Interaction Primitives for Human-Robot Cooperation Tasks

Heni Ben Amor[1], Gerhard Neumann[2], Sanket Kamthe[2], Oliver Kroemer[2], Jan Peters[2,3]

***Abstract*— To engage in cooperative activities with human partners, robots have to possess basic interactive abilities and skills. However, programming such interactive skills is a challenging task, as each interaction partner can have different timing or an alternative way of executing movements. In this paper, we propose to learn interaction skills by observing how two humans engage in a similar task. To this end, we introduce a new representation called *Interaction Primitives*. Interaction primitives build on the framework of dynamic motor primitives (DMPs) by maintaining a distribution over the parameters of the DMP. With this distribution, we can learn the inherent correlations of cooperative activities which allow us to infer the behavior of the partner and to participate in the cooperation. We will provide algorithms for synchronizing and adapting the behavior of humans and robots during joint physical activities.**

I. INTRODUCTION

Creating autonomous robots that assist humans in situations of daily life has always been among the most important visions in robotics research. Such human-friendly assistive robotics requires robots with dexterous manipulation abilities and safe compliant control as well as algorithms for human-robot interaction during skill acquisition. Today, however, most robots have limited interaction capabilities and are not prepared to appropriately respond to the movements and behaviors of their human partners. The main reason for this limitation is the fact that programming robots for such interaction scenarios is notoriously hard, as it is difficult to foresee many possible actions and responses of the human counterpart.

Over the last ten years, the field of imitation learning [12] has made tremendous progress. In imitation learning, a user does not specify the robot's movements using traditional programming languages. Instead, he only provides one or more demonstrations of the desired behavior. Based on these demonstrations, the robot autonomously generates a control program that allows it to generalize the skill to different situations. Imitation learning has been successfully used to learn a wide range of tasks in robotics [2], such as basic robot walking [4], [5], driving robot cars [10], object manipulation [9], and helicopter manoeuvring [1]. A particularly successful approach to imitation learning is based on Dynamic Motor Primitives (DMPs)[6]. DMPs use dynamical systems as a way of representing control policies, which can be generalized to new situations. Several motor primitives can be chained together to realize more complex movement sequences.

Fig. 1. A human-robot interaction scenario as investigated in this paper. A robot needs to learn when and how to interact with a human partner. Programming such a behavior manually is a time-consuming and error-prone process, as it hard to foresee how the interaction partner will behave.

In this paper, we generalize the concept of imitation learning to human-robot interaction scenarios. In particular, we learn interactive motor skills, which allow anthropomorphic robots to engage in joint physical activities with a human partner. To this end, the movements of two humans are recorded using motion capture and subsequently used to learn a compact model of the observed interaction. In the remainder of this paper, we will call such a model an Interaction Primitive (IP). A learned IP is used by a robot to engage in a similar interaction with a human partner. The main contribution of this paper is to provide the theoretical foundations of interaction primitives and their algorithmic realization. We will discuss the general setup and introduce three core components, namely methods for (1) phase estimation, (2) learning of predictive DMP distributions, and (3) correlation the movements of two agents. Using examples from handwriting synthesis and human-robot interaction tasks, we will clarify how these components relate to each other. Finally, we will apply our approach to real-world interaction scenarios using motion capture systems.

II. RELATED WORK

Finding simple and natural ways of specifying robot control programs is a focal point in robotics. Imitation learning, also known as programming by demonstration, has been proposed as a possible solution to this problem [12]. Most ap-

[1]Institute for Robotics and Intelligent Machines, Georgia Institute of Technology, 801 Atlantic Drive, Atlanta, GA 30332-0280, USA

[2] Intelligent Autonomous Systems, Department of Computer Science, Technical University Darmstadt, Hochschulstr. 10, 64289 Darmstadt, Germany

[3] Max Planck Institute for Intelligent Systems Spemannstr. 38, 72076 Tuebingen, Germany

proaches to imitation learning obtain a control policy which encodes the behavior demonstrated by the user. The policy can subsequently be used to generate a similar behavior that is adapted to the current situation. Another way of encoding policies is to use statistical modeling methods. For example, in the mimesis model [8] a continuous hidden Markov model is used for encoding the teacher's demonstrations. A similar approach to motion generation is presented by Calinon et al. [3] who used Gaussian mixture regression to learn gestures.

The methods discussed so far are limited to single agent imitation learning scenarios. Recently, various attempts have been undertaken for using machine learning in human-robot-interaction scenarios. In [13], an extension of the Gaussian process dynamics model was used to infer the intention of a human player during a table-tennis game. Through the analysis of the human player's movement, a robot player was able to determine the position to which the ball will be returned. This predictive ability allowed the robot to initiate its movements even before the human hit the ball. In [7], Gaussian mixture models were used to adapt the timing of a humanoid robot to a human partner in close-contact interaction scenarios. The parameters of the interaction model were updated using binary evaluation information obtained from the human. While the approach allowed for human-in-the-loop learning and adaptation, it did not include any imitation of observed interactions. In a similar vein, the work in [8] showed how a robot can be actively involved in learning how to interact with a human partner. The robot performed a previously learned motion pattern and observed the partner's reaction to it. Learning was realized by recognizing the observed reaction and by encoding the action-reaction patterns in a HMM. The HMM was then used to synthesize similar interactions.

In our approach, learning of motion and interaction are not split into two separate parts. Instead, we learn one integrated interaction primitive which can directly synthesize an appropriate movement in response to an observed movement of the human partner. Furthermore, instead of modelling symbolic action-reaction pairs, our approach models the joint movement of a continuous level. This continuous control is realized through the use of DMPs as the underlying representation. By introducing probabilistic distributions and bayesian inference in the context of DMPs, we obtain a new set of tools for predicting and reacting to human movements.

III. INTERACTION PRIMITIVES

The goal of learning an Interaction Primitive is to obtain a compact representation of a joint physical activity between two persons and use it in human robot interaction. An interaction primitive specifies how a person adapts his movements to the movement of the interaction partner, and vice versa. For example, in a handing-over task, the receiving person adapts his arm movements to the reaching motion of the person performing the handing-over. In this paper, we propose an imitation learning approach for learning such interaction primitives. First, one or several demonstrations of the interaction task are performed by two human demonstrators in a motion capture environment. Using the motion capture data, we extract an interaction primitive which specifies the reciprocal dependencies during the execution of the task. Finally, the learned model is used by a robot to engage in a similar interaction with a human partner. An IP should also be applicable to a wide variety of related interactions, for example, handing over an object at different locations. An example of an interaction of a humanoid robot with a human partner performing a high-five movement is given in Fig. 2.

At the core of our approach is a new representation for interaction tasks. An IP can formally be regarded as a special type of DMP which represents a joint activity of two interaction partners. After an IP is learned from the demonstration data, it is used to control a robot in a similar task. For the sake of notational clarity, we will refer to the first interaction partner, i.e., the human, as the *observed agent*, while the second interaction partner, i.e., the robot, will be called *controlled agent*.

The IP performs three steps to infer an appropriate way of reacting to the movement of the observed agent.

1) **Phase Estimation:** The actions of the interaction partners are executed in synchrony. In particular, the robot adapt its timing such that it matches the timing of the human partner. For this synchronization, we need to identify the current phase of the interaction
2) **Predictive DMP distributions:** As a next step, we compute predictions over the behavior of an agent given a partial trajectory $\boldsymbol{\tau}_o$ of the observed agent. To do so, we use a probabilistic approach and model a distribution $p(\boldsymbol{\theta})$ over the parameters of the DMPs. This distribution can be conditioned on a new, partial observation, i.e., to obtain an updated parameter distribution $p(\boldsymbol{\theta}|\boldsymbol{\tau}_o)$. We use samples of this distribution to predict the future behavior of the agent.
3) **Correlating both Agents:** In order to perform a successful cooperation, the movement of the robot needs to be correlated with the movement of the human. Such operation is a straightforward extension to the predictive DMP distributions. Instead of conditioning on the observation of all DoFs, we only condition on the DoFs of the observed agent, and, hence, we also obtain a distribution over the DMP parameters of the controlled agent, that can be used to control the robot.

In the following, we will address each of the steps above in detail. First, we will recapitulate the basic properties and components of DMPs. Subsequently, we will describe how phase estimation, adaptation, and correlation can be realized within the DMP framework in order to produce an interactive motor primitive.

A. Dynamic Motor Primitives

A DMP is an adaptive representation of a trajectory representing a human or robot movement [6]. In this section, we will give a brief recap of DMPs. The general idea is to encode a recorded trajectory as dynamical systems, which can be used to generate different variations of the original

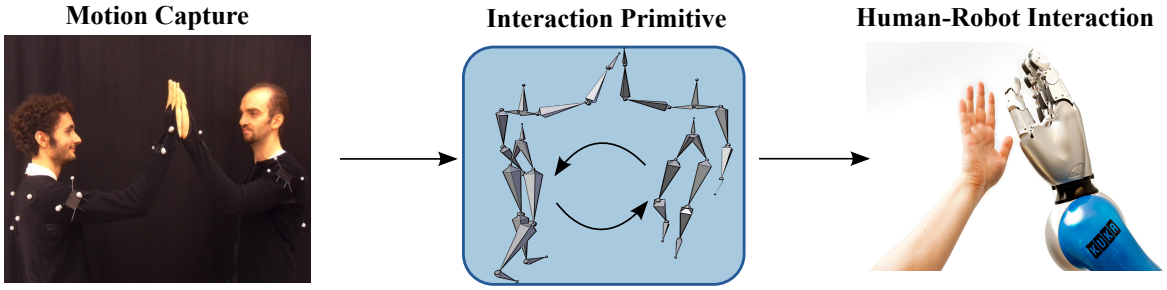

Fig. 2. An overview of the presented machine learning approach. Left: Using motion capture we first record the movements of two persons during an interaction task. Center: Given the recorded motion capture data, we learn an interaction primitive specifying each persons' movement as well as the dependencies between them. Right: During human-robot interaction, the learned interaction primitive is used by the robot to adapt its behavior to that of his human interaction partner.

movement. As a result, a robot can generalize a demonstrated movement to new situations that may arise. Formally, a DMP can be written as a dynamical system

$$\ddot{y} = (\alpha_y (\beta_y (g - y) - ((\dot{y})/\tau)) + f(x)) \tau^2 \quad (1)$$

where y is a state variable such as the joint angle to be controlled, g is the corresponding goal state, and τ is a time scaling factor. The first set of terms represents a critically-damped linear system with constant coefficients α_y and β_y. The last term is called the *forcing function*

$$f(x) = \frac{\sum_{i=1}^{m} \psi_i(x) w_i x}{\sum_{j=1}^{m} \psi_j(x)} = \boldsymbol{\phi}(x)^T \mathbf{w} \quad (2)$$

where $\psi_i(x)$ are Gaussian basis functions and $\mathbf{w}$ the corresponding weight vectors. The basis functions only depend on the phase variable x, which is the state of a canonical system shared by all degrees of feedom (DoFs). The canonical system acts as a timer to synchronize the different movement components. It has the form $\dot{x} = -\alpha_x x \tau$, where $x_0 = 1$ at the beginning of the motion and, thereafter, it decays towards zero. The elements of the weight vector $\boldsymbol{w}$ are denoted as shape-parameters, as they determine the acceleration profile of the movement, and, hence, indirectly also the shape of the movement. Typically, we learn a separate set of shape parameters $\boldsymbol{w}$ as well as the goal attractor g for each DoF. The goal attractor g can be used to change the target position of the movement while the time scaling parameter τ can be used to change the execution speed of the movement.

The weight parameters $\boldsymbol{w}$ of the DMP can be straightforwardly obtained from observed trajectories $\{y_{1:T}, \dot{y}_{1:T}, \ddot{y}_{1:T}\}$ by first computing the forcing function that would reproduce the given trajectory, i.e.,

$$\boldsymbol{F}_i = \frac{1}{\tau^2} \ddot{y}_i - \alpha_y (\beta_y (g - y_i) - \dot{y}_i / \tau). \quad (3)$$

Subsequently, we can solve the system $\boldsymbol{\Phi}\mathbf{w} = \boldsymbol{F}$ in a least squares sense, i.e.,

$$\boldsymbol{w} = (\boldsymbol{\Phi}^T \boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}^T \boldsymbol{F}, \quad (4)$$

where $\boldsymbol{\Phi}$ is a matrix containing of basis vectors for all time steps, i.e., $\boldsymbol{\Phi}_t = \boldsymbol{\phi}_t^T = \boldsymbol{\phi}(x_t)$.

B. Phase Estimation by Dynamic Time Warping

For a joint activity to succeed, the movement of the interaction partners needs to be temporally aligned. During the execution of human-robot interaction, the robot observes a partial movement of the human counterpart. Given this partial movement sequence, we need to determine the current state of the interaction. This is achieved by determining the current value of the phase variable x. To this end, we will use the dynamic time warping (DTW) algorithm [11]. DTW is a method for the alignment of time series data. Given two time series $\mathbf{u} = (u_1, \cdots, u_N)$ and $\mathbf{v} = (v_1, \cdots, v_M)$ of size N and M, DTW finds optimal correspondences between data points, such that a given distance function D is minimized. This task can be formulated as finding an alignment between a reference trajectory $\mathbf{u}$ and an observed subsequence $\mathbf{v}$. In our specific case, the reference trajectory is the movement of the observed agent during the original demonstration of the task, and the query trajectory is the currently seen partial movement sequence of the human interaction partner. We define an alignment π as a set of tuples (π_1, π_2) specifying a correspondence between point π_1 of the first time series and point π_2 of the second time series. To find such an alignment, we first calculate the accumulated cost matrix, which is defined as

$$D(1,m) = \Sigma_{k=1}^{m} c(u_1, v_k), m \in [1:M] \quad (5)$$

$$D(n,1) = \Sigma_{k=1}^{n} c(u_k, v_1), n \in [1:N] \quad (6)$$

$$D(n,m) = \min\{D(n-1,m-1), D(n,m-1), \ldots D(n-1,m)\} + c(u_n, v_m) \quad (7)$$

where c is a local distance measure, which is often set to the squared Euclidean distance, i.e., $c = ||u - v||^2$. In the original DTW formulation, finding the optimal alignment is cast as the problem of finding a path from $(1,1)$ to (N,M) producing minimal costs according to the accumulated cost matrix. This optimization is achieved using a dynamic programming recursion. The DTW approach above assumes that both time series have approximately the same length. However, in our case we want to match a partial movement to the reference movement. To this end, we modify the DTW algorithm and determine the path with minimal distance starting at $(1,1)$

Fig. 3. Phase estimation and pattern completion using DTW and DMPs. Given the partial observation (black), we estimate the current phase, and use it to generate the unseen part (red) of the letter. The goal does not have to be specified and is estimated alongside the other parameters.

and ending at (n^*, M), where n^* is given by

$$n^* = \underset{n}{\operatorname{argmin}}\ D(n, M). \tag{8}$$

The index n^* reflects the frame in the reference movement which produces minimal costs with respect to the observed query movement. As a result it can be used to estimate the current value of the phase variable x of the canonical system. More specifically, calculating (n^*/N) yields an estimated of the relative time that has passed, assuming a constant sampling frequency. Scaling this term nonlinearly yields an estimate of the phase x of the canonical system

$$x = \exp\left(-\alpha_x \left(\frac{n^*}{N}\right)\tau\right). \tag{9}$$

A simple example for the explained phase estimation algorithm can be see Fig 3. The grey trajectories show the demonstrated handwriting samples for which DMPs have been learned. The black trajectories show a new, partial handwriting sample. Using DTW, we can identify the phase of the DMP and then use it to automatically complete the observed letter. To this end, we can set the starting position of the DMP to the last point in the partial trajectory, and set the phase according to the estimated x. By specifying any goal position g, we can generate the missing part of the observed movement.

C. Predictive Distributions over DMPs

In this section we will introduce predictive distributions over DMP parameters that can be used to predict the behavior of an agent given a partial observed trajectory. We will first describe how to generate such predictive distribution for a single agent and later show in the next section how this model can be easily extended to infer a control policy for the controlled agent in an interaction scenario.

In our probabilistic approach, we model a distribution $p(\boldsymbol{\theta})$ over the parameters of a DMP. In order to also estimate the target position of the movement, we include the shape parameters $\boldsymbol{w}_i$ as well as the goal attractors g_i for all DoFs i in the parameter vector $\boldsymbol{\theta}$ of the DMP, i.e., $\boldsymbol{\theta} = [\boldsymbol{w}_1^T, g_1, \ldots, \boldsymbol{w}_N^T, g_N]$, where N is the number of DoFs for the agent. Given the parameter vector samples $\boldsymbol{\theta}^{[j]}$ of multiple demonstrations $j = 1 \ldots S$, we estimate a Gaussian distribution over the parameter $\boldsymbol{\theta}$, i.e., $p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}_{\boldsymbol{\theta}}, \boldsymbol{\Sigma}_{\boldsymbol{\theta}})$, with

$$\boldsymbol{\mu}_{\boldsymbol{\theta}} = \frac{\sum_{j=1}^{S} \boldsymbol{\theta}^{[j]}}{S}, \quad \boldsymbol{\Sigma}_{\boldsymbol{\theta}} = \frac{\sum_{j=1}^{S} (\boldsymbol{\theta}^{[j]} - \boldsymbol{\mu})(\boldsymbol{\theta}^{[j]} - \boldsymbol{\mu})^T}{S}. \tag{10}$$

In order to obtain a predictive distribution, we observe a partial trajectory $\boldsymbol{\tau}_o = \boldsymbol{y}_{1:t^*}$ up to time point t^* and our goal is to estimate the distribution $p(\boldsymbol{\theta}|\boldsymbol{\tau}_o)$ over the parameters $\boldsymbol{\theta}$ of the DMP. These parameters can be used to predict the remaining movement $\boldsymbol{y}_{t^*:T}$ of the observed agent. The updated distribution $p(\boldsymbol{\theta}|\boldsymbol{\tau}_o)$ can simply be computed by applying Bayes rule, i.e.,

$$p(\boldsymbol{\theta}|\boldsymbol{\tau}_o) \propto p(\boldsymbol{\tau}_o|\boldsymbol{\theta}) p(\boldsymbol{\theta}). \tag{11}$$

In order to compute this operation, we first have to discuss how the likelihood $p(\boldsymbol{\tau}_o|\boldsymbol{\theta})$ can be implemented. The parameters of a DMP directly specify the forcing function, however, we also include the goal positions g_i in the forcing function f_i for the ith DoF. Therefore, we reformulate the forcing function, i.e., for a single degree of freedom i, we will write the forcing function $f_i(t)$ as

$$f_i(x_t) = \boldsymbol{\phi}_t^T \boldsymbol{w}_i + \alpha_y \beta_y g_i \tag{12}$$

$$= [\boldsymbol{\phi}_t^T, \alpha_y \beta_y] \begin{bmatrix} \boldsymbol{w}_i \\ g_i \end{bmatrix}. \tag{13}$$

The forcing function can be written in matrix form for all DoFs of the observed agent and for all time steps $1 \leq t \leq t^*$, i.e.,

$$\boldsymbol{F} = \underbrace{\begin{bmatrix} \tilde{\boldsymbol{\Phi}} & \mathbf{0} & \ldots\ldots \\ \mathbf{0} & \tilde{\boldsymbol{\Phi}} & \mathbf{0} \ldots \\ \vdots & \vdots & \vdots \ \vdots \\ \mathbf{0} & \ldots\ldots & \tilde{\boldsymbol{\Phi}} \end{bmatrix}}_{\boldsymbol{\Omega}} \begin{bmatrix} \boldsymbol{w}_1 \\ g_1 \\ \vdots \\ \boldsymbol{w}_N \\ g_N \end{bmatrix} = \boldsymbol{\Omega}\boldsymbol{\theta}, \tag{14}$$

where the t-th row $\tilde{\boldsymbol{\Phi}}_{t,\cdot} = [\boldsymbol{\phi}_t^T, \alpha_y \beta_y]$ of $\tilde{\boldsymbol{\Phi}}$ contains the basis functions for time step t and a constant as basis for the goal attractor g_i. The matrix $\boldsymbol{\Omega}$ contains the basis functions for all DoFs on its block-diagonal. The vector $\boldsymbol{F}$ contains the value of the forcing function for all time steps and all degrees of freedom, i.e., $\boldsymbol{F} = [\boldsymbol{f}_1^T, \ldots, \boldsymbol{f}_N^T]^T$, where $\boldsymbol{f}_i$ contains the values of the forcing function for all time steps for the ith DoF. By concatenating the forcing vectors $\boldsymbol{f}_i$ for the single DoFs and by writing $\boldsymbol{\Omega}$ as a block-diagonal matrix, we achieve that the parameters of all DoF can be concatenated in the vector $\boldsymbol{\theta}$.

Given an observed trajectory $\boldsymbol{y}_{1:t^*}$, we can also compute the forcing function vector $\boldsymbol{F}^*$ from the observation, i.e., the elements of the $\boldsymbol{F}^*$ vector are given by

$$f_i^*(x_t) = \frac{1}{\tau^2}\ddot{o}_i(t) - \alpha_y(-\beta_y o_i(t) - \dot{o}_i(t)/\tau). \tag{15}$$

Now, we can use a Gaussian observation model for the likelihood

$$p(\boldsymbol{\tau}_o|\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{F}^*|\boldsymbol{\Omega}\boldsymbol{\theta}, \sigma^2 \boldsymbol{I}), \tag{16}$$

where σ^2 is the observation noise variance which will act as regularizer in the subsequent conditioning.

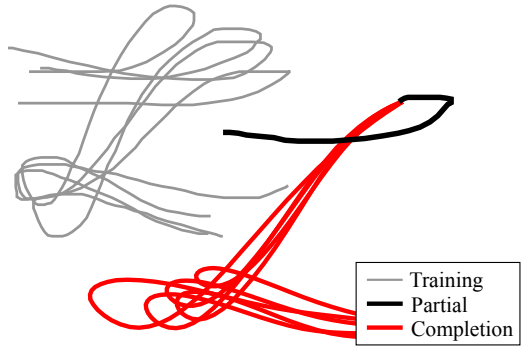

Fig. 4. Given a set of training trajectories (in gray) we learn a predictive distribution over the DMP weights. The distribution can then be used to sample new trajectories with a similar shape. In this example, DTW is used to determine the current phase of a partially observed trajectory (black). The completions of this trajectory are performed by estimating the most likely distribution of DMP weights.

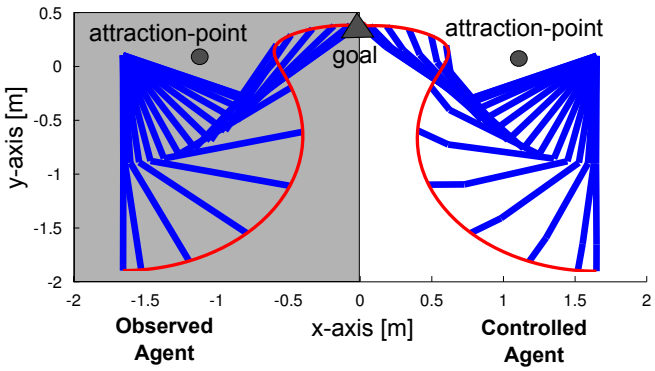

Fig. 5. A simple simulation setup for studying Interaction Primitives. Two opposing (robot) arms of different kinematic structure (2 links vs. 3 links) execute a high-five movement to a specific goal. Using optimal control methods we can calculate optimal trajectories that reach the goal, while at the same time being attracted by "attraction points". The resulting data set contains strong correlations between the movements of the two agents.

In order to perform the conditioning, we first write $p(\boldsymbol{\tau}_o, \boldsymbol{\theta}) = p(\boldsymbol{\tau}_o|\boldsymbol{\theta})p(\boldsymbol{\theta})$ as a joint Gaussian distribution and, subsequently, condition on $\boldsymbol{\tau}_o$. The joint distribution is given by

$$p(\boldsymbol{\tau}_o, \boldsymbol{\theta}) = \mathcal{N}\left(\begin{bmatrix}\boldsymbol{F}^* \\ \boldsymbol{\theta}\end{bmatrix} \middle| \begin{bmatrix}\boldsymbol{\Omega\theta} \\ \boldsymbol{\mu}_{\boldsymbol{\theta}}\end{bmatrix}, \begin{bmatrix}\boldsymbol{A} & \boldsymbol{\Omega\Sigma}_{\boldsymbol{\theta}} \\ \boldsymbol{\Sigma}_{\boldsymbol{\theta}}\boldsymbol{\Omega}^T & \boldsymbol{\Sigma}_{\boldsymbol{\theta}}\end{bmatrix}\right) \quad (17)$$

with $\boldsymbol{A} = \sigma^2\boldsymbol{I} + \boldsymbol{\Omega\Sigma}_{\boldsymbol{\theta}}\boldsymbol{\Omega}^T$.

The conditional distribution $p(\boldsymbol{\theta}|\boldsymbol{\tau}_o)$ is now also Gaussian with mean and variance

$$\begin{aligned}\boldsymbol{\mu}_{\boldsymbol{\theta}|\boldsymbol{\tau}_o} &= \boldsymbol{\mu}_{\boldsymbol{\theta}} + \boldsymbol{\Sigma}_{\boldsymbol{\theta}}\boldsymbol{\Omega}^T\boldsymbol{A}^{-1}(\boldsymbol{F}^* - \boldsymbol{\Omega\mu}_{\boldsymbol{\theta}}), \\ \boldsymbol{\Sigma}_{\boldsymbol{\theta}|\boldsymbol{\tau}_o} &= \boldsymbol{\Sigma}_{\boldsymbol{\theta}} - \boldsymbol{\Sigma}_{\boldsymbol{\theta}}\boldsymbol{\Omega}^T\boldsymbol{A}^{-1}\boldsymbol{\Omega\Sigma}_{\boldsymbol{\theta}}.\end{aligned} \quad (18)$$

Using the above distribution we can compute the most likely weights $\boldsymbol{\mu}_{\boldsymbol{\theta}|\boldsymbol{\tau}_o}$ of the DMPs for any observed partial movement. In Fig. 4, we see an example of using the above procedure for estimating weights from partial observations. On the left side, we see the demonstrations that have been used to train the DMPs. The different demonstrations reflect different versions of the same alphabet letter. On the right side, we see the partial observation (black) of a new handwritten sample as well as the automatic reconstruction using a DMP with estimated weights.

D. Correlating two Agents with Predictive Distributions

Correlating the controlled agent with its interaction partner is now a straightforward extension of the predictive DMP framework. We now assume that we have two agents, the observed and the controlled agent. In order to capture the correlation between the agents, we use a combined parameter vector $\boldsymbol{\theta} = [\boldsymbol{\theta}_o^T, \boldsymbol{\theta}_c^T]^T$ which contains the DMP parameters $\boldsymbol{\theta}_o$ of the observed and the parameters $\boldsymbol{\theta}_c$ of the controlled agent. We can now use the approach for obtaining a predictive distribution $p(\boldsymbol{\theta}|\boldsymbol{\tau}_o)$, however, in contrast to the previous section, the observed trajectory only contains the DoFs of the observed agent. Hence, in order to write the forcing vector $\boldsymbol{F} = \boldsymbol{\Omega\theta}_o$ in terms of the complete parameter vector $\boldsymbol{\theta}$, we need to append a zero-matrix to the feature matrix $\boldsymbol{\Omega}$, i.e., $\boldsymbol{F} = \tilde{\boldsymbol{\Omega}}\boldsymbol{\theta}$, with $\tilde{\boldsymbol{\Omega}} = [\boldsymbol{\Omega}, \boldsymbol{0}]$. The conditioning equations given in (18) can now be applied straightforwardly by replacing $\boldsymbol{\Omega}$ with $\tilde{\boldsymbol{\Omega}}$. The estimated parameter vector $\boldsymbol{\theta}$ can, thus, be used to to predict the remaining movement of the observed agent (using $\boldsymbol{\theta}_o$) but also to control the robot in the current situation (using $\boldsymbol{\theta}_c$). Hence, its behavior is always related the behavior of the human interaction partner.

IV. Experiments

To evaluate our approach under controlled and varying conditions, we designed a simple simulation environment that can be used to study interaction scenarios. The simulation consists of two opposing robot arms with different kinematic structures as can be seen in Fig. 5. The robot on the left side (the observed agent) consists of two links which are connected through a hinge joint, while the robot on the right side (the controlled agent) has three links and two joints. The task of the robots is to execute a high-five movement. In order to generate training data for this task we first synthesize a set of training trajectories for both agents. Using optimal control we determine for each agent a joint angle trajectory which brings it from its start position to a specified goal position. Attractor points are added in order to generate realistic-looking high-five movements. During the synthesis of training data, both the goal position and the attractor positions are varied.

Given this training data, we learn an IP capturing the mutual dependencies of the agents during the high-five movement. After learning, we the use the IP to control the three linked robot arm. Given partial joint angle trajectories of the observed agent, we use the IP and conditioning to (1) determine the most likely current goal, (2) the ideal joint angle configurations of the controlled agent.

Fig. 6 depicts task space trajectories for the controlled agent after conditioning. On the left side we see the results of conditioning when 40% of the movement of the observed agent is seen. On the right side are the results after conditioning with 60% of the data. Each trajectory is a possible

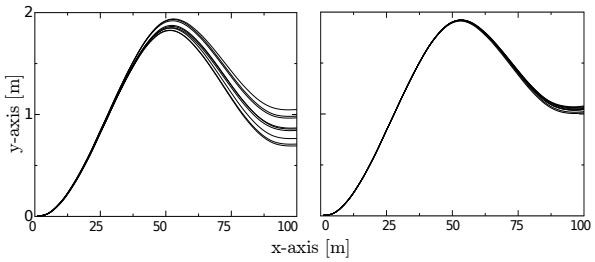

Fig. 6. The uncertainty over the goal position shrinks with increasing amount of data points. Left: distribution after observing 40% of the partners movements. Right: distribution after observing 60% of the partners movements

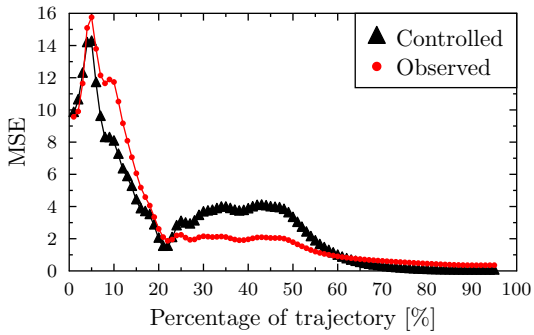

Fig. 7. Mean squared error based on different percentage of partially observed trajectories of the interaction partner. The red curve shows the accuracy of predicting the movements of the observed agent from partial trajectories. The black curve shows the accuracy in inferring the right reaction in response to the observed movement.

prediction of the task space movement of the agent. The figure shows that the uncertainty significantly shrinks when we transition fom 40% to 60% in this example.

To further analyze the accuracy of prediction, we generated a new test data set consisting of interactions to different goal positions that were not part of the original training data. We then calculated the mean squared error (MSE) between the predicted joint angle trajectories generated from the IP and the ground-truth data. Fig. 7 shows the evolution of the MSE for observed partial trajectories of increasing size. We can see that the MSE in the prediction of the observed agent significantly drops at around 20% and again at 60%. A similar trend, albeit with higher variance, can also be seen in the predictions for the ideal response of the controlled agent. The plot suggests that after seeing 60% of the movement of the interaction partner we can already roughly infer the goal he is aiming at.

We also analyzed the difference in task space between the inferred and the ground-truth data for the controlled agent. More specifically, we evaluated how far the predicted goal is from the true goal of the trajectory. Fig. 8 shows the true goals and the inferred goals in task space after seeing 60% of the movements of the observed agent. We can see that the predicted goals are in close proximity to the true goal locations. Please note, that the depicted positions are in task space while the inferred values are in joint space.

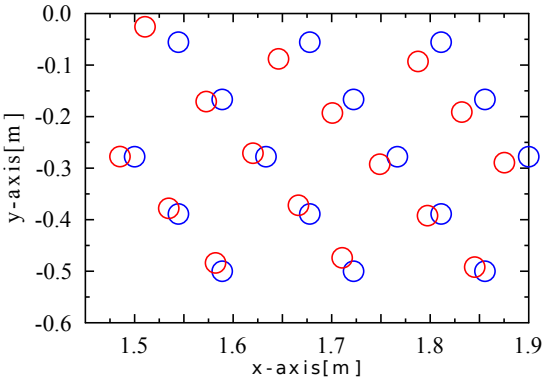

Fig. 8. Difference between ground truth and predicted task space goals. Blue circles show the true positions of the goals. Red circles depict the predicted goals after observing 60% of the interaction.

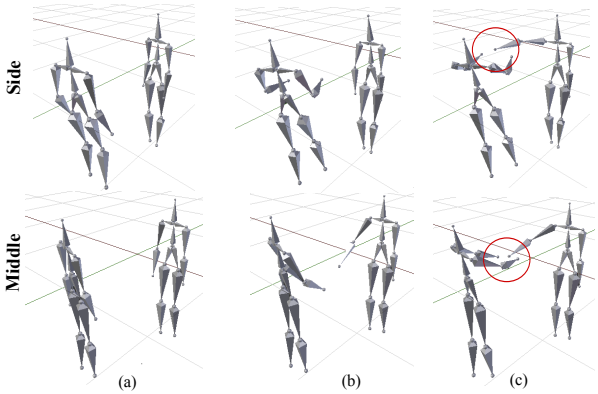

Fig. 9. Two reactions synthesized from an Interaction Primitive for a handing-over task. The humanoid on the left side (controlled agent) is controlled through the IP and has to receive an object from the humanoid on the right side. Depending on the location where the object is handed over, the reaction of the controlled agent is changed. Training of the IP was performed using motion capture data of two humans.

The depicted position is calculated by performing forward kinematics using the inferred joint angle data. As a result, even small errors in the joint angle of the first link will propagate and produce larger errors at the end-effector. Still, the results indicate that the approach can be efficiently used for intention inference in human-robot tasks. By predicting the most likely goal of the human interaction partner, the robot can proactively initiate his own response.

We also conducted experiments using real motion capture data collected from the interactions of two human subjects. In a first experiment we asked two humans to demonstrate a "handing-over" in which the first subject hands a cup to the second subject. The demonstrations were then used to learn an IP. However, in order to cope with the high dimensionality of the dataset, we applied Principal Component Analysis (PCA) as a pre-processing step, in order to project the data onto a five-dimensional space. After training, we tested the learned IP on a set of data points that have not been used during training. Fig. 9 shows two example situations, where the controlled agent (left humanoid) automatically infers the optimal reaction to the behavior of his interaction partner

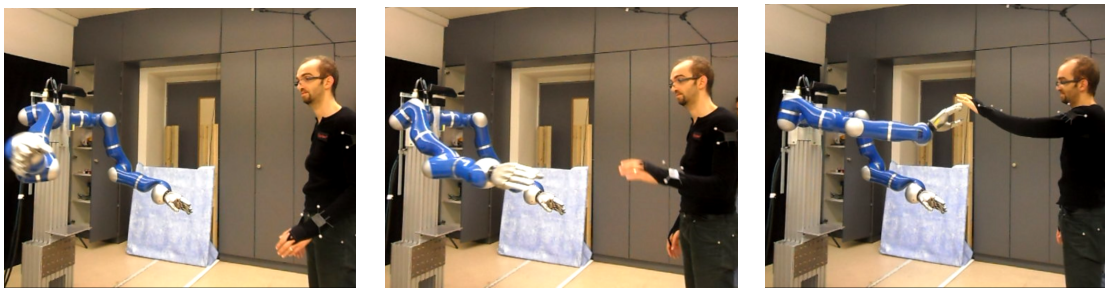

Fig. 10. A frame sequence from a high-five interaction between a human and a humanoid. The robot automatically reacts to the movement of the human and estimates the appripriate location of the executed high-five. The human interaction partner is tracked using an OptiTrack motion capture system.

(right humanoid). In the first sequence the controlled agent correctly turns to the left side to receive an object. In contrast to that, in the second sequence, the agent reaches for the middle in order to properly react to the observed movement.

Finally, we performed a set of interaction experiments on a real humanoid robot. The humanoid has two arms with 7 DoF each. During the experiment we used one arm with four DoFs. More specifically, we trained an Interaction Primitive for the high-five. Again, we collected motion capture data from two humans for training this IP. After training, the robot used the IP to predict the joint configuration at the goal position as well as the weight parameters of the DMP. Fig. 10 shows an example interaction realized via the presented approach. Using prediction in this task is important, as it helps the robot to match the timing of the human interaction partner. Notice that the starting location of the robot is quite far from the rest poses in the learning database.

V. Conclusion

In this paper, we proposed the novel Interaction Primitive framework based on DMPs and introduced a set of basic algorithmic tools for synchronizing, adapting, and correlating motor primitives between cooperating agents. The research introduced here lays the foundation for imitation learning methods that are geared towards multi-agent scenarios. We showed how demonstrations recorded from *two* interacting persons can be used to learn an interaction primitive, which specifies both the executed movements, as well as the correlations in the executed movements. The introduced phase estimation method based on dynamic time warp proved to very important for applying a learned interaction primitive in new situations. Timing is a highly variable parameter, which varies among different persons, but can also vary depending on the current mood or fatigue.

In future work, we plan to concentrate on more complex interaction scenarios, which are composes of several interaction primitives. These primitives could executed in sequence or in a hierarchy in order to produce complex interactions with a human partner. We are also already working on using the interaction primitive framework for predicting the most likely future movements of a human interaction partner. The underlying idea is that the same representation which is used for movement synthesis can also be used for movement prediction. The predicted actions of the human could then be integrated into the action selection process of the robot, in order to avoid any dangerous situations.

VI. Acknowledgements

The work presented in this paper is funded through the Daimler-and-Benz Foundation and the European Communitys Seventh Framework Programme under the grant agreement n ICT-600716 (CoDyCo).

References

[1] P. Abbeel, A. Coates, and A. Ng. Autonomous Helicopter Aerobatics through Apprenticeship Learning. *International Journal of Robotic Research*, 29:1608–1639, 2010.

[2] A. Billard, S. Calinon, R. Dillmann, and S. Schaal. Survey: Robot Programming by Demonstration. In *Handbook of Robotics*, volume chapter 59. MIT Press, 2008.

[3] S. Calinon, E.L. Sauser, A.G. Billard, and D.G. Caldwell. Evaluation of a probabilistic approach to learn and reproduce gestures by imitation. In *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, pages 2381–2388, Anchorage, Alaska, USA, May 2010.

[4] R. Chalodhorn, D. Grimes, K. Grochow, and R. Rao. Learning to walk through imitation. In *Proceedings of the 20th international joint conference on Artifical intelligence*, IJCAI'07, pages 2084–2090, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

[5] D. Grimes, R. Chalodhorn, and R. Rao. Dynamic imitation in a humanoid robot through nonparametric probabilistic inference. In *In Proceedings of Robotics: Science and Systems (RSS)*. MIT Press, 2006.

[6] A. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal. Dynamical movement primitives: Learning attractor models for motor behaviors. *Neural Comput.*, 25(2):328–373, February 2013.

[7] S. Ikemoto, H Ben Amor, T. Minato, B. Jung, and H. Ishiguro. Physical human-robot interaction: Mutual learning and adaptation. *IEEE Robotics and Automation Magazine*, 19(4):24–35, Dec.

[8] D. Lee, C. Ott, and Y. Nakamura. Mimetic communication model with compliant physical contact in human-humanoid interaction. *Int. Journal of Robotics Research.*, 29(13):1684–1704, November 2010.

[9] M. Mühlig, M. Gienger, and J. Steil. Interactive imitation learning of object movement skills. *Autonomous Robots*, 32:97–114, 2012.

[10] D. Pomerleau. ALVINN: an autonomous land vehicle in a neural network. In David S. Touretzky, editor, *Advances in Neural Information Processing Systems 1*, pages 305–313. San Francisco, CA: Morgan Kaufmann, 1989.

[11] H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 26(1):43–49, 1978.

[12] S. Schaal. Is imitation learning the route to humanoid robots? *Trends in Cognitive Sciences*, 3:233–242, 1999.

[13] Z. Wang, M. Deisenroth, H. Ben Amor, D. Vogt, B. Schoelkopf, and J Peters. Probabilistic modeling of human dynamics for intention inference. In *Proceedings of Robotics: Science and Systems (R:SS)*, 2012.