2013 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)
November 3-7, 2013. Tokyo, Japan

Learning Responsive Robot Behavior by Imitation

Heni Ben Amor[1], David Vogt[2], Marco Ewerton[1], Erik Berger[2], Bernhard Jung[2], Jan Peters[1]

Abstract— **In this paper we present a new approach for learning responsive robot behavior by imitation of human interaction partners. Extending previous work on robot imitation learning, that has so far mostly concentrated on learning from demonstrations by a single actor, we simultaneously record the movements of two humans engaged in on-going interaction tasks and learn compact models of the interaction. Extracted interaction models can thereafter be used by a robot to engage in a similar interaction with a human partner. We present two algorithms for deriving interaction models from motion capture data as well as experimental results on a humanoid robot.**

I. INTRODUCTION

While robots are becoming increasingly better at performing a wide range of motor skills, they are still limited in their human interaction capabilities. To date, most robots are not prepared to appropriately respond to the movements or the behavior of a human partner. However, with application domains of robots coming closer to our everyday life, there is a need for adaptive algorithms that ensure responsive robot behavior for human-robot interaction.

We present a new approach to robot learning that allows anthropomorphic robots to learn a library of interaction skills from demonstration. Traditional approaches to modelling interactions assume a pre-specified symbolic representation of the available actions. For example, they model interactions in terms of commands such as *wait*, *pick-up*, and *place*. Instead of such a top-down approach, we want to focus on learning responsive behavior in a bottom-up fashion using a trajectory based approach. The key idea behind our approach is that the observation of human-human collaborations can provide rich information specifying how and when to interact in a particular situation. For example, by observing how two human workmen collaborate on lifting a heavy box, a robot could use machine learning algorithms to extract an *interaction model* that specifies the states, movements, and situational responses of the involved parties. In turn, such a model can be used by the robot to assist in a similar lifting task. Our approach is as an extension of imitation learning [3] to multi-agent scenarios, in which the behavior and the mutual interplay between two agents is imitated.

In this paper, we describe the general multi-agent imitation learning setup for learning interaction models from motion capture data. We also provide two first algorithms that enable a robot to learn such interaction models between interacting agents. The first algorithm *PPCA-IM* (Probabilistic Principal Component Analysis-Interaction Model) frames the task as a missing value estimation problem. The second algorithm called *PM-IM* (Path Map-Interaction Model) uses a Hidden Markov Model (HMM) [20] to represent the mutual dependencies of the interacting agents. A set of shared latent states is used to map the behavior of one agent to the behavior of the interaction partner. The principal difference between the two algorithms presented in this paper is the representation of the temporal dynamics of interaction. The PPCA-IM uses an implicit representation of time via a temporal embedding of the training data. In contrast, the PM-IM uses an explicit representation of time via a discrete set of hidden nodes.

Through a series of experiments, we will show how the two algorithms can be used to create a responsive robot that learns to react to the movements and gestures of humans. We will also provide a comparison of PPCA-IM and PM-IM and discuss the advantages and drawbacks of each approach.

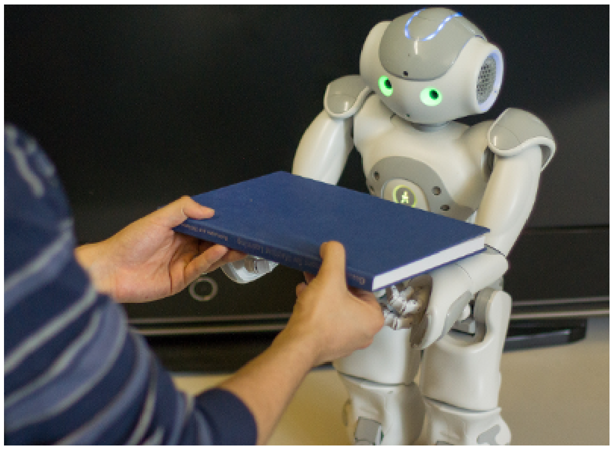
Fig. 1. A humanoid robot receives a book that is handed over by a human interaction partner. The robot learned what to do in this situation by observing a similar situation between two humans.

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 [22]. Based on human-provided demonstrations of a specific skill, a robot autonomously generates a control program that allows it to generalize the skill to different situations. Most approaches 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

[1]Heni Ben Amor, Marco Ewerton and Jan Peters are with the Technische Universitaet Darmstadt, Intelligent Autonomous Systems, Darmstadt, Germany. {amor,ewerton,peters}@ias.tu-darmstadt.de

[2]David Vogt, Erik Berger and Bernhard Jung are with the Technische Universitaet Bergakademie Freiberg, Virtual Reality and Multimedia Group, Freiberg, Germany. {david.vogt,bergere,jungb}@tu-freiberg.de

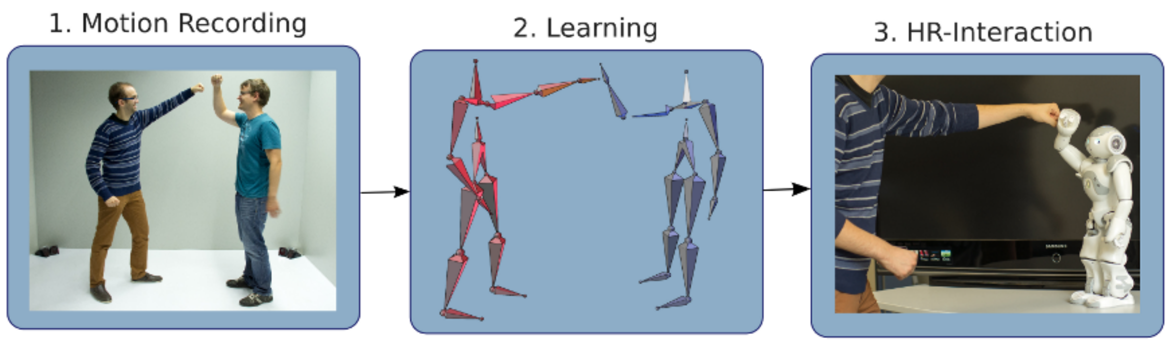

Fig. 2. Overview of the interaction learning approach presented in this paper. The interaction behavior of two humans is observed, analyzed and imitated in order in human-robot interaction scenarios. Left: The movements of two persons are recorded using motion capture technology. Middle: A compact interaction model specifying the mutual influences and responses is learned. Right: The interaction model enables a robot to compute the best response to the current behavior of a human interaction partner.

adapted to the current situation.

For example, the Dynamical Motor Primitive (DMP) [13] approach uses dynamical systems to represent control policies. The DMP approach has been widely accepted in the imitation learning community and has been used to learn various motor skills such as locomotion [16], or drumming [13]. Another way of encoding policies is to use statistical modelling methods. For example, in the Mimesis Model [17] 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. [7] who used Gaussian Mixture Regression to learn gestures. The advantage of statistical and probabilistic approaches, is the ability to naturally model the spatial and temporal variability of human motion.

The methods discussed so far are limited to single agent imitation learning scenarios. Once the behavior is learned, it is executed without taking into account the reaction of an interaction partner. In recent years, various attempts have been undertaken for using machine learning in human-robot interaction scenarios. In [15], a recurrent neural network was used to learn a simple interaction game between a human and a robot. More recently, Wang et al. [24] presented an extension of the Gaussian Process Dynamics Model that 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 [14], Gaussian mixture models were used to adapt the timing of a humanoid robot to that of 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 [17] 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 contrast, in our approach, learning of motion and interaction is not split into two parts. Instead, we learn one integrated interaction model which can directly synthesize an appropriate movement in response to an observed movement of the human partner. Further, instead of modelling symbolic action-reaction pairs, our approach is based on modelling the joint dynamics *during* the execution of a movement.

In general, while the learning approaches discussed above are placed within human-robot interaction settings, they only learn from demonstrations by a single actor at a time. In contrast, the work presented here focuses on imitation learning from simultaneously recorded movements by two human interaction partners in order to learn integrated models of the joint interaction.

III. Learning Interaction Models

The goal of learning an interaction model is to derive a compact representation of how two agents behave and, in particular, how they react to each other when they perform a cooperative or competitive task together. The approach followed in this paper derives such a representation from observations of human-human interactions. In Figure 2 we see an overview of this approach. First, the movements of a pair of persons performing a competitive (or cooperative) task are recorded using motion capture technology. Subsequently, an interaction model is learned from the recorded data. The interaction model captures the reciprocal influences during the execution of the task. In turn, the interaction model enables us to predict the state (skeletal configuration) of one human based on the observed states of the second human. Finally, the learned model is used by a robot to engage in a similar interaction with a human partner. In the example depicted in Figure 2, the humanoid robot learns to perfom defensive movements in response to a human performing punching movements.

An interaction model can be regarded as a mapping from the current state of one agent to the state of a second agent. In our particular application, we want to learn a mapping from the state of the *opponent agent* (i.e. the human) to the state of the *controlled agent* (i.e. the robot)[1]. Input to the learning algorithms are the two data sets $\mathbf{A}$ (controlled agent) and $\mathbf{B}$ (opponent agent) consisting of joint angle configurations of the two agents. Each point in $\mathbf{A}$ contains information about the skeletal configuration of the controlled agent at a particular time step, while $\mathbf{B}$ contains a joint angle configurations for the opponent agent. Once a mapping from $\mathbf{B}$ to $\mathbf{A}$ is learned, it can be used to compute the most appropriate response of the controlled agent, given the observed movements of the opponent agent. In this section, we will present two algorithms that can learn such a mapping.

A. Algorithm 1: PPCA-IM

The first algorithm that we present is the Probabilistic Principal Component Analysis - Interaction Model. The method exploits the low-dimensional nature of human movement in order to create a compact model of the interaction. It is well known from human motor control, that motor tasks, e.g., grasping [21], walking [9], and also interactions between humans [4] lie on low-dimensional manifolds. Such a manifold typically has a much smaller dimensionality than the total number of joints involved in the motor task. Therefore, instead of finding a mapping in the high-dimensional space involving all joints, we can find a low-dimensional space in which the relationship between the postures and movements can be learned in a more efficient way. After learning is finished, the joint values of the controlled agent are treated as missing values that are estimated by maximizing the likelihood in the low-dimensional latent space.

The first step to PPCA-IM is the temporal embedding of the opponent agent's data. Temporal embedding allows us to disambiguate between similar movements by including information from prior time steps. In the second step, namely dimensionality reduction, we then compute a low-dimensional projection of the data set and use this space to estimate the most likely response for the controlled agent.

1) Temporal Embedding: One possible approach to learning an interaction model is to learn a mapping between individual pairs of samples in $\mathbf{A}$ and $\mathbf{B}$ directly. However, such an approach does not take the temporal offset between action and reaction into account, and is therefore prone to fail for many behaviors. For example, a stretched out arm can either mean that the opponent wants to shake hands or perform a Karate movement. To disambiguate the behavior in such scenarios it is important to take the temporal development of the movement into account. When using PPCA-IM we perform a temporal embedding of the data in $\mathbf{B}$ yielding a new data set $\mathbf{B}^*$. To each point in $\mathbf{B}$ we add joint angles of the τ last time steps:

$$\mathbf{b}_t^* = \begin{bmatrix} \mathbf{b}_t, \\ \vdots \\ \mathbf{b}_{t-\tau} \end{bmatrix} \forall\ \mathbf{b}_t \in \mathbf{B}, t > \tau. \tag{1}$$

This embedding is comparable to modelling the interaction as a Markov chain of order τ, rather than a traditional first-order Markov chain. A similar preprocessing of the data was proposed in [2].

2) Dimensionality Reduction: The next step in PPCA-IM is to create a shared latent space that models the interaction dynamics of the two agents. As the name of the algorithm suggests, we use Probabilistic Principal Component Analysis (PPCA) to learn a shared low-dimensional representation of the movements. To this end, we create a combined data set $\mathbf{Z}$ which is a concatenation of $\mathbf{A}$ and $\mathbf{B}^*$:

$$\mathbf{z} = [\mathbf{a}_t, \mathbf{b}_t^*]\ \forall\ \mathbf{a}_t, \mathbf{b}_t^* \in \mathbf{A}, \mathbf{B}^* \tag{2}$$

On the new data set $\mathbf{Z}$ we can then perform PPCA. PPCA is an iterative version of the original PCA algorithm which uses Expectation-Maximization (EM) [11] to determine the optimal projection matrix $\mathbf{C}$ that maps the data set $\mathbf{Z}$ onto a lower-dimensional principal component space. The PPCA algorithm used here is based on [19]. An advantage of PPCA over PCA is that it provides a probabilistic framework for performing PCA on data sets that have missing values. In our case, we treat the current joint angles of the controlled agent as missing values that need to be estimated. The EM-algorithm can be used to estimate the missing values of our data. This estimation is done by adding an additional entry $\mathbf{z}$ to $\mathbf{Z}$, which consists of an observed part $\mathbf{z}_o$ and a hidden part $\mathbf{z}_h$. The observed part contains (temporally embedded) joint values $\mathbf{z}_o$ of the opponent agent. The hidden part $\mathbf{z}_h$ will be estimated during the EM-algorithm and is initially filled with zeros.

Before starting the EM algorithm, we initialize the projection matrix $\mathbf{C}$ and the variance σ^2 with random values between zero and one. In the E-step, we first calculate new estimates for the covariance matrix $\mathbf{\Sigma}$ and matrix $\mathbf{Y}$ of projected points:

E-Step:

$$\begin{aligned} \mathbf{\Sigma} &\rightarrow \left[\mathbf{I} + \sigma^{-2}\mathbf{C}\mathbf{C}^T\right]^{-1}, \\ \mathbf{Y} &\rightarrow \sigma^{-2}\mathbf{\Sigma}\mathbf{C}\mathbf{Z}, \end{aligned}$$

Based on these estimates, we can update in the M-step the projection matrix $\mathbf{C}$ and the variance σ^2:

M-Step:

$$\begin{aligned} \mathbf{C} &\rightarrow \mathbf{Z}\mathbf{Y}^T(S\mathbf{\Sigma} + \mathbf{Y}\mathbf{Y}^T)^{-1}, \\ \sigma^2 &\rightarrow \frac{1}{SD}\left[S\ \mathrm{Tr}\{\mathbf{C}^T\mathbf{\Sigma}\mathbf{C}\} + \sum_{i=1}^{S} ||\mathbf{z}_i - \mathbf{C}^T\mathbf{y}_i||^2 + D_h\sigma^2\right], \end{aligned}$$

where S is the number of samples, D is the dimensionality of the samples, and D_h is the total number of missing

[1]For the sake of clarity we will henceforth use the terms controlled agent and opponent agent to refer to the agents involved in an interaction. Note that this naming convention does not restrict the application to competitive tasks only.

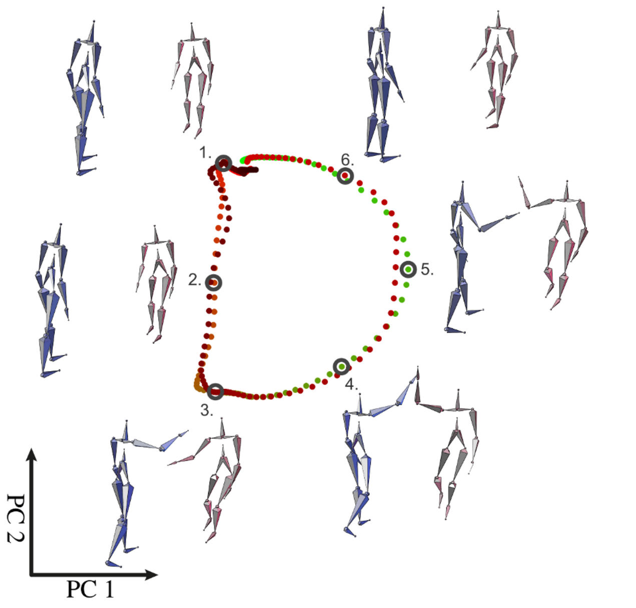

Fig. 3. The projection of *high-five* motions into a low-dimensional space using Probabilistic Principal Component Analysis. Each point in this space corresponds to specific interaction situation and defines the postures of both agents. Even if we observe the postures of one agent only, we can still infer the most likely posture of the interaction partner using PPCA and missing value estimation.

values in $\mathbf{Z}$. The missing values $\mathbf{z}_h$ of the matrix $\mathbf{Z}$ are re-estimated before performing the next E-Step by first calculating $\mathbf{Z}_{estim}$:

$$\mathbf{Z}_{estim} = \mathbf{C}^T\mathbf{Y}, \tag{3}$$

and then replacing the missing values $\mathbf{z}_h$ with the newly estimated values from $\mathbf{Z}_{estim}$. The above EM-steps can be iterated until the change in the error of the following objective function is below a given threshold (in our experiments the threshold is 10^{-5}):

$$\Psi(\mathbf{C},\sigma) = SD\log\sigma^2 + \sigma^{-2}D_h\sigma^2_{old} + \sigma^{-2}\left[\sum_{i=1}^{S}||\mathbf{z}_i - \mathbf{C}^T\mathbf{y}_i||^2 + \text{Tr}\left\{\mathbf{C}^T\mathbf{\Sigma}\mathbf{C}\right\}\right],$$

where σ^2_{old} is the previous value for the variance. Once the EM-algorithm is finished, we can use the missing values $\mathbf{z}_h$ as the new desired joint angles for the controlled agent.

Figure 3 shows the low-dimensional projection of a high-five interaction calculated with PPCA. Each point in the low-dimensional space encodes the reaction of the controlled agent with respect to the previous movements of the opponent agent. We can see that the interaction forms a smooth trajectory in the low-dimensional space.

To better understand the role of temporal embedding in our learning algorithm, we computed several PPCA projections with different values for the parameter τ (from Equation (1)). For this purpose, we recorded a defensive movement in which a human starts in a rest pose, then moves both arms in a defensive stance, and finally goes back to the rest pose. Figure 4 shows the projected movement for different values of parameter τ.

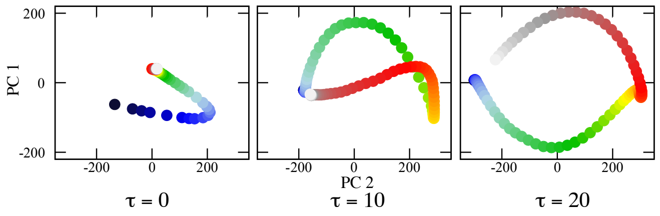

Fig. 4. PPCA projections of a defense interaction for different values of τ. During the interaction the opponent agent attacks and retracts back to the rest pose while the controlled agent goes to defense stance and then retracts to the rest pose. When $\tau = 0$ the temporal context of a posture is not taken into account.

When $\tau = 0$ (Figure 4 left), the postures for going *towards* the defensive stance (green) and the postures for retracting *back* from the defensive stance (red) are mapped onto the same points in the low-dimensional space. As a result a robot cannot distinguish between the two different modes of this particular movement and would produce the same reaction in both situations. We can also see in Figure 4, that with increased value for τ the points are more and more disentangled. We can see that $\tau = 20$ produces a clear separation between the two modes, i.e. going to and pulling back from the defense stance, of the movement. The trajectory starts at the rest posture in $(-250, 0)^T$, moves to the defense stance at $(250, 0)^T$, and then moves back to a position close to the rest posture. Note, that in this example we used a two-dimensional projection for visualization purposes only. To find a suitable value for the dimensionality of the low-dimensional space, we can use intrinsic dimensionality estimation methods [5]. A simpler approach, which was used in this paper, is to use the number of principal components that insures that 95% of the information in our training data is retained after PPCA.

B. *Algorithm 2: PM-IM*

The second algorithm for learning interaction models is called Path Map-Interaction Model (PM-IM). The algorithm uses a HMM to represent the mutual dependency of the interaction partners at different time steps.

A HMM is an efficient tool for modelling probability distributions over time-series'. It assumes that a set of observations was generated by a process with hidden internal states. Given the Markov assumption, any state s_t only depends on the predecessor state s_{t-1}. Following the notation in [20] and [6], a HMM can be defined as the tuple $\theta = (\mathcal{S}, \pi, P_{j\rightarrow i}, p_i(\mathbf{o}|s_i))$ where

- $\mathcal{S} = \{s_1, ..., s_N\}$ is the set of states s_i of the HMM
- $\pi = \{\pi_1, ..., \pi_N\}$ is a probability distribution specifying the probability π_i of starting in state i
- $P_{j\rightarrow i}$ is the state transition matrix defining the probability $p(s_j|s_i)$of transitioning from state s_i to s_j

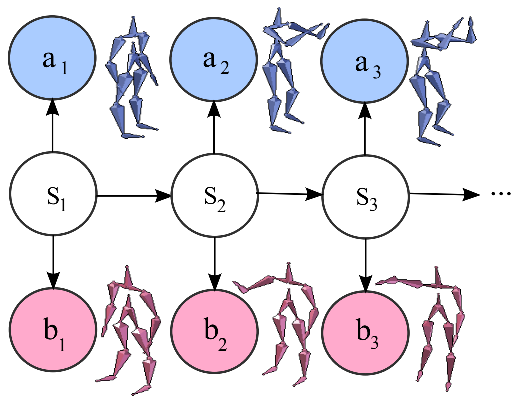

Fig. 5. The graphical model of a path map Hidden Markov Model. Each hidden node (white) is connected to two observed nodes (colored) corresponding to each of the interacting agents. Each observed node contains the joint angle configuration of the respective agent which is depicted by a small skeleton. The diagram shows a path map for an punch/defense interaction.

- $p(\mathbf{o}|s_i)$ is the emission probability distribution which defines the probability of observing output $\mathbf{o}$ while in state s_i. The emission probability distribution is modelled using a Gaussian distribution.

As already mentioned, the nodes of an HMM can be divided into observable nodes and hidden nodes. Typically, an HMM is defined in such a way that each hidden node is connected to one observable node only. In the following, however, we will use an extension of HMM, sometimes also referred to as a *path map*[6], which has a different graph structure.

A path map relates the time-series behavior of a cue system to the behavior a target system. This is achieved by connecting each hidden node to two observables nodes: one observable for the cue system and one observable for the target system. A path map for the task of interaction modelling can be seen in Figure 5. The colored nodes correspond to the observables of the controlled agent (blue) and the opponent agent (red) respectively. Each observable state holds the full joint angle configuration of the respective agent in the current situation. The white nodes depict the hidden states of the interaction task. Each hidden state models a specific context or situation during the interaction.

In contrast to the standard HMM, a path map contains two emission probability distributions $p_A(\mathbf{a}_t|s_t)$ and $p_B(\mathbf{b}_t|s_t)$; one for each of the two agents. The training of the path map, however, can be performed using the same approach as for a standard HMM. First, a K-Means[18] clustering algorithm is used to initialize the hidden states of the HMM. Using the EM [11] algorithm, we can then estimate all missing parameters of the HMM. A detailed description of HMM training can be found in [20]. Once it is learned, the path map in Figure 5 allows us to estimate the behavior of one agent by observing the movements of the other. We first calculate the most likely sequence of states given the observed behavior of the opponent agent. Using the emission probability distribution $p_B(\mathbf{b}_t|s_t)$ of each state, we can then generate an appropriate response for the controlled agent in every situation.

An interesting feature of HMMs is the ability to use several HMM models in parallel. For example, assume that we learn two HMMs for different interaction tasks, e.g., punching and handing-over. Given a new observed movement of the attacking agent, we can calculate the likelihood of this movement with respect to each learned HMM and select the model with highest likelihood according to:

$$\theta^* = \arg\max_{\theta} p(\mathbf{b}_t|\theta). \qquad (4)$$

Once the HMM with highest likelihood is selected, we can calculate the emissions for the current situation and use the resulting joint angle values for controlling the robot. The above feature allows us to use the knowledge of several HMMs in order to recognize an interaction scenario and also to respond to the behavior of the human partner. The set of interaction skills can, therefore, be gradually expanded.

IV. Experiments

To evaluate the algorithms proposed in this paper, we conducted a set of interaction experiments and analyzed the results. In the following sections, we will report the results achieved by applying the algorithms in simulation as well as on a real robot performing human-robot interaction with a human partner.

A. Interaction Data

Before training any specific model, we first collected a set of training data representing different competitive and cooperative interaction tasks. Specifically, we collected motion capture data of two interacting humans. The data set consisted of various interaction tasks acquired from the CMU Motion Capture Library[2], as well as additional data gathered using two Kinect cameras and two human subjects. In order to be independent of a specific tracking device, we transformed all motion capture data into the BioVision file format and used the resulting joint angle information as input to the learning algorithms. In total, we used 18 joints, with each joint being parametrized by three joint angles. The final data set consisted of four different interaction tasks:

- Boxing: One agent attacks with punches at different heights and from different directions while the other agent defends.
- Martial Arts: One agent attacks with punches and kicks while the other defends.
- High Five: Both agents perform a high-five movement.
- Handing over: One agent hands a book over to the other agent.

B. Runtimes

In interactive scenarios, a robot needs to quickly respond to the behavior of the human partner. In the following we will, therefore, analyze the computational demands of the proposed algorithms and the runtimes for predicting the appropriate response in the current situation. Figure 6 depicts

[2]http://mocap.cs.cmu.edu/

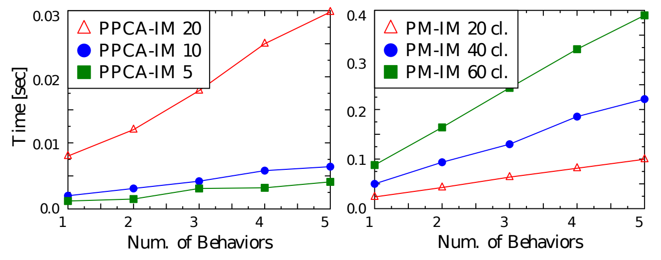

Fig. 6. The runtime of the PPCA-IM and the PM-IM algorithms. The values indicate the measured time needed for predicting the optimal response of the robot given the human's action at a particular time step. With increasing number of learned behaviors, the time needed to predict the optimal response increases, too. Additionally, the plots also show how the size of the temporal embedding window (20,10,5) affects the runtime of the PPCA. The right plot shows how the number of states/clusters affects the runtime of the PM-IM.

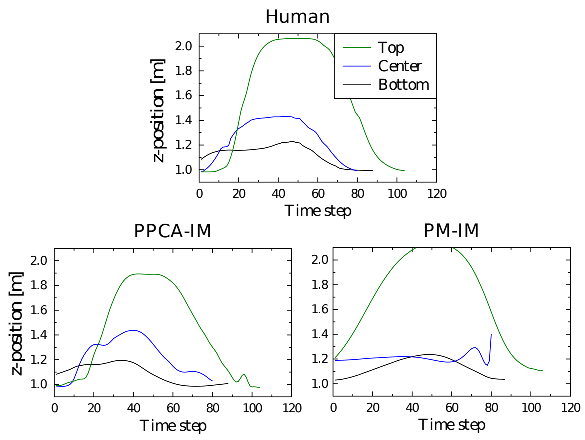

Fig. 7. The wrist position of the controlled agent for different defenses. The human raises his hand to different heights depending on the type of punch he receives. The defense movement for a center punch was not learned. The PPCA-IM algorithm is still able to generalize to this situation. The PM-IM algorithm does not generalize well in this situation.

the runtimes of the PPCA-IM and the PM-IM algorithms. For training we used an increasingly complex data set with one to five different behaviors. Each behavior consisted of approx. 120 data samples. The plots show how the number of interactive behaviors affects the response time of the robot when applying an interaction model *after* learning.

As can be seen in Figure 6, PPCA-IM has a significantly faster response time. Especially, with increasing number of clusters/states in the PM-IM, the response times quickly deteriorate. With 60 hidden states, the PM-IM requires about 0.4 seconds to compute a prediction. A smaller number of states can be used to speed up the algorithm. However, this comes at the price of a significantly lower quality of the learned model. In the above example, the PM-IM was only able to produce accurate responses when 55 or more states were used. The PPCA-IM was used with a 7 dimensional latent space.

C. Generalization

Another important feature of interaction models is the ability to generalize learned behaviors to new situations. To analyze the generalization ability, we conducted a set of experiments in which we trained interaction models for a boxing/defending behavior. The models were trained with high- and low-punches, and were later tested with several other punches that aimed at a position inbetween the trained punches. Figure 7 shows the z-position of the wrist of the controlled agent while trying to defend several punches.

The gound truth data gathered from the human clearly shows that the hand needs to be lifted to different levels, in order to defend from the upcoming punch. The trajectories generated with the PPCA-IM model have similar characteristics to the human trajectories. The blue trajectory in Figure 7 corresponds to a movement that was not seen in the training data. Despite that it was not trained, the PPCA-IM was still able to generalize the learned movement to this new situation. Both the shape and height of the trajectory are close to the ground truth of the human demonstrator. In contrast to that, the PM-IM does not exhibit a similar generalization ability. In the depicted case, the PM-IM repeatedly switches between states for high-punches and low-punches leading, over time, to oscillations with an increasing amplitude.

An interesting property of PPCA-IM is the fact that it automatically produces continuous outputs in every time step. The controlled agent reacts even to small changes in the behavior of the opponent agent. By nature the PM-IM is a discrete model and does not produce different outputs in every time step. Still, a continuous output can be generated by using interpolation (as done in the above examples) or by incorporating velocity information into the model.

D. Robot Experiments

In order to validate our results on a real robot, we conducted an experiment in which a NAO robot learned a set of interaction skills that can be performed cooperatively or competitively with a human partner.

However, in order to replay any of the synthesized movements with the used NAO robot, we first have to find a mapping between the body parts of the human demonstrator and the body parts of the robot. This problem is commonly referred to as the *correspondence problem* [8] and is a fundamental problem of imitation learning. In this paper, we performed the mapping by using inverse kinematics (IK) on the extremities of the robot. The human skeleton was scaled to the size of the robot and IK was used to ensure that the positions of the feet, hands, pelvis and head of the robot matched the positions of the human extremities. More specifically, we used the iTaSC [10] IK algorithm for fitting the human skeleton to the robot. We have released the software package for IK-based correspondence matching of the NAO robot as an open-source tool for the general public[3].

To test our algorithms, we trained interaction models for the martial arts data set. The robot learns to recognize and

[3]The software can be downloaded as a *Blender*-extension from https://bitbucket.org/JuvenileFlippancy/naoblender

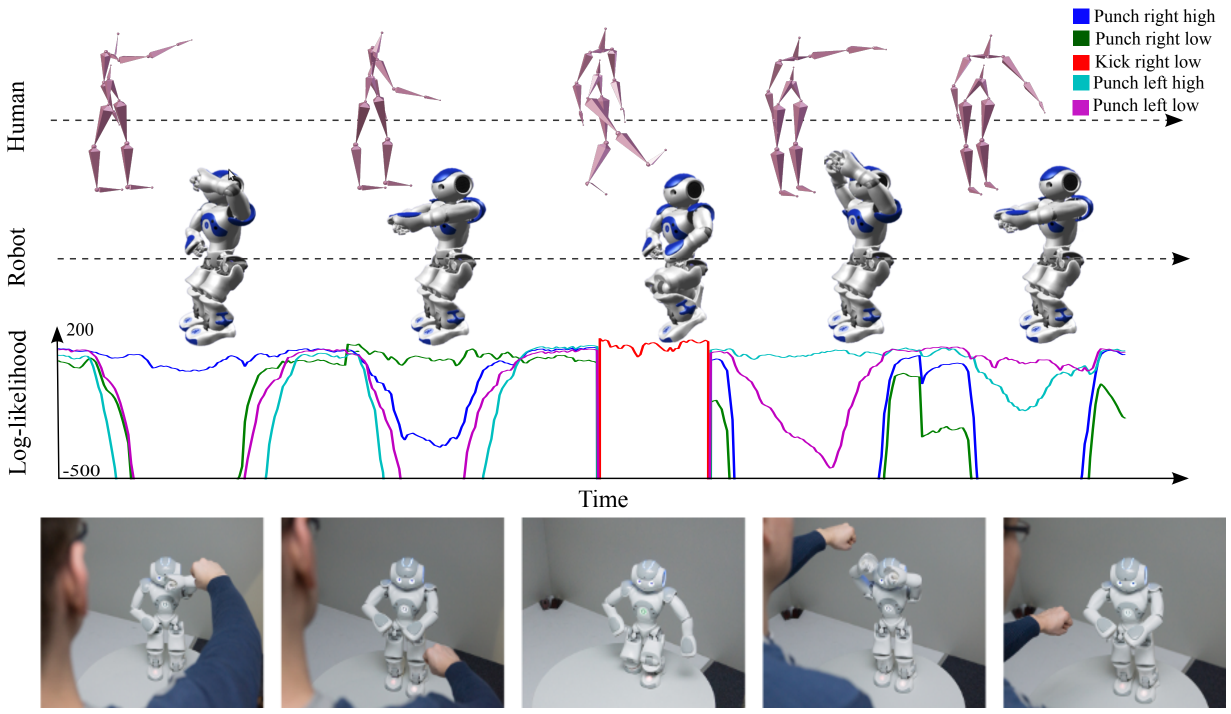

Fig. 8. A martial arts scenario trained and executed with the PM-IM algorithm. Top: The captured movement of the human. Second row: The joint angle configurations generated by the PM-IM after observing the human movement. Third row: The log-likelihoods for the different behaviors. Below: Pictures of the interaction between the human and the robot. The human movement was recorded with a Kinect camera.

defend different types of attacks, e.g., punchrighthigh, punchleftlow, kickrightlow. A set of 12 behaviors was used for training. Depending on the type of attack a different defense behavior is executed. For learning the PM-IM we used 60 to 70 hidden states. The number of hidden states for each behavior was estimated using cross-validation on the training data. For PPCA-IM we used a sampling rate of $10Hz$ and $\tau = 20$.

Figure 8 shows the movements of the human and the responses of the NAO robot. Note that the defense posture for an attack with the right hand and attacks with the left hand are different: robot lifts only one arm or both arms for defense. Similarly, the defense stance for a low-kick requires the robot to kneel down and block with one arm. The Figure also shows the log-likelihoods for the different behaviors that are generated by the HMMs. Interestingly the difference in the log-likelihood is very high when the opponent agent executes a kick-right-low movement. The robot can easily disambiguate this case as it is only when of two trained behaviors which use the leg. Both the PM-IM as well as the PPCA-IM model can solve the above task and produce appropriate responses for the NAO robot. The PPCA-IM model was again trained with 7 latent dimensions. Apart from martial arts examples we have also trained interaction models for the other data sets. Figure 1 shows the behavior of the robot after training a handing-over interaction task.

E. Discussion

The results suggest that both PPCA-IM and PM-IM can be used encode and reproduce the joint dynamics of the interaction partners in a shared task. However, the results also show various advantages and shortcomings of these algorithms. The PPCA-IM approach is particularly well suited for modelling continuous reponses and correlations in the movements of the interaction partners. It has limited computational demands and generalizes to some extent to new situations. This shows that dimensionality reduction can be an effective measure for extracting the hidden structure in interaction data. Without dimensionality reduction, the use of the temporal embedding for motion capture data would be computationally prohibitive and the learning would require a significantly larger amount of data.

At the same time, PPCA-IM does not provide information about the state or the development of the interaction. In this regard, the HMM-based PM-IM algorithm provides a richer set of tools for recognizing and estimating of the current state of the interaction. Yet, this comes at the price of significantly higher computational demands, as well as limited generalization abilities. Consequently, it would be interesting to combine the approaches presented in this paper by using a HMM with PPCA-IM models as emissions. In such a case, the HMM can be used to realize a space-time linearization of the training data, while the PPCA-IM can take on the role of modelling the correlations in the movements of the interaction partners in a particular temporal context. Recent advances in HMM training [23] also suggest a closer relationship between dimensionality reduction and temporal models such as HMMs.

V. CONCLUSIONS

In this paper we presented a new approach for teaching robots how to respond to the movements of a human partner. Using motion capture technology, the movements of a pair of persons are first recorded, and then processed using machine learning algorithms. The result is a model of how each person adapted its behavior to the movements of the respective other. Once an interaction model is learned, it can be used by a robot to engage in a similar task with a human counter part. We have also provided two algorithms called PPCA-IM and PM-IM, that are extensions to known methods, which can be used for learning interaction models from motion capture data. The algorithms allow a robot to learn *when* and *how* to respond to the behavior of a human partner. All methods were implemented on a NAO humanoid robot and were evaluated in cooperative and competitive tasks. After learning an interaction model, the NAO robot was able to generate appropriate defense responses in a challenging martial arts scenario. The discussion of the advantages and shortcomings of each of the two algorithms suggests that a combination of temporal models and dimensionality reduction can be an interesting path for developing more sophisticated models of interactions.

While the results in this paper are encouraging, there are various aspects of imitation learning in multi-agent scenarios that need further investigation. In particular, it is interesting to investigate how learned models can be used to predict the future behavior of an interaction partner given the actions of the controlled agents. This can be helpful in avoiding decisions that potentially lead to dangerous situations or injuries. In this paper, we did not investigate the aspect of force transfer between a human and a robot. Even small forces that are exchanged between interaction partners can have a significant impact on the execution and success of a joint task. First research results on incorporating force transfer in interaction models can be found in [1]. Another aspect that needs further investigation is task space control. For some interaction tasks it is important that constraints are fulfilled within the task space. We are currently investigating the use of Interaction Meshes [12] for this purpose. Finally, it is also important to include tertiary objects, e.g., a jointly lifted box, into the interaction model.

VI. ACKNOWLEDGMENT

The work presented in this paper is funded through the CoDyCo project of the European Community's Seventh Framework Programme under the grant agreement n ICT-600716 (CoDyCo).

REFERENCES

[1] E. Berger, H. Ben Amor, N. Haji-Ghassemi, D. Vogt, and B. Jung. Inferring guidance information in cooperative human-robot tasks. In *IEEE-RAS International Conference on Humanoid Robots (HUMANOIDS)*. IEEE, 2013 (submitted).

[2] F. Biessmann, F. C. Meinecke, A. Gretton, A. Rauch, G. Rainer, N. Logothetis, and K.-R. Müller. Temporal kernel canonical correlation analysis and its application in multimodal neuronal data analysis. *Machine Learning*, 79(1-2), 2009.

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

[4] D.P. Black, M.A. Riley, and C.K. McCord. Synergies in intra- and interpersonal interlimb rhythmic coordination. *Motor Control*, 11(4):348–73, 2007.

[5] C. Bouveyron, G. Celeux, and G. Stphane. Intrinsic dimension estimation by maximum likelihood in isotropic probabilistic {PCA}. *Pattern Recognition Letters*, 32(14):1706 – 1713, 2011.

[6] M. Brand and A. Hertzmann. Style machines. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '00, pages 183–192, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[7] 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.

[8] K. Dautenhahn and C. L. Nehaniv. *Imitation in Animals and Artifacts*. MIT Press, Campridge, 2002.

[9] A. d'Avella, P. Saltiel, and E. Bizzi. Combinations of muscle synergies in the construction of a natural motor behavior. *Nat Neurosci*, 6(3):300–8, 2003.

[10] J. De Schutter, T. De Laet, J. Rutgeerts, W. Decré, R. Smits, E. Aertbeliën, K. Claes, and H. Bruyninckx. Constraint-based task specification and estimation for sensor-based robot systems in the presence of geometric uncertainty. *Int. J. Rob. Res.*, 26(5):433–455, May 2007.

[11] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

[12] E. Ho, T. Komura, and C. Tai. Spatial relationship preserving character motion adaptation. *ACM Transactions on Graphics*, 29(4):1–8, 2010.

[13] A. Ijspeert, J. Nakanishi, and S. Schaal. Learning attractor landscapes for learning motor primitives. In Suzanna Becker, Sebastian Thrun, and Klaus Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 1523–1530. MIT Press, 2002.

[14] 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.

[15] M. Ito and J. Tani. On-line imitative interaction with a humanoid robot using a dynamic neural network model of a mirror system. *Adaptive Behavior*, 12(2):93–115, 2004.

[16] Z. Kolter, P. Abbeel, and A. Ng. Hierarchical Apprenticeship Learning with Application to Quadruped Locomotion. In John C. Platt, Daphne Koller, Yoram Singer, and Sam T. Roweis, editors, *Advances in Neural Information Processing Systems (NIPS)*. MIT Press, 2007.

[17] 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.

[18] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In L. M. Le Cam and J. Neyman, editors, *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press, 1967.

[19] J. Porta, J. Verbeek, and B. Krose. Active appearance-based robot localization using stereo vision. *Autonomous Robots*, 18(1):59–80, 2005.

[20] L. Rabiner. A tutorial on HMM and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, February 1989.

[21] M. Santello, M. Flanders, and J. F. Soechting. Postural Hand Synergies for Tool Use. *The Journal of Neuroscience*, 18(23):10105–10115, December 1998.

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

[23] L. Song, B. Boots, S. M. Siddiqi, G. J. Gordon, and A. J. Smola. Hilbert space embeddings of hidden Markov models. In *Proc. 27th Intl. Conf. on Machine Learning (ICML)*, 2010.

[24] 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.