems group (IAS) in TU Darmstadt as a Research Scientist and Team

Leader. His research interests are in robotics, specifically in control, learning for control, and their applications to human–robot interaction. He is currently leading the participation of the IAS team in the EU-funded FP7 programme 3rdHandRobot Project.



Gerhard Neumann is an Assistant Professor at the TU Darmstadt since September 2014 and head of the Computational Learning for Autonomous Systems (CLAS) Group. Before becoming Assistant Professor, he joined the IAS group as Post-Doc in November 2011 and became a Group Leader for Machine Learning for Control in October 2013. Gerhard did his Ph.D. at the TU Graz under the supervision of Wolfgang Maass.



Marco Ewerton is a Ph.D. Student at the IAS under the supervision of Guilherme Maeda and Jan Peters since January 2015. He works on the BIMROB Project, that investigates how humans and robots can improve their movements by interacting with each other. He obtained his master’s degree in Electrical and Informatic Engineering from the TU Darmstadt in 2014. His master’s thesis work focused on modeling human–robot interaction with probabilistic movement representations.

From April 2012 to December 2013, he was a research assistant at the IAS under the supervision of Heni Ben Amor. During that time, the main topics of Marco’s work were “3D reconstruction from multiple Kinect cameras” and “Human–Robot Interaction”.



Rudolf Lioutikov joined the Intelligent Autonomous System Lab in November 2013 as a Ph.D. student. His research will amongst others include robust and safe policy search methods, skill learning, human–robot interaction and intention learning. During his Ph.D., Rudolf is working on the 3rd Hand Project where he will develop and evaluate new methods in the field of semi-autonomous human–robot interaction tasks.



Oliver Kroemer is a postdoctoral researcher at the University of Southern California (USC). He received his Masters and Bachelors degrees in engineering from the University of Cambridge in 2008. He was a Ph.D. student at the Max Planck Institute for Biological Cybernetics from 2009 to 2011. In 2014, Oliver defended his Ph.D. thesis at the Technische Universitaet Darmstadt. He was a finalist for the 2015 Georges Giralt Ph.D. Award for the best robotics Ph.D.

thesis in Europe. Oliver's research interests are in machine learning and robotics, with a focus on learning grasping and manipulation skills.



Jan Peters is a Full Professor (W3) for Intelligent Autonomous Systems at the Computer Science Department of the Technische Universitaet Darmstadt and at the same time a senior research scientist and group leader at the Max-Planck Institute for Intelligent Systems, where he heads the interdepartmental Robot Learning Group. Jan Peters has received the Dick Volz Best 2007 US Ph.D. Thesis Runner-Up Award, the 2012 Robotics: Science & Systems -

Early Career Spotlight, the 2013 INNS Young Investigator Award, and the IEEE Robotics & Automation Society's 2013 Early Career Award. In 2015, he was awarded an ERC Starting Grant. Jan Peters has studied Computer Science, Electrical, Mechanical and Control Engineering at TU Munich and FernUni Hagen in Germany, at the National University of Singapore (NUS) and the University of Southern California (USC), and been an visiting researcher at the ATR Telecommunications Research Center in Japan. He has received four Master's degrees in these disciplines as well as a Computer Science PhD from USC.