Imitation Learning of Dual-Arm Manipulation Tasks in Humanoid Robots

Tamim Asfour, Florian Gyarfas, Pedram Azad and Rüdiger Dillmann
University of Karlsruhe
Institute for Computer Science and Engineering (CSE/IAIM)
P.O. Box 6980, D-76128 Karlsruhe, Germany
Email: {asfour,azad,dillmann}@ira.uka.de, florian.gyarfas@alumni.uni-karlsruhe.de

***Abstract*— In this paper, we deal with imitation learning of arm movements in humanoid robots. Hidden Markov Models (HMM) are used to generalize movements demonstrated to a robot multiple times. They are trained with the characteristic features (key points) of each demonstration. Using the same HMM, key points that are common to all demonstrations are identified; only those are considered when reproducing a movement. We also show how HMM can be used to detect temporal dependencies between both arms in dual-arm tasks. We created a model of the human upper body to simulate the reproduction of dual-arm movements and generate natural-looking joint configurations from tracked hand paths. Results are presented and discussed.**

I. Introduction

Humanoid robots are expected to exist and work together with human beings in everyday environments some day. In doing so they need to be able to interact and cooperate with humans. Interaction is facilitated if the robot behaves in a human-like way which implies that his movements look natural. This is not only advantageous for tasks involving direct physical cooperation between humans and robots; even if a robot acts independently his movements should appear familiar and predictable to us humans. In addition to that, it is probably also necessary for a robot to have a human-like appearance to be accepted by society. Given the dynamic character of the environment in which humanoid robots are expected to work, they need to have a high degree of flexibility. They need to be able to adapt to changes in the environment and to learn new tasks continuously, and they are expected to carry out a huge variety of different tasks. This distinguishes them from industrial robots which normally only need to perform a small number of rather primitive tasks in a static environment. It seems impossible to create a humanoid robot with built-in knowledge of all possible states and actions. Therefore, there has to be way of teaching the robot new tasks.

Teaching a robot can be done in a number of ways, for example by means of a robot programming language or a simulation-based graphical programming interface. Another method is “teaching by guiding”, where the instructor operates a robot manipulator while its motion is recorded. The recorded motions are then added to the robot’s action repertoire. Such techniques are well-suited for industrial robots; however, in the domain of humanoid robots, where robots are expected to cooperate with unexperienced users, they suffer from several drawbacks. They are lengthy, complex, inflexible, require special programming skills or costly hardware [1]. That contradicts the purpose of humanoid robots which is to make life easier for us. It is essential that the control of such robots will not be too difficult and time-consuming.

An approach that addresses both issues (human-like motion and easy teaching of new tasks) is *Imitation Learning*: It facilitates teaching a robot new tasks and at the same time make the robot move like a human. Imitation learning is basically the concept of having a robot observe a human instructor performing a task and imitating it when needed. Robot learning by imitation, also referred to as *programming by demonstration* has been dealt with in the literature as a promising way to teach humanoid robots and several imitation learning systems and architectures based on the perception and analysis of human demonstrations have been proposed [2]–[7]. In most architectures, the imitation process proceeds through three stages: perception/analysis, recognition and reproduction [8]. An overview of the basic ideas of imitation learning in robots as well as humans is given by Schaal in [9].

In this paper we focus on a simple form of imitation: A movement is demonstrated to a robot multiple times by a human instructor, subsequently generalized (using the data from all demonstrations) and finally reproduced by the robot without trying to infer the goal of the movement. As described in Section III, we have not yet reproduced movements on a robot but instead simulated the reproduction stage using a software model that we created for this purpose.

Our work was largely inspired by previous work of Calinon and Billard. In [5] and [10], they describe an approach to imitation learning that makes use of Hidden Markov Models (HMM) [11] to learn and reproduce movements demonstrated by a human instructor multiple times (while HMM have become very popular for the *recognition* of gestures or speech, they have not been widely used for the *reproduction* of movements). After the hand path and joint angle trajectories have been perceived, the data is reduced to a subset of critical features in a preprocessing stage using certain criteria (see [10]). Separate HMM (one for the hand path and one for each joint angle trajectory) are trained with those “key points”. The HMM are subsequently used to recognize further demonstrations of the same movement as well as to imitate the demonstrated task.

II. OUR APPROACH

We implemented a similar approach as in [10] for dual-arm movements that also uses HMM to imitate movements shown to a robot multiple times. In our method, however, the data is preprocessed in a slightly different way, and, more importantly, not all states of the resulting HMM are used to reproduce movements. Instead, we try to identify characteristic features that can be observed in all (or many) demonstrations and only consider those when reproducing a movement. We also show how those common features can be used to detect possible temporal interdependencies between both arms in dual-arm tasks. Moreover, we track the orientation of the hand and not just its position, since the correct orientation is essential for carrying out complex tasks. In contrast to [5], we do not, however, try to determine which features of a movement (hand path, joint angles etc.) are most relevant for the correct imitation of a task.

We use three different HMM for each arm, one to encode the position of the TCP (Tool Center Point, a reference point on the hand), i.e. the hand path, with the Cartesian coordinates being represented by three-dimensional output distributions, one for the orientation of the TCP (described by three angles) and another one for the joint angle trajectories where the dimension of the output distributions is equal to the number of observed joint angles (in our case, 7; see Section III). Those HMM are denoted by λ_p, λ_o and λ_j.

A. Preprocessing: Detection of characteristic features

A recorded movement is represented by a set of time-discrete sequences: Per arm there is one sequence $P_{d,1}, P_{d,2}, \ldots, P_{d,l(d)}$ that describes the positions of the TCP over time, one sequence $O_{d,1}, O_{d,2}, \ldots, O_{d,l(d)}$ that describes the orientations of the TCP, and seven more sequences $\theta^j_{d,1}, \theta^j_{d,2}, \ldots, \theta^j_{d,l(d)}$ that each specify the joint angle trajectory of joint j ($l(d)$ denotes the length of demonstration d). We omit the demonstration index d whenever it is not needed to distinguish between demonstrations. P_i and O_i are three-dimensional vectors. As in [10], we detect characteristic features of the perceived movement - key points - in a preprocessing stage and only use those to train the HMM. In doing so, we avoid having a high number of states and facilitate the matching of (or between) multiple demonstrations (see II-C). Those features are a subset of the aforementioned observation sequences. Since we try to detect distinctive features, it should still be possible to reconstruct the original movement well.

We detect key points separately for the position, orientation and joint angles of each arm. We use different criteria than [10] to identify key points, described as follows:

For joint angles, we use the following criterion: Let $\tau_{d,m}$ denote the time stamp of the m-th joint angle key point of demonstration d, i.e. its position in the sequence $\theta^j_{d,1}, \theta^j_{d,2}, \ldots, \theta^j_{d,l(d)}$. Then $\theta_{d,i} = (\theta^1_{d,i}, \ldots, \theta^J_{d,i})$, where J denotes the number of joints, is the m-th key point of the joint angles ($jK_{d,m}$, or just $K_{d,m}$ if the context is clear), $\tau_{d,m} = i$, if and only if

$$\begin{aligned} & \exists j: \quad \dot{\theta}^j_{d,i} = 0, i - \tau_{d,m-1} > \epsilon_1, |\theta^j_{d,i} - \theta^j_{d,\tau_{d,m-1}}| > \epsilon_2 \\ \vee \quad & \exists j: \quad |\theta^j_{d,i} - \theta^j_{d,\tau_{d,m-1}}| \geq \epsilon_3, i - \tau_{d,m-1} > \epsilon_4, \\ & \qquad |\theta^j_{d,n} - \theta^j_{d,\tau_{d,m-1}}| < \epsilon_3 \ \forall n \in [i, \tau_{d,m-1}) \end{aligned}$$

That means that whenever for any j, $\theta^j_{d,i}$ reaches an extremum (i.e. changes direction) or stops changing, sufficient time (ϵ_1) has passed since the creation of the previous key point and the joint angle at i differs from the angle at $\tau_{d,m-1}$ by at least ϵ_2, a key point $K_{d,l}$ is created. The second part of the condition ensures that when an angle that has stayed the same for some amount of time (ϵ_4) starts changing again, a key point is created as well. Figure 1 shows three different candidates for key points that given appropriate thresholds could all satisfy the above criterion. Any point between θ_b and θ_c, however, would not be a key point.

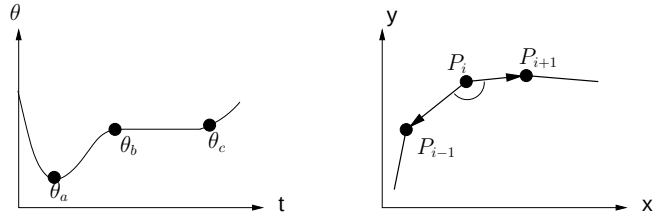

Fig. 1. Potential key points (left) and key point criterion (right)

For the orientation angles of the TCP (key points oK) the same criterion is used, but with possibly different threshold values ($\epsilon_5 - \epsilon_8$).

For TCP positions, we use a slightly different criterion. A point $P_{d,i}$ is a key point $pK_{d,m}$ (or just $K_{d,m}$) in the sequence of points $P_{d,1}, \ldots, P_{d,l(d)}$ if and only if

$$\begin{aligned} & \angle(\vec{P_{d,i}} - \vec{P_{d,i-1}}, \vec{P_{d,i}} - \vec{P_{d,i+1}}) < 180^\circ - \epsilon_9 \\ \vee \quad & ||\vec{P_{d,i}} - \vec{P_{d,i-1}}|| < \epsilon_{10}, i - \tau_{d,m-1} > \epsilon_{11}, \\ & ||\vec{P_i} - \vec{P_{d,\tau_{d,m-1}}}|| > \epsilon_{12} \\ \vee \quad & ||\vec{P_{d,i}} - \vec{P_{d,\tau_{d,m-1}}}|| \geq \epsilon_{13}, i - \tau_{d,m-1} > \epsilon_{14}, \\ & ||\vec{P_{d,n}} - \vec{P_{d,\tau_{d,m-1}}}|| < \epsilon_{15} \ \forall n \in [\tau_{d,m-1}, i) \ , \end{aligned}$$

where $\tau_{d,m}$ denotes the time stamp of the m-th position key point of demonstration d.

So if the angle between the vector that goes from P_i to its predecessor P_{i-1} and the vector that goes from P_i to its successor P_{i+1} is less than $180^\circ - \epsilon_9$ (see Fig. 1), P_i is considered a key point. In practice you would want ϵ_9 to be fairly high so that only sharp corners in the hand path would be detected as key points. Also, as for the joint angle trajectories, if the position remains unchanged for some time or starts changing again, a key point is created as well. Reasonable values for $\epsilon_1, \ldots \epsilon_{15}$ can be determined experimentally.

In [10] hand path key points are created only when there is a change in X, Y or Z direction. However, there can be significant changes in direction in a 3D path that do not necessarily result in a reversed direction of any single

coordinate. Our criterion for the hand path has the advantage that such key points are also detected because it takes the angle into account.

B. HMM structure

As mentioned before, we use multiple HMM for different kinds of observations (joint angles, TCP position, TCP orientation). Each HMM is trained with the key points of all demonstrations using the Baum-Welch algorithm for multiple observations ([12]). Each training sequence consists of the key points of the respective demonstration.

For a given observation sequence, the Viterbi algorithm (see [11]) returns the optimal state sequence of an HMM with respect to that observation sequence, i.e. the sequence of states most likely to generate that observation sequence. Therefore, if used on an HMM with the key points of one of the demonstrations as the observation sequence, the Viterbi algorithm would yield a sequence of states that could be said to correspond (in a probabilistic way) to the key points of the observed movements. Of course, the key point corresponding to a state might not occur in all demonstrations; a state could very well represent a key point of only one demonstration.

Since the states of the HMM roughly correspond to the key points, it seems reasonable to set the number of states equal to the number of key points. However, we use multiple demonstrations to train the various HMM and each demonstration might have different key points; therefore, the HMM should have as many states as there are distinct key points in all demonstrations. We do not know in advance, though, which of the key points of different demonstrations are equivalent, so we have to use an estimate for the number of states. In our experiments, we set the number of states to $\max_d k(d) \cdot 2$, where $k(d)$ denotes the number of key points in demonstration d.

This is of course not a very satisfying solution and we are still working on this problem. The standard HMM architecture that we use does not appear to be perfectly suited for our approach. What we would like to have is a model that had a certain number of main states for important key points that appear in most demonstrations and that allowed for inserting as many states as necessary in between those main states for key points that appear in only one or only a few demonstrations. It should also allow for skipping states that might be key points in most but not all demonstrations. Profile HMM, a special kind of HMM which are commonly used in bioinformatics to align DNA sequences, have those properties [13]. However, they use discrete output variables and we have yet to determine how such an architecture could be used with continuous variables and in the context of our method.

The HMM we use are left-right models [11], i.e. there are no state transitions between two states S_i and S_j if $j <= i$. That is because we want the models to reflect the sequence of key points over time and thus it should not be allowed to go backwards in time.

We use continuous HMM, with the output distribution in each state representing whatever is encoded in the HMM (for example, the TCP position or the angle of a specific joint) at a certain time given by the time stamp of the corresponding key point(s) (if multiple key points correspond to a state we compute the average of their time stamps and associate that average time stamp with the state).

The output probability is modeled by the density function of a multivariate Gaussian distribution. We do not use mixtures of such density functions - this is crucial if we want to use the HMM to reproduce a generalized movement because we consider the means of those functions to be the generalized positions, orientation angles and joint angles (as explained also in the next two paragraphs); as such, they have to be scalar values.

Before the training takes place, the HMM are initialized as follows: The initial state probabilities π_i are set to equal values that sum to 1. They are not particularly important since they get changed quickly by the Baum-Welch algorithm. We set the initial transition probabilities a_{ij} in such a way that a transition from a state S_i to a state S_j is most likely for $j = i + 1$ and less likely the higher $j - i$ gets. For the sake of simplicity and to make the models more robust, our covariance matrices cov_{ij} of the multivariate Gaussian density functions are diagonal matrices, which means $\text{cov}_{ij} = 0$ for $i \neq j$. The variances (cov_{ii}) are initialized with high values. The means μ_i are all initially set to 0 (unlike in [10] we do not initialize them with the key point values because we do not know in advance which state corresponds to which key point).

C. Common key points

As described earlier, the states of the Hidden Markov Models represent key points of the demonstrations. A key point may, however, only occur in some of the demonstrations. Thus, the question arises which states of the HMM should be used for the reproduction. In [10], the reproduction of a movement is triggered by another demonstration of the same movement. The Viterbi algorithm is then used to determine which states of the HMM match the key points of that demonstration most closely; those states are then used for the reproduction.

In our approach, however, the reproduction of a movement is not necessarily triggered by another demonstration of that same movement. We use only those key points for the reproduction of a movement that are common to all (or almost all) demonstrations and call them “common key points”. The reason for that is that a key point of a *single* demonstration is not necessarily a characteristic feature of the movement, in which case it would be reasonable not to consider it for the reproduction. But how do we know what key point in one demonstration corresponds to some key point in another demonstration? This is a classic matching problem which could for instance be solved by DP matching. However, we actually use our HMM to match key points across demonstrations. We use only those states of the HMM for the reproduction that represent key points which are shared by all (or almost all) demonstrations.

Formal definition: Let D denote the number of demonstrations. A common key point is defined as a 3-tuple:

$$C_j = ((i_{j,1}, \ldots, i_{j,D}), T_j, \nu_j),$$

where $(i_{j,1}, \ldots, i_{j,D})$ denotes the indices of the corresponding key points of the different demonstrations. The D-tupel $T_j = (\tau_{1,i_{j,1}}, \ldots, \tau_{D,i_{j,D}})$ contains the time stamps of those key points, while the n-tuple $\nu_j = (\nu_j^1, \ldots, \nu_j^n)$ describes the values of the trajectory to be reproduced (i.e. joint angles where n=number of joints, Cartesian coordinates where n=3 or orientation angles where n=3 as well) at that common key point. It should be noted that those values are not equal to the values of any key points; they are the means of the HMM's output density function of the state that corresponds to the key points $(K_{1,i_{j,1}}, \ldots, K_{D,i_{j,D}})$.

Detection of common key points: For each demonstration we have a list of key points: $K_{d,1}, \ldots, K_{d,k(d)}$, where d denotes the number of the demonstration, and $k(d)$ stands for the number of key points. By using the Viterbi algorithm on the HMM for each such sequence of key points, we get the sequences of states that correspond best to those key points (more precisely, the sequences of states that would most likely output the key points). Let $S_{d,j}$ denote the state that corresponds to the j-th key point of the d-th demonstration. Common key points can then be detected as follows:

$$\begin{aligned} &K_{1,i_{j,1}}, K_{2,i_{j,2}}, \ldots, K_{D,i_{j,D}} \text{ form a common key point} \\ &C_j = ((i_{j,1}, \ldots, i_{j,D}), T_j, \nu_j) \\ &\Leftrightarrow S_{1,i_{j,1}} = S_{2,i_{j,2}} = \ldots = S_{D,i_{j,D}} \end{aligned}$$

So only those key points of one demonstration that correspond to a state which also corresponds to key points in all other demonstrations are considered common key points. In figure 2 the bold circles are the optimal state sequence returned by the Viterbi algorithm for the key points of each demonstration. For example, S_4 corresponds to the key points $K_{1,4}$, $K_{2,3}$ and $K_{3,2}$. Clearly, only S_1, S_4 and S_5 represent common key points here.

In general, some of the demonstrations are likely to be erroneous, so it seems reasonable to ease the above constraint by saying that it suffices if the majority (say, 4/5) of the demonstrations have a key point that corresponds to the same state.

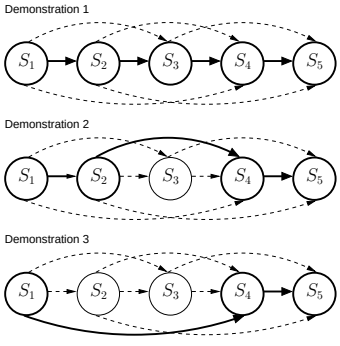

Fig. 2. Identifying common key points - S_1, S_4 and S_5 are the only states representing common key points

Generating a generalized movement with common key points: We can reproduce a generalized movement that is basically the average of all demonstrations and - more importantly - only uses the common key points of all demonstrations, as follows: We consider only those states of the HMM that correspond to common key points; we then take the means of the output density functions (PDFs) of those states (in order) and what we get is a sequence of positions, orientations and joint angles that can be used to reproduce the movement. However, two issues remain to be resolved before the movement can be regenerated from the common key points. First, what time stamps should be associated with each of those common key points? One possibility is to simply take the average of the time stamps T_j of all the key points that constitute a common key point. Second, the positions and orientations are of no use by themselves if we want to reproduce a movement on a robot or a software model of the human body; they have to be transformed to joint angles. That can be done with an inverse kinematics algorithm.

D. Reproduction

The detection of common key points is performed separately for each of the six HMM. Thus, we get sequences of points that are not necessarily in lockstep; they do not have to occur simultaneously. We interpolate between common key points to solve that problem. For example, for each common joint angle key point, we interpolate between the common position and orientation key points closest in time to determine the TCP position and orientation at the same time. That way we obtain a sequence of common key points of which each common key point has values for TCP position, TCP orientation and all joint angles of both arms.

Since the common key points only describe characteristic features of the trajectories and not the whole trajectories, we also have to interpolate between them to obtain actual trajectories that can be used to imitate a movement. Both spline and linear interpolation can be used for that purpose (we have only implemented linear interpolation so far).

So eventually we obtain time-discrete sequences for the TCP position and orientation as well as for all joint angle trajectories that each describe the movement at every point in time (using a certain sampling rate). The sequences of TCP positions and orientation angles are then transformed to joint angles with an inverse kinematics algorithm, so that they can be used for the reproduction.

The resulting joint angle sequences $\hat{\theta}_i^j$ are different from the joint angle sequences θ_i^j that are obtained directly from the joint angle HMM. A weighting factor $\omega \in [0, 1]$ is used to determine the relative influence of $\hat{\theta}_i^j$ and θ_i^j on the joint angles $\tilde{\theta}_i^j$ that are used to generate the final motion to be executed by the robot:

$$\tilde{\theta}_i^j = \omega * \theta_i^j + (1 - \omega) * \hat{\theta}_i^j$$

This weighting factor can be set by the user. Since the joints of the robot are not usually totally equivalent to the joints of the human demonstrator, the human joints have to be mapped to the robot's joints. This can be done in different ways: One can try to approximate the joint angle trajectories of the demonstrated movement as closely as possible, making the imitation look natural. That way the hand path might not be reproduced very well, though. Alternatively, one can try to imitate the hand path as accurately as possible using joint angles determined by an inverse kinematics algorithm which might deviate strongly from the joint angle trajectories perceived in the demonstration. The former is achieved by using a weighting factor of $\omega = 1$ while for the latter one would set ω to 0.

The whole process from perception to reproduction is depicted in figure 3. As in [10], the HMM can of course also be used to recognize (classify) movements. That has not been the focus of our work, though.

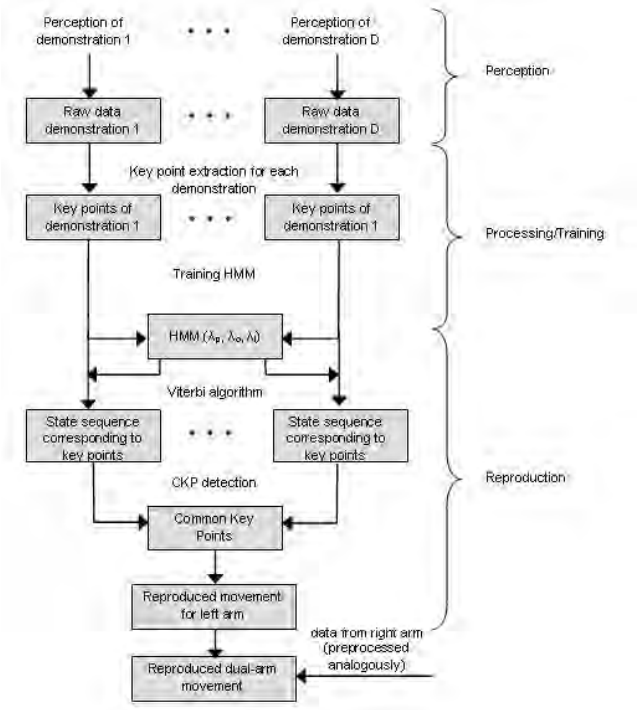

Fig. 3. Overview on the entire approach from perception to reproduction

E. Temporal coordination

When imitating dual-arm movements, often certain constraints have to be satisfied. Dual-arm movements can be uncoordinated or coordinated. No matter what method is chosen for the reproduction, information about coordination between the two arms can be very useful. There are various kinds of coordination; one way to classify two-arm coordination is to distinguish between temporal and spatial coordination. Spatial coordination means that the position and orientation of one hand is at least partially determined by the position and orientation of the other hand. As opposed to that, temporal coordination means that one arm must accomplish a certain subgoal as moving the hand to a specific position before the other arm may continue executing its movement. When imitating a two-arm movement, such dependencies should be detected and reproduced properly. For example, if with your left hand you pour water from a bottle into a glass that you hold in your right hand, you have to move the glass to its correct position before you can start pouring. It would be helpful if one could find out whether an observed temporal relation is just coincidence or whether it constitutes a necessary coordination. Multiple demonstrations allow you to determine the likelihood of some temporal relation being a true coordination.

The HMM approach in conjunction with the common key points introduced in this Section can be used to detect temporal relations between characteristic points of the left hand path and the right hand path that are unlikely to be coincidental.

Let $(\tau_{1,i_{r,1}}, \ \ldots, \ \tau_{D,i_{r,D}})$ be the time stamps of the key points that form a common key point C_r of the right arm's movement and $(\tau_{1,i_{l,1}}, \ \ldots, \ \tau_{D,i_{l,D}})$ be the time stamps of some C_l of the left arm. Then we can conclude that most likely there exists temporal coordination between the point C_r and C_l if in all demonstrations C_r is reached before C_l - or the other way round, i.e., if:

$$\forall k : \left(\tau_{k,i_{r,k}} < \tau_{k,i_{l,k}} \vee \tau_{k,i_{r,k}} > \tau_{k,i_{l,k}}\right)$$

It could also be possible that both arms must reach some points at the same time. That is accounted for by the following additional criterion for temporal coordination (δ is fixed and should be chosen close to zero):

$$\forall k : |\tau_{k,i_{r,k}} - \tau_{k,i_{l,k}}| < \delta$$

Once again, detecting those temporal relations only requires knowledge about common key points which may be acquired without HMM and may thus be implemented independently from the rest of our approach.

III. Human Kinematics Model

The goal of the methods described in this paper is to eventually have robots imitate human arm movements. To achieve this, we need to be able to perceive and analyze demonstrations, transform them to a robot joint representation and reproduce them. To begin with, we have so far only simulate the reproduction on a kinematic model of the human upper body with 18 degrees of freedom (Fig. 4), where each arm is modeled by nine DOFs: two in the inner shoulder joint (sternoclavicluar), three in the outer shoulder joint (glenohumeral), two at the elbow and two at the wrist ([14]).

To make imitation look natural, joint angles must be preserved as well as possible. However, measuring the joint angles of a human demonstration requires a complex tracking system (for example a marker-based system) which is not very usable in everyday life. So instead, we have incorporated two different

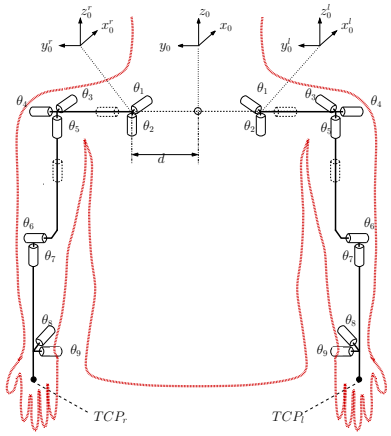

Fig. 4. The kinematics model of both arms

methods for the acquisition of the required data: a method based on a magnetic tracking system, and a purely vision-based markerless human motion capture system.

Using an easy to set up magnetic tracking system (Fastrak, `www.polhemus.com`), the position and orientation of both hands can be tracked. The joint angles can then be reconstructed using an approach based on neurophsychological studies. In [15] and [16], Soechting and Flanders have shown that arm movements are planned in shoulder-centered spherical coordinates and suggest a sensorimotor transformation model that maps the Cartesian wrist position to a natural arm posture using a set of representation parameters, which are the upperarm elevation, the forearm elevation, the upperarm yaw and the forearm yaw, respectively.

In neurophysiology evidence exists that arm and hand postures are independent of each other. This means that one can find the forearm and upper arm posture to match the hand position and then determine the joint angles for the wrist to match the hand orientation. In [17] we proposed a new algorithm which incorporates the physiological observation into a closed-form solution of the inverse kinematics problem to generate natural looking arm postures. For the reconstruction of the arm joint angles, we geometrically derived equations for the arm joint angles $\theta_3, \ldots, \theta_9$ in a closed form. The remaining two shoulder joints θ_1 and θ_2 supported by the arm kinematics model can be set manually by the user. For more details the reader is referred to [18].

The approach described in section II is based on the tracked hand path as well as the seven reconstructed joint angles.

Recently, we have also developed a purely image-based markerless human motion capture system [19], [20]. The input of the system are stereo color images of size 320×240 captured at 25 Hz, with two calibrated Dragonfly cameras built-in into the head of the humanoid robot ARMAR III. The input images are preprocessed, generating output for the gradient cue, the distance cue, and an optional region cue, as described in [20]. Based on the output of the image processing pipeline, a particle filter is used for tracking the movements in configuration space. The overall likelihood function to

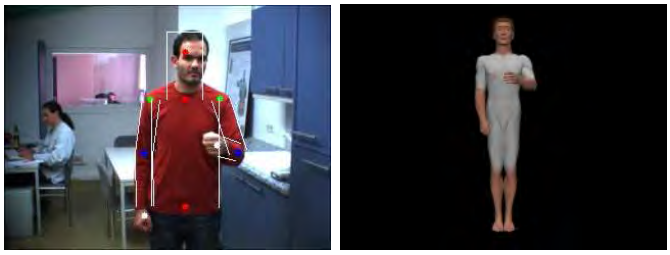

Fig. 5. Illustration of the performance of the markerless human motion capture system. Left: projection of the estimated configuration into the left camera image. Right: 3D visualization of the estimated configuration with an articulated human model.

compute the a-posteriori probabilities is formulated as:

$$p(\mathbf{z}|\mathbf{s}) \propto \exp\left\{-\frac{1}{2}\left(\frac{1}{\sigma_d^2}\sum_{i=1}^{3} d_i(\mathbf{s},\mathbf{c}_i) + \frac{1}{\sigma_g^2}(1 - \frac{1}{M_g}\sum_{m=1}^{M_g} g_m)\right)\right\},$$

where $\mathbf{s}$ is the configuration to be evaluated, $\mathbf{z}$ is a general denotation for the current observations i.e. the current input image pair, and $\mathbf{c}_i \in \mathbb{R}^3$ with $i \in \{1,2,3\}$ denotes the triangulated 3D position of the hands and the head. The function $d_i(\mathbf{s},\mathbf{c})$ is defined as:

$$d_i(\mathbf{s},\mathbf{c}) := \begin{cases} |f_i(\mathbf{s}) - \mathbf{c}|^2 & : \quad \mathbf{c} \neq \mathbf{0} \\ 0 & : \quad \text{otherwise} \end{cases},$$

where $n := \dim(\mathbf{s})$ is the number of DOF of the human model. The transformation $f_i : R^n \rightarrow R^3$ transforms the n-dimensional configuration of the human model into the 3D position of the left hand, right hand or head respectively, using the forward kinematics of the human model. The g_m with $m \in \{1,2,...,M_g\}$ denote the intensity values in the gradient image (which is derived from the input images $\mathbf{z}$) at the M_g pixel coordinates of the projected contour of the human model for a given configuration $\mathbf{s}$. This process is performed for both input images using the calibration parameters of each camera. For each image pair of the input sequence the output of the system is the estimation of the particle filter, given by the weighted mean over all particles. A detailed description is given in [20].

In contrast to the acquisition method based on the magnetic tracking system, the joint angle values θ_3, θ_4, θ_5, and θ_6 are calculated *directly* and therefore the position of the elbow does not have to be approximated based on empirical studies but is determined explicitly.

IV. Experiments

We used the kinematic model described in the previous section to conduct several experiments. We tested the preprocessing, generalization and reproduction stages of our approach with three different dual-arm movements:

- pick-and-place task: we picked up a box with both arms, moved it to the right and put it down again
- pouring motion: we poured water from a bottle held by the left hand into a glass that was held by the right hand

- unscrewing motion: we unscrewed the lid of a jar (a complex motion that is different every time it is demonstrated; this did not work too well with our system)

The joint angles were obtained from the position of the hand using the algorithm based on findings in [15] and [16] as described in the previous Section. Our kinematic model was clearly capable of computing natural looking joint angles that way. For our experiments, we used the model to simulate the imitation of movements. We were able to display the simulated movement using a visualization of the human upper body that we created with OpenInventor and combined with our kinematics model. We implemented our own HMM for this system and did not make use of existing HMM toolbox.

Each of the movements mentioned above was demonstrated between five and ten times by the same person. Select training samples as well as the reproduced (generalized) hand path for the pick-and-place task are shown in Figure 6. It can be seen how the generalized trajectory is interpolated linearly between the common key points that were found using all demonstrations.

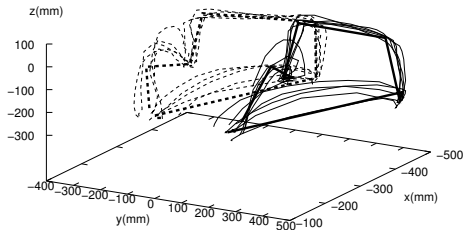

Fig. 6. Pick and place task: Training samples and the generalized trajectory for $\omega = 0$

Figure 7 shows only the reproduced trajectory of the left hand, but for three different ω. As expected, if the reproduction is based on joint angles and not only on the observed hand path ($\omega = 1$ or $\omega = 0.5$), the hand path becomes somewhat erratic. On the other hand, however, the reproduced movement seems more realistic and less artificial in that case which can be explained by the fact that more key points are created for joint angle trajectories than for the hand path (a joint angle key point is generated each time *any* joint changes its direction). If you used spline interpolation between key points instead of linear interpolation, you would probably obtain more realistic looking trajectories for $\omega = 0$ as well.

In Figure 8, the original joint angle trajectory of joint θ_7 (left arm, see also Figure 4) of one randomly selected demonstration (solid line) as well the generalized trajectory of that joint for $\omega = 0$ (dotted line) and $\omega = 1$ (dashed line) are shown. For $\omega = 0$ the spike at about $t = 20s$ is ignored because obviously no key point is detected in the hand path at that point in time. The key points detected in one of the demonstrations are shown in Figure 9 which seems to be what one would expect (characteristic features, i.e. significant changes in direction, have been detected).

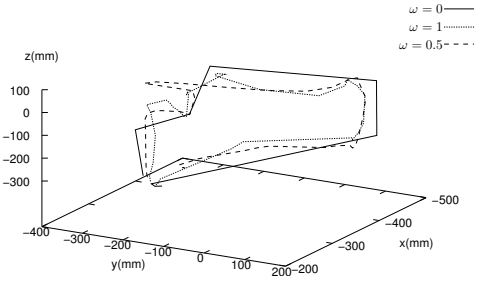

Fig. 7. Generalized trajectory of the left hand for different values of ω

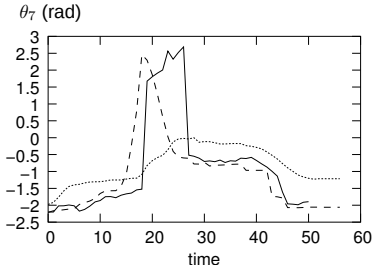

Fig. 8. The generated joint angle trajectory for joint θ_7 of the left arm

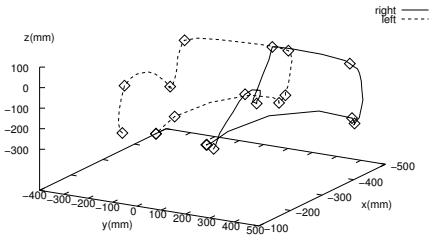

Fig. 9. All detected key points in the pick-and-place demonstration

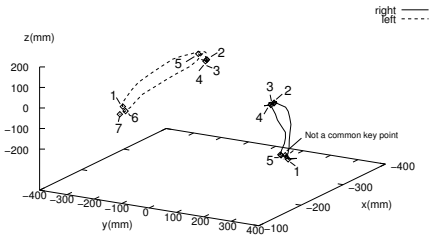

Fig. 10. All detected key points in the pouring motion

Figure 10 shows the key points detected in a training sample of the pouring motion. One of the key points is not a common key point, though, and will thus not be considered for the imitation of the movement.

Table I shows the temporal relations that were identified between the common key points of both arms. The numbers in the precondition column specify for each arm which common key point the other arm's hand must reach before the motion can be continued. Before the left hand is allowed to proceed to the third common key point, the right hand has to reach its second common key point, which is what one would expect.

CKP (right)	pre-condition (left)	CKP (left)	pre-condition (right)
1		1	1
2	1	2	
3		3	2
4	3	4	
5	6	5	
		6	
		7	

TABLE I

TEMPORAL RELATIONS BETWEEN COMMON KEY POINTS OF THE ARMS

Of course, a few other relations that might not be critical for the correct execution of the task were detected as well.

V. DISCUSSION, CONCLUSIONS AND FUTURE WORK

We presented an HMM-based approach for imitation learning of arm movements in humanoid robots. HMM are used to generalize movements demonstrated to a robot multiple times. Common key points in all demonstrations are identified and used for the reproduction of the movements. The results we obtained using a software model specifically created for this purpose show that our approach is an encouraging step in the effort to teach a robot new tasks in a flexible and natural way.

We would like to mention once again the work of Calinon and Billard in [5] and [10] due to the similarity to parts of our work. However, while our work was clearly inspired by their method, only the basic ideas are based on it. We have introduced new aspects such as the detection of common key points and identifying temporal coordination between two arms and use a different preprocessing method. The kinematic models presented in Section III are also part of the contribution of this paper.

So far, we have only simulated generating generalized trajectories, using a kinematic model of the human arms. A natural next step would be the reproduction of movements on the humanoid robot ARMAR-III. That would involve mapping the joints of our kinematic model to those of the robot and will be part of future work. Another aspect of our method that should be improved is how the trajectories are encoded in the HMM. As described in Section II, a generic HMM architecture does not seem to be ideal for our approach.

Also, our system is not invariant to translations or rotations yet, a highly desirable property for any imitation learning system to be used in practice.

Furthermore, it seems essential for the accurate imitation of tasks to take into account objects that are manipulated during the demonstration. Not only do we need to make sure that an imitated movement results in the same manipulation of an object as the demonstrated movement, but considering objects would also be a significant step towards recognizing the goal of a task, which we have not attempted to do yet but which is clearly an important aspect of imitation learning.

ACKNOWLEDGEMENT

The work described in this paper was partially conducted within the EU Cognitive Systems project PACO-PLUS (FP6-2004-IST-4-027657) and funded by the European Commission and the German Humanoid Research project (SFB 588) funded by the German Research Foundation (DFG: Deutsche Forschungsgemeinschaft).

REFERENCES

[1] Y. Kuniyoshi, M. Inaba, and H. Inoue, "Learning by watching: Extracting reusable task knowledge from visual observation of human performance," *IEEE Transactions on Robotics and Automation*, vol. 10, pp. 799–822, 1994.

[2] R. Dillmann, "Teaching and learning of robot tasks via observation of human performance," *Robotics and Autonomous Systems*, vol. 47, no. 2-3, pp. 109–116, 2004.

[3] A. Billard and R. Siegwart, "Robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 47, no. 2-3, pp. 65–67, 2004.

[4] A. Billard, Y. Epars, S. Calinon, S. Schaal, and G. Cheng, "Discovering optimal imitation strategies," *Robotics and autonomous systems*, vol. 47, pp. 69–77, 2004.

[5] S. Calinon, F. Guenter, and A. Billard, "Goal-directed imitation in a humanoid robot," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2005)*, 2005.

[6] C. G. Atkeson and S. Schaal, "Robot learning from demonstration," in *Machine Learning: Proceedings of the Fourteenth International Conference (ICML 1997)*, 1997, pp. 1040–1046.

[7] S. Schaal, A. Ijspeert, and A. Billard, "Computational approaches to motor learning by imitation," *Philosophical Transactions of the Royal Society of London: Series B, Biological Science*, vol. 358, no. 1431, pp. 537–547, 2003.

[8] Y. Demiris and G. Hayes, "Imitation as a dual-route process featuring predictive learning components: a biologically-plausible computational model," *Imitation in animals and artifacts*, pp. 327–361, 2002.

[9] S. Schaal, "Is imitation learning the route to humanoid robots?" *Trends in Cognitive Sciences*, vol. 3, no. 6, pp. 233–242, 1999.

[10] S. Calinon and A. Billard, "Stochastic gesture production and recognition model for a humanoid robot," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005)*, 2005, pp. 2769–2774.

[11] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," in *Proceedings of the IEEE*, vol. 77, 1989, pp. 257–286.

[12] J. Bilmes, "A gentle tutorial on the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models," University of California - Berkeley, Tech. Rep., 1997.

[13] S. R. Eddy, "Profile hidden markov models," *Bioinformatics*, vol. 14, pp. 755–763, 1998.

[14] T. Asfour, "Sensomotorische Bewegungskoordination zur Handlungsausführung eines humanoiden Roboters (german)," Ph.D. dissertation, University of Karlsruhe, Karlsruhe, Germany, 2003.

[15] J. F. Soechting and M. Flanders, "Sensorimotor Representations for Pointing to Targets in Three-Dimensional Space," *Journal of Neurophysiology*, vol. 62, no. 2, pp. 582–594, 1989.

[16] ——, "Errors in Pointing are Due to Approximations in Targets in Sensorimotor Transformations," *Journal of Neurophysiology*, vol. 62, no. 2, pp. 595–608, 1989.

[17] T. Asfour and R. Dillmann, "Human-like Motion of a Humanoid Robot Arm Based on Closed-Form Solution of the Inverse Kinematics Problem." in *The IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)*, Las Vegas, USA, 2003.

[18] F. Gyarfas, "Imitation Learning zur modellbasierten Generierung menschenähnlicher Zweiarm-Bewegungen," Master's thesis, University of Karlsruhe, Germany, 2005.

[19] P. Azad, A. Ude, R. Dillmann, and G. Cheng, "A Full Body Human Motion Capture System using Particle Filtering and On-The-Fly Edge Detection," in *International Conference on Humanoid Robots (Humanoids)*, Santa Monica, USA, 2004.

[20] P. Azad, A. Ude, T. Asfour, G. Cheng, and R. Dillmann, "Image-based Markerless 3D Human Motion Capture using Multiple Cues," in *International Workshop on Vision Based Human-Robot Interaction*, Palermo, Italy, 2006.