2009 IEEE International Conference on Robotics and Automation
Kobe International Conference Center
Kobe, Japan, May 12-17, 2009

Task-level Imitation Learning using Variance-based Movement Optimization

Manuel Mühlig, Michael Gienger, Sven Hellbach, Jochen J. Steil, Christian Goerick

***Abstract*— Recent advances in the field of humanoid robotics increase the complexity of the tasks that such robots can perform. This makes it increasingly difficult and inconvenient to program these tasks manually. Furthermore, humanoid robots, in contrast to industrial robots, should in the distant future behave within a social environment. Therefore, it must be possible to extend the robot's abilities in an easy and natural way. To address these requirements, this work investigates the topic of imitation learning of motor skills. The focus lies on providing a humanoid robot with the ability to learn new bi-manual tasks through the observation of object trajectories. For this, an imitation learning framework is presented, which allows the robot to learn the important elements of an observed movement task by application of probabilistic encoding with Gaussian Mixture Models. The learned information is used to initialize an attractor-based movement generation algorithm that optimizes the reproduced movement towards the fulfillment of additional criteria, such as collision avoidance. Experiments performed with the humanoid robot ASIMO show that the proposed system is suitable for transferring information from a human demonstrator to the robot. These results provide a good starting point for more complex and interactive learning tasks.**

I. INTRODUCTION

One of the manifold research topics of the *Honda Research Institute Europe* is the integration of biologically inspired learning methods into a humanoid robot. The robot shall be able to learn autonomously by interacting with its surrounding environment. Especially the learning of new movement skills is an interesting topic and this work investigates the use of an imitation learning paradigm to acquire those skills by observing a teacher.

The topic of imitation learning is very broad with respect to different levels of abstraction. While for example [1], [2] propose to use imitation learning to learn trajectory level information about a movement task, also higher-level approaches like [3] exist that try to learn complex tasks in form of graph structures of basic movements. Furthermore, hierarchical approaches such as [4], [5] combine aspects of several abstraction levels. In this work we focus on learning trajectory information and to introduce the problems that need to be solved, the learning of a common bi-manual task is taken as an example. The robot has to learn to pour a beverage from a bottle into a glass by observing a teacher demonstrating this task. This choice is arbitrary and the researched methods do not depend on this specific choice but are general. An overview of the whole imitation learning process is depicted in figure 1 and explained in detail within the upcoming sections.

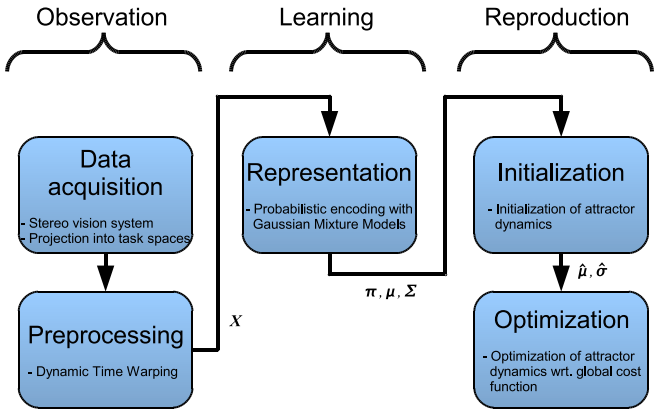

Fig. 1. Structure of the imitation learning process

The first step necessary is to acquire the movement information from the teacher. Several methods are applicable for this, such as kinesthetic teaching [6], recording human postures with marker-based [7], [8] or marker-less vision systems [9]. Within this work, the movement is acquired using a marker-less stereo vision system, which is described in section II. The teacher's posture is disregarded and learning as well as movement reproduction are based on object trajectories only.

As learning performance and generalization capabilities are major keypoints for interactive learning of movement tasks, it is unfavorable to learn within the full high-dimensional configuration space of the robot. A common approach to avoid this is the application of dimension reduction techniques, such as the principal component analysis [10]. However, such methods often project the observed movement information into rather abstract dimensions that do not necessarily improve the generalization capabilities. Hence, this work follows the concept of task spaces to model the movement. For the *pouring task* example, the task space can simply consist of the relative position of bottle and glass and their orientations. Section III describes that this does not only avoid the correspondence problem between the teacher's and robot's kinematic structure but also enhances generalization.

M. Mühlig and J. J. Steil are with the Research Institute for Cognition and Robotics, Bielefeld University, 33594 Bielefeld, Germany. {muehlig,jsteil}@cor-lab.uni-bielefeld.de

M. Mühlig, M. Gienger and C. Goerick are with the Honda Research Institute Europe, Carl-Legien-Strasse 30, 63073 Offenbach/Main, Germany. {michael.gienger,christian.goerick}@honda-ri.de

S. Hellbach is with the Neuroinformatics and Cognitive Robotics Lab, Technical University Ilmenau, 98693 Ilmenau, Germany. sven.hellbach@tu-ilmenau.de

After the *pouring task* was observed and modelled using task spaces, it is learned within a representation that allows the robot to adapt it to new situations. We believe that the important parts of a movement task are mostly defined by their invariance over several demonstrations and that this variance should be directly incorporated into the robot's movement generation. This allows the robot to diverge from variant and therefore less important parts of the movement in order to fulfill additional criteria, such as collision or joint-limit avoidance. The authors of [1], [2], [11] already showed that probabilistic representations, such as Hidden Markov or Gaussian Mixture Models (GMM), are well suited to encode the mean and variance information of a movement. Section IV therefore revisits the necessary temporal normalization using Dynamic Time Warping and the movement representation within Gaussian Mixture Models.

Until then, the task is completely learned in a compact, probabilistic representation. Section V describes how this representation can be incorporated into the robot's motion generation. The attractor-based optimization scheme by [12] is extended with a cost function that penalizes differences between the generated and learned movement and that continuously weights these differences with the variance information of the GMM encoding. The result is that the robot does not only repeat the movement, but also adapts to new situations or environments. This can be compared to other approaches such as in [13], where reinforcement techniques are used to achieve the adaptive behavior.

Finally, in section VI we evaluate the presented imitation learning scheme within an interactive experiment using the humanoid robot ASIMO.

The major keypoints of this work in the context of imitation learning with humanoid robots are:

- The variance information over several task demonstrations is used as an importance measure. This information is continuously incorporated into the movement generation process.
- Task spaces are used to model the observed movement task. This handles equally dimension reduction, generalization and the correspondence problem.
- The task learning is based on object trajectories only and no assumptions about the teacher's or robot's postures are made.
- To reproduce a learned movement, an attractor-based movement optimization scheme is utilized that also operates on task spaces.

II. DATA ACQUISITION

The focus of this work lies on learning object-related movements. Due to the concept of task spaces, later described in section III, no assumption on the teacher's posture needs to be made. The data acquisition can therefore rely solely on tracking object trajectories instead of full human postures. Hence, there are no requirements for markers and the robot's on-board vision system can be used to track the objects.

For simplification, a slightly modified version of the color tracking algorithm presented in [14] is used and it is assumed that both objects are colored uniformly. The information that is extracted consists of the absolute position of the objects and their rotation angle around the gaze vector of the stereo camera head[1].

The actual learning of the movement that is introduced in section IV depends on a pre-segmentation of the continuous object trajectories into distinct demonstrations by using motion stillness as a segment border. If the observed objects are held still for a defined duration, the current demonstration is finished and the next one starts when the objects are moving again.

III. MOVEMENT MODELLING

Before a probabilistic representation of a movement task can be learned, the observed demonstrations of the teacher are preprocessed. There are several points that this preprocessing needs to consider. First, the correspondence problem between teacher and robot needs to be solved. This problem results from different kinematic structures of both. As an example, a humanoid robot like ASIMO is smaller than the teacher and lacks some degrees of freedom. Second, the movement task should not be learned within the full configuration space of the humanoid robot. This would lead to a very high-dimensional representation of the movement that doesn't generalize well to novel situations. The approach in [2] solves these problems by performing the movement within the robot's configuration space using kinesthetic teaching and afterwards applying a principal component analysis to reduce the data dimensionality.

Within this work, the concept of task spaces is used to solve the mentioned problems, as it was already proposed by [15]. The observed movement is projected into a task-specific space and the correspondence problem is avoided by solely focussing on the object trajectories without making any assumption on the teacher's postures during the demonstration. This has several advantages. The actual movement of the robot is calculated only during the reproduction. It can therefore take into account new situations, differing robot kinematic structures and additional constraints, such as collision avoidance. Furthermore, the concept of task spaces provides a generalization capability by design.

Returning to the example of pouring from a bottle into a glass, an appropriate task space consists of the position of the bottle relative to the position of the glass and their orientations. Referring to figure 2, the elements of the task space are

$$\mathbf{x}_{\text{task}} = \left(\mathbf{x}_{\text{diff}}^T \; \boldsymbol{\varphi}_b^T \; \boldsymbol{\varphi}_g^T\right)^T \quad . \tag{1}$$

The relative position $\mathbf{x}_{\text{diff}}$ of both objects is defined in the world frame and not in the coordinate system of one of the objects. Otherwise, the orientation of one object would affect the position of the other[2]. The elements $\boldsymbol{\varphi}_b$ and $\boldsymbol{\varphi}_g$ are vectors containing the angle gathered from the vision system

[1]The direction is defined through the object's longest elongation.

[2]However, there are also tasks that would profit from this behavior.

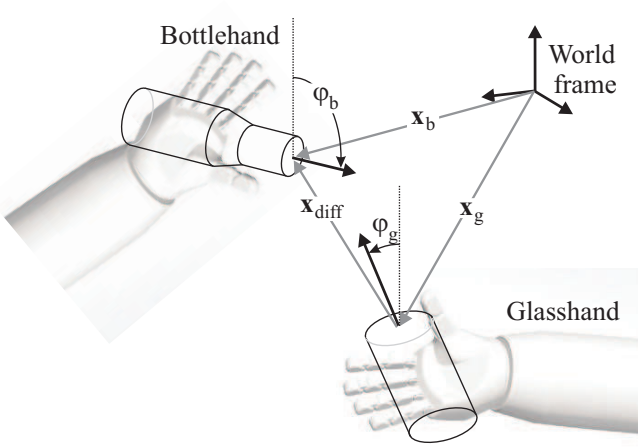

Fig. 2. Illustration of the different variables used for modelling the *pouring task* with task spaces

and a second component that defines the angle's plane to be vertical as it was observed.

If the pouring movement is learned within this task space, a first generalization is already achieved because the robot can repeat the movement at differing absolute positions and therefore adapt to new situations. Note further that task spaces themselves are no restriction of the movement capabilities of the robot. Although it is unlikely necessary, a task space could also comprise the full joint space of the robot.

IV. MOVEMENT LEARNING

The key aspect of imitation learning is to obtain a generalized representation of a movement task from several demonstrations of the teacher. This representation allows the robot to repeat the observed task, but also adapt it to a new environment and situation. The basic idea of using a probabilistic representation for movement tasks is that important task elements are usually invariant over several demonstrations. We therefore directly exploit the variant parts of the movement to allow the robot to fulfill additional criteria, such as collision avoidance or staying balanced, while still reproducing the invariant features of the movement. This section describes how such a representation is learned while section V shows how the variance information is explicitly used during the robot's movement generation.

A. Dynamic Time Warping

After the observed demonstrations are projected into appropriate task spaces, there is one further preprocessing step necessary. To learn meaningful variance information within the probabilistic representation, temporal normalization of the trajectories is crucial. Note that the inter-trial variance between multiple demonstrations is of interest here, not the variance within a single demonstration. If a human teacher performs the same task several times, there will always be non-linear temporal distortions between the demonstrations. Therefore, a temporal normalization has to be applied in advance of learning. The authors of [6], [1], [2], [16] propose different methods, such as the use of left-right Hidden Markov Models or the Dynamic Time Warping (DTW) algorithm. For our approach we have chosen the latter one, because left-right Hidden Markov Models would introduce an additional unwanted smoothing, depending on their number of hidden states.

The principle of the Dynamic Time Warping algorithm [17] is to find a temporal deformation of one signal to minimize the distance to another signal. The first step is therefore the definition of the distance measure. In our case, this distance measure is the Euclidean distance of the elements of the task space the demonstrations are projected into. It is additionally weighted to account for the scale of the individual task space dimensions. Using this distance measure, a matrix $\mathbf{V}$ is filled with the pair-wise distance of all data samples of one demonstration to all data samples of another. Scalar $v_{i,j} \in \mathbf{V}$ then refers to the weighted Euclidean distance of element i of the first and element j of the second demonstration. With the following dynamic programming approach, a path starting from the bottom right element $v_{n,m}$ to $v_{1,1}$ is determined that minimizes the sum of the path's elements:

$$dtw(i,j) = \begin{cases} \infty & \text{for } i = 0 \vee j = 0 \quad , \\ v_{i,j} & \text{for } i = 1 \wedge j = 1 \quad , \\ v_{i,j} + min \begin{pmatrix} dtw(i-1,j-1), \\ dtw(i-1,j), \\ dtw(i,j-1) \end{pmatrix} & \text{else .} \end{cases} \tag{2}$$

Figure 3 illustrates the algorithm with two signals α_A and α_B to be normalized. The color coding represents the magnitude of $v_{i,j}$ and the white line is the path that was found by the Dynamic Time Warping. With the indices of the path, the signals can now easily be warped in the time domain in order to minimize their distance from each other. The gray, dashed lines are example associations for 3 timesteps.

With Dynamic Time Warping all demonstration are non-linearly morphed to share the same length. Figure 4 shows that meaningful inter-trial variance information can only be extracted by preprocessing with Dynamic Time Warping. However, one has to obey that individual timing properties are neglected in favor of the meaningful inter-trial variances. While this is irrelevant for the tasks considered in this work, it probably leads to problems with highly dynamic movements. Hitting a ball with a tennis rack is an example for this, because the dynamic properties of the movement directly influence the achievement of the task goal.

B. Gaussian Mixture Models

After the observation phase (see fig. 1), the data can be learned within a probabilistic representation. We choose Gaussian Mixture Models to learn the underlying probability density function of the observed trajectories $\mathbf{x}_i \in \mathbf{X}$. The dimensionality D of the data samples $\mathbf{x}_i$ equals the number of task space dimensions and an additional temporal dimension, which is also learned by the GMM. The probability density

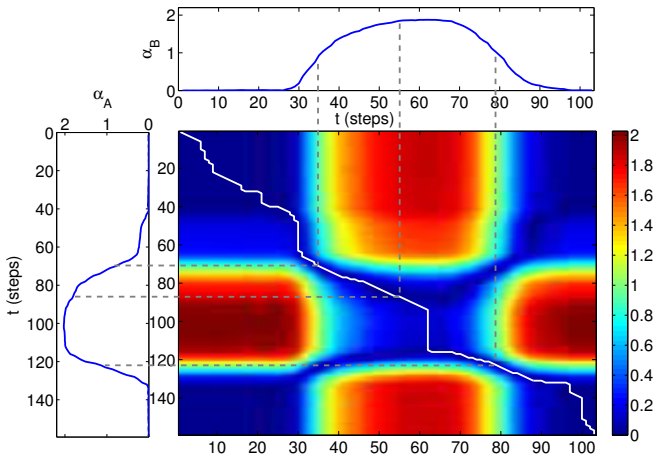

Fig. 3. The Dynamic Time Warping determines the white line that represents the minimal path through the distance matrix of the two signals α_A and α_B. The gray lines indicate that the indices of the path can be used to morph one signal to match the other

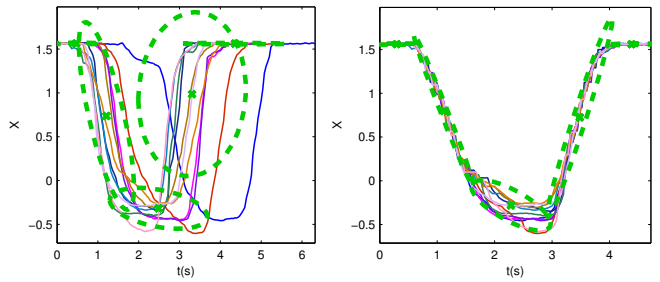

Fig. 4. GMM representation without (left) and with (right) DTW preprocessing, respectively. The green ellipses represent the learned mean and covariance information

function $p(\mathbf{x}_i)$ is estimated using a mixture of K Gaussian distributions:

$$p(\mathbf{x}_i) = \sum_{k=1}^{K} \pi_k p(\mathbf{x}_i|k) \tag{3}$$

where π_k is the a priori probability of Gaussian component k within the GMM ($\sum_{k=1}^{K} \pi_k = 1$) and $p(\mathbf{x}_i|k)$ is the conditional probability density function that can be derived from the D-dimensional normal distribution $\mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, depending on mean vector $\boldsymbol{\mu}_k$ and covariance matrix $\boldsymbol{\Sigma}_k$ of Gaussian k:

$$\begin{aligned} p(\mathbf{x}_i|k) &= \mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \\ &= \frac{1}{\sqrt{(2\pi)^D \cdot |\boldsymbol{\Sigma}_k|}} \cdot e^{-\frac{1}{2}\left((\mathbf{x}_i - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1} (\mathbf{x}_i - \boldsymbol{\mu}_k)\right)} \quad . \end{aligned} \tag{4}$$

The GMM that models the probability density function of the input is therefore fully described by π_k, $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ denoting prior, mean vector and covariance matrix of all Gaussian components k respectively. These parameters can be learned with a standard Expectation-Maximization algorithm [18]. However, the number of Gaussian components and their initialization has to be determined first.

1) Estimating Gaussian components: To estimate the optimal number of Gaussian components, literature proposes different approaches. One for example is the use of the Bayesian Information Criterion[3] (BIC) [19] as a tradeoff between model complexity and representation quality. A typical heuristic is to steadily increase the number of Gaussian components, train the Gaussian Mixture Model, and calculate the BIC value each time. The smallest value in the series of BIC values refers then to the estimated optimal number of components.

Because the training of a Gaussian Mixture Model is considerably cost-intensive and the interactive aspect is keypoint within this work, we modify the common heuristic as follows. Instead of training a full Gaussian Mixture Model within each iteration, only a fast K-Means clustering is applied to $\mathbf{X}$. Then π_k, $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ of all components of the Gaussian Mixture Model are initialized with the cluster information. The full training of the GMM is then skipped and the BIC value is calculated as usual using the following equation:

$$\text{BIC} = -2\mathcal{L} + P \ln(N) \quad . \tag{5}$$

Scalar $\mathcal{L}$ denotes the log-likelihood that the GMM represents all N data samples $\mathbf{x}_i$ and P equals the number of free parameters of the model. Both, $\mathcal{L}$ and P are calculated using the following two equations:

$$\mathcal{L} = \frac{\sum_{i=1}^{N} \log\left(\sum_{k=1}^{K} \pi_k p(\mathbf{x}_i|k)\right)}{N} \quad , \tag{6}$$

$$P = \overbrace{(K-1)}^{\text{for } \boldsymbol{\pi}} + K \left(\overbrace{D}^{\text{for } \boldsymbol{\mu}} + \overbrace{\frac{1}{2} D(D+1)}^{\text{for } \boldsymbol{\Sigma}} \right) \quad . \tag{7}$$

By skipping the time-intensive training of the Gaussian Mixture Models the estimation of the optimal number of components gets less accurate but faster. Figure 5 visualizes a comparison between the common heuristic and our modified version. Both are applied to test data from the experiment in section VI and the BIC values for each number of Gaussian components, ranging from 1 to 30, are calculated. One can observe that the modified heuristic prefers choosing too many rather than choosing too few components for the representation. This is because the Expectation-Maximization algorithm is proved to converge while increasing the log-likelihood and secondly due to the fact that the representation quality increases with the number of Gaussian components. This inaccuracy increases the training time of the Gaussian Mixture Model during the learning phase. However, this is more than compensated by the speedup of the estimation. Estimating the number of Gaussian components in the range of 1 to 30 using our approach is about two orders of magnitude faster than the common heuristic.

2) EM algorithm: After the number of Gaussian components is estimated in the previous step, the components are initialized with the information resulting from the K-Means clustering of the input data $\mathbf{X}$. Afterwards, the Gaussian Mixture Model can be trained using a common

[3]Also referred to as Schwartz Information Criterion.

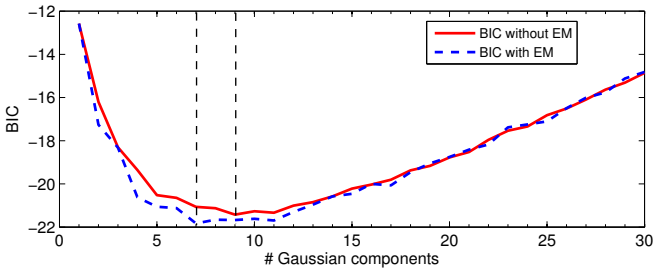

Fig. 5. Comparison of the BIC values of the two heuristics. The faster approach chooses 9 instead of 7 Gaussian components

Expectation-Maximization algorithm [18]. The goal during the optimization is to maximize the log-likelihood that the Gaussian Mixture Model represents the probability density function of the given input data (see eq. 6). For this, two steps are iterated until the change of the log-likelihood is below a certain threshold.

The first step (expectation step) of the algorithm follows the Bayes theorem to calculate the likelihood of each Gaussian component k given the data set $\mathbf{X}$:

$$p(k|\mathbf{x}_i) = \frac{\pi_k p(\mathbf{x}_i|k)}{\sum_{j=1}^{K} \pi_j p(\mathbf{x}_i|j)} \quad (8)$$

$$= \frac{\pi_k \mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \quad ,$$

$$e_k = \sum_{i=1}^{N} p(k|\mathbf{x}_i) \quad . \quad (9)$$

This is calculated by using the parameters of the previous optimization step or, in the beginning, of the initialization with the K-Means clustering.

During the maximization step, the parameters of the GMM are adapted to maximize the likelihood that the GMM represents the probability density function of data $\mathbf{X}$:

$$\pi_k = \frac{e_k}{N} \quad , \quad (10)$$

$$\boldsymbol{\mu}_k = \frac{\sum_{i=1}^{N} p(k|\mathbf{x}_i)\mathbf{x}_i}{e_k} \quad , \quad (11)$$

$$\boldsymbol{\Sigma}_k = \frac{\sum_{i=1}^{N} p(k|\mathbf{x}_i)(\mathbf{x}_i - \boldsymbol{\mu}_k)(\mathbf{x}_i - \boldsymbol{\mu}_k)^T}{e_k} \quad . \quad (12)$$

The algorithm converges to an estimation of the probability density function of all observed input values. Due to the initialization with the K-Means clustering and a good estimation for the number of components, the EM algorithm reaches a local minimum relatively fast.

At this point, the observed movement task is learned in a compact representation that encodes not only the mean movement itself but also a continuous importance weighting in form of variance information. The next section explains how this representation is combined with the motion generation methods in order to reproduce the learned movement task.

V. MOVEMENT OPTIMIZATION

The probabilistic movement representation described in the previous section accounts for the robot's effector movement. However, it does not yet consider the limits associated to joint ranges, self-collisions etc. To handle these aspects, we incorporate a gradient-based trajectory optimization scheme, which has been presented in [12]. It operates on an attractor-based trajectory generation [20] that describes the task space trajectories with attractor dynamics and projects these trajectories to the joint space movement with a kinematic whole body control system. The key idea is to optimize a scalar cost function by finding an optimal sequence of such task space attractor vectors that determines the robot's motion.

For this, we consider an integral scalar cost function over the overall movement composed of two terms. The first term subsumes criteria that depend on single time steps, like costs that depend on the posture of the robot. Specifically, we use criteria to account for collisions and proximities between collidable objects throughout the trajectory and joint limit proximities. The second term subsumes costs for transitions in joint space and depends on the current and the previous time steps. It is suited to formulate criteria like the global length of the trajectory in joint space.

During optimization, we iteratively compute all costs and analytical gradients of the attractor point locations with respect to the chosen criteria and update the location of the attractor points accordingly until convergence. The scheme has already been applied to reaching and grasping problems, and finds solutions within a short time, as such being suitable for interactive scenarios.

We choose the number of attractor points according to the number of Gaussian components of the Gaussian Mixture Model that represents the learned movement. While this is not explicitly necessary, it achieved good results within the experiments and eliminates another free parameter that would elsewise need to be chosen manually.

Further, we extend the set of criteria with a *similarity criterion*. It penalizes the deviation of the robot's task space trajectory from the observed one. The key idea is to apply an adaptive weighting scheme that weights the similarity with the variance of the observation. In phases with higher variance, we assume that the movement doesn't need to be tracked precisely. By assigning a low weight to the similarity criterion, its effect will be reduced, as such giving higher influence to the other criteria governing the movement. This results in a movement that tracks the observed trajectory rather precisely in phases of low variance, while it is characterized by other criteria (joint limit and collision avoidance etc.) in phases of higher variance.

A. Gaussian Mixture Regression

Other approaches apply the Gaussian Mixture Regression to generate the movement previously acquired in the learning process. However, the represented movement does not necessarily account for the limitations (e.g., joint-limits, collisions) of the robot. We therefore use it to initialize an optimization

problem that respects the similarity of the generated and learned movement as one optimization criterion.

Using this regression technique, the mean and variance information of each dimension of the task space is calculated for a given timestep. The mean values then refer to the learned movement and the variance information can be interpreted as an importance measure for parts of the movement. This importance weighting is later directly incorporated into the *similarity criterion* of the optimization process.

Gaussian Mixture Regression is based on the theorem of Gaussian conditioning and the linear combination of Gaussian probability distributions. For the movement reproduction the task space trajectories are needed. Therefore, the temporal dimension can be seen as an input, while the remaining spatial dimensions are the output. Under this assumption, the means and covariance matrices of the Gaussian components can be split into a temporal (denoted by subscript t) and a spatial (denoted by subscript s) part:

$$\boldsymbol{\mu}_k = (\mu_{t,k}, \boldsymbol{\mu}_{s,k}^T)^T \quad , \tag{13}$$

$$\boldsymbol{\Sigma}_k = \begin{pmatrix} \sigma_{tt,k} & \boldsymbol{\sigma}_{ts,k}^T \\ \boldsymbol{\sigma}_{st,k} & \boldsymbol{\Sigma}_{ss,k} \end{pmatrix} \quad . \tag{14}$$

Given the temporal input x_t, the conditional expectation $\hat{\mathbf{x}}_{s,k}$ and the estimated covariance matrix $\hat{\boldsymbol{\Sigma}}_{s,k}$ for each Gaussian component k can be calculated using the equations of the Gaussian conditioning theorem:

$$\hat{\mathbf{x}}_{s,k} = \boldsymbol{\mu}_{s,k} + \boldsymbol{\sigma}_{st,k}(\sigma_{tt,k})^{-1} \cdot (x_t - \mu_{t,k}) \quad , \tag{15}$$

$$\hat{\boldsymbol{\Sigma}}_{s,k} = \boldsymbol{\Sigma}_{ss,k} - \boldsymbol{\sigma}_{st,k}(\sigma_{tt,k})^{-1} \cdot \boldsymbol{\sigma}_{ts,k}^T \quad . \tag{16}$$

These conditional expectations and covariance matrices are then mixed according to the probabilities β_k that input x_t is modelled by Gaussian k:

$$\hat{\mathbf{x}}_s = \sum_{k=1}^{K} \beta_k \hat{\mathbf{x}}_{s,k} \quad , \tag{17}$$

$$\hat{\boldsymbol{\Sigma}}_s = \sum_{k=1}^{K} \beta_k^2 \hat{\boldsymbol{\Sigma}}_{s,k} \tag{18}$$

with

$$\beta_k = \frac{\pi_k p(x_t|k)}{\sum_{i=1}^{K} \pi_i p(x_t|i)} = \frac{\pi_k \mathcal{N}(x_t; \mu_{t,k}, \sigma_{tt,k})}{\sum_{i=1}^{K} \pi_i \mathcal{N}(x_t; \mu_{t,i}, \sigma_{tt,i})} \quad . \tag{19}$$

Evaluating these equations for consecutive values of x_t results in an estimation for the means of all task space dimensions over time and their associated covariance matrices. For simplification, we introduce two new symbols. The symbol $\hat{\boldsymbol{\mu}}_t$ stands for the value $\hat{\mathbf{x}}_s$ at timestep t. The elements of $\hat{\boldsymbol{\sigma}}_t$ refer to the diagonal of $\hat{\boldsymbol{\Sigma}}_s$ at timestep t.

B. Similarity criterion

The attractor points that are defined in the task space of the learned movement, are then initialized with the mean values that were calculated with the Gaussian Mixture Regression. Without optimization this leads to a movement that does not fully match the learned one and, more importantly, that allows self-collisions. Therefore, an optimization, according to the scheme mentioned in the beginning of this section, needs to be performed. To achieve similarity between the learned and the reproduced movement, this scheme is extended with the *similarity criterion*:

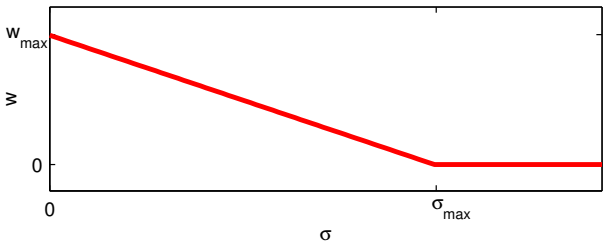

Fig. 6. Function that maps the variance to a weighting factor

$$c_{\text{im}} = (\mathbf{x}_t - \hat{\boldsymbol{\mu}}_t)^T \mathbf{W}_t (\mathbf{x}_t - \hat{\boldsymbol{\mu}}_t) \quad . \tag{20}$$

For each timestep t, this cost function penalizes a deviation of the state of the task space $\mathbf{x}_t$ from the learned mean values $\hat{\boldsymbol{\mu}}_t$ weighted with the time-dependent diagonal matrix $\mathbf{W}_t$ that is calculated using the estimated variances $\hat{\boldsymbol{\sigma}}_t$:

$$w_{i,i} = \begin{cases} w_i^{\max} - \frac{w_i^{\max}}{\sigma_i^{\max}} \cdot \hat{\sigma}_{t,i} & \text{for } 0 \leq \hat{\sigma}_{t,i} < \sigma_i^{\max} \\ 0 & \text{for } \hat{\sigma}_{t,i} \geq \sigma_i^{\max} \end{cases} \quad . \tag{21}$$

Variables with superscript *max* are constants that can be used to shape the mapping between the variance and the weighting factor (see fig. 6) in order to account for the scale of the individual task space dimensions. The gradient of the *similarity criterion* that is used during the trajectory optimization is

$$\frac{\partial c_{\text{im}}}{\partial \mathbf{x}_t} = 2(\mathbf{x}_t - \hat{\boldsymbol{\mu}}_t)^T \mathbf{W}_t \quad . \tag{22}$$

With this cost function, the variance information is directly included in the optimization process, continuously over all dimensions of the task space and all timesteps. The robot is allowed to diverge from variant and therefore unimportant parts of the movement in order to minimize other cost functions (e.g., collision costs or joint-limit costs).

VI. EXPERIMENT

The experiment presented in this section is the *pouring task* example that is discussed throughout this paper. The robot is required to imitate a pouring motion of approximately 4-5 seconds length. Figure 7 shows the experimental setup and the enclosed video contains the whole imitation learning scenario.

The teacher stands in front of ASIMO and demonstrates the task five times. As mentioned in section II, the distinct demonstrations are separated by holding the objects still for about one second. After the task was observed by the robot, the movement information is projected into the task space, which consists of the relative object positions and orientations, such as already described in section III. The information is then temporally aligned using the Dynamic Time Warping algorithm and learned within a Gaussian

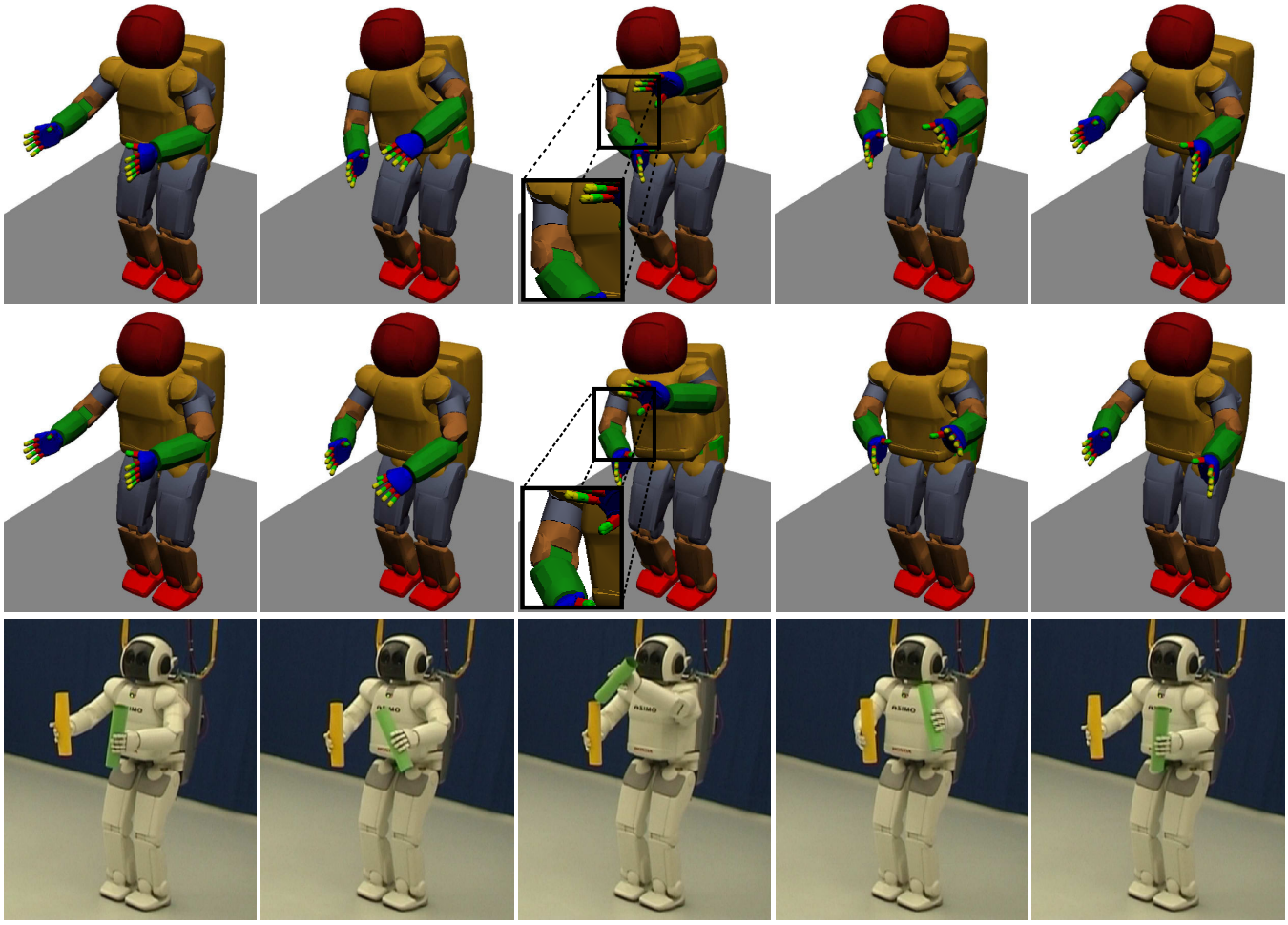

Fig. 8. Imitation of the *pouring task* without (top row) and with collision costs (center row). Note how the movement is optimized in order to avoid the self-collision in the middle of movement while the *pouring task* is still being performed. The bottom row shows the real performance of ASIMO after the movement optimization

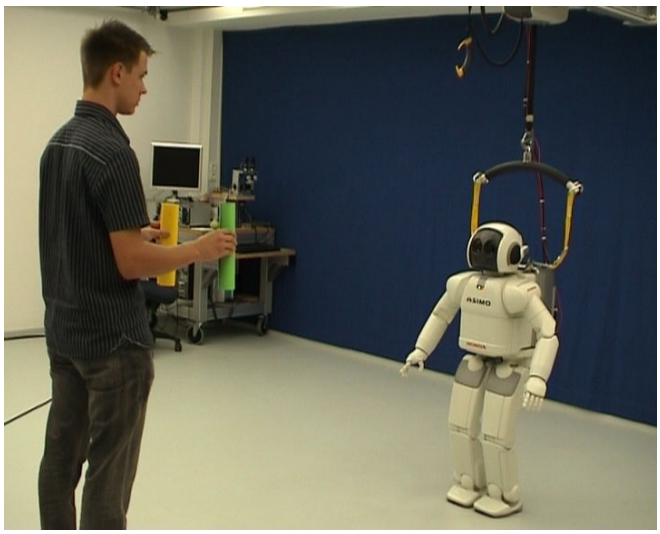

Fig. 7. Setup for the interactive imitation learning experiment

Mixture Model. The learning takes about two seconds and additional 10 seconds are needed for the subsequent attractor-based movement optimization. Afterwards, the robot is able to directly reproduce the learned movement task. The cost terms that are included in the optimization penalize collisions, proximities to joint-limits and deviations from the learned movement.

Figure 8 shows snapshots of the robot's performance of the *pouring task*. The top row shows the resulting trajectory if the cost term that penalizes self-collisions is left out during the optimization process. For the center row this term is included. One can observe that in order to avoid self-collisions between the right arm and the upper body, the robot's movement diverges from the optimal imitation trajectory. Figure 9 illustrates this in more detail for the relative position of both hands in direction of the Z axis. With an increasing weight of the collision costs, the robot diverges even more from the actual learned movement. However, this behavior is limited to the high variant part between timestep 0.5s to 1.5s. This shows that the variance is explicitly exploited during the movement optimization in

order to diverge from less important rather than critical parts of the learned movement. Further, it can be seen that the robot performs the task in a more dynamic way than the teacher showed it. The glass hand actively moves towards the bottle. This is a wanted behavior that results from the chosen task space that comprises relative positions.

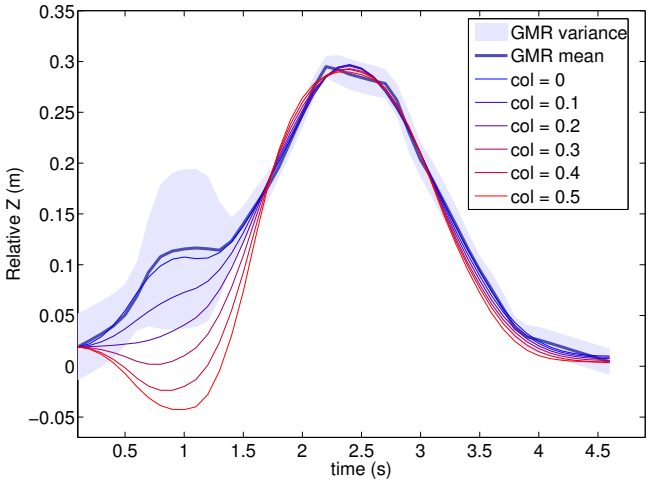

Fig. 9. The variant parts of the learned movement are tracked less accurately if the weight of the collision criterion increases. Less variant and therefore important parts of the learned movement task are still fulfilled correctly

VII. CONCLUSION

We have presented a framework that allows a humanoid robot to learn new movement tasks through imitation. Unlike other imitation learning approaches, we employ task spaces in order to avoid the correspondence problem, reduce the dimensionality of the training data and to achieve a first generalization. The movement task is defined through object trajectories, which are observed using a marker-less stereo vision system. Statistical information coming from multiple task demonstrations is learned within Gaussian Mixture Models.

For the reproduction of the movement task, a previously introduced attractor-based movement optimization scheme is utilized. This scheme is extended with a new cost term that rates the similarity between the produced movement and the learned one. This *similarity criterion* directly incorporates the variance information from the learned representation. This enables the robot to adapt to new situations and to diverge from the learned movement in phases of high variance, while still fulfilling less variant and therefore more important parts of the movement. Besides similarity, this behavior therefore concurrently regards other criteria, such as collision avoidance. We have presented an interactive experiment with the humanoid robot ASIMO that confirms this behavior.

The presented work provides a good starting point for our future research in direction of imitation learning. Major points will include the automatic determination of task spaces, based not only on statistical information, but also on interactive guidance and insights from parent-infant research. Further, the interactive aspect is of great interest. The whole imitation learning process should become a spontaneous interaction rather than a fixed dialog.

REFERENCES

[1] A. Billard, S. Calinon, and F. Guenter, "Discriminative and adaptive imitation in uni-manual and bi-manual tasks," *Journal of Robotics and Autonomous Systems*, vol. 54, no. 5, pp. 370–384, May 2006.

[2] S. Calinon, F. Guenter, and A. Billard, "On learning, representing, and generalizing a task in a humanoid robot," *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 37, no. 2, pp. 286–298, Apr. 2007.

[3] R. Zöllner, O. Rogalla, M. Ehrenmann, and R. Dillmann, "Mapping complex tasks to robots: Programming by demonstration in real-world environments," *Advances in Human-Robot Interaction*, vol. 14, pp. 119–136, 2004.

[4] D. C. Bentivegna, C. G. Atkeson, and G. Cheng, "Learning tasks from observation and practice," *Journal of Robotics and Autonomous Systems*, vol. 47, pp. 163–169, Jun. 2004.

[5] A. Chella, H. Dindo, and I. Infantino, "A cognitive framework for imitation learning," *Journal of Robotics and Autonomous Systems*, vol. 54, no. 5, pp. 403–408, May 2006.

[6] S. Calinon, F. Guenter, and A. Billard, "On learning the statistical representation of a task and generalizing it to various contexts," in *Proceedings of the 2006 IEEE International Conference on Robotics and Automation*, 2006, pp. 2978–2983.

[7] J. B. Cole, D. B. Grimes, and R. P. N. Rao, "Learning full-body motions from monocular vision: Dynamic imitation in a humanoid robot," in *Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007.

[8] C. A. Acosta-Calderon and H. Hu, "Robot imitation: Body schema and body percept," *Applied Bionics and Biomechanics*, vol. 2, no. 3, pp. 131–148, Mar. 2005.

[9] T. Asfour, F. Gyarfas, P. Azad, and R. Dillmann, "Imitation learning of dual-arm manipulation tasks in humanoid robots," in *International Conference on Humanoid Robots, 2006 6th IEEE-RAS*, 2006, pp. 40–47.

[10] D. B. Grimes, R. Chalodhorn, and R. P. N. Rao, "Dynamic imitation in a humanoid robot through nonparametric probabilistic inference," in *Proceedings of Robotics: Science and Systems (RSS'06)*. MIT Press, 2006.

[11] T. Inamura, I. Toshima, H. Tanie, and Y. Nakamura, "Embodied symbol emergence based on mimesis theory," *The International Journal of Robotics Research*, vol. 23, pp. 363–377, 2004.

[12] M. Toussaint, M. Gienger, and C. Goerick, "Optimization of sequential attractor-based movement for compact movement representation," in *Proceedings of the IEEE-RAS/RSJ International Conference on Humanoid Robots*, Dec. 2007.

[13] F. Guenter, M. Hersch, S. Calinon, and A. Billard, "Reinforcement learning for imitating constrained reaching movements," *Advanced Robotics*, vol. 21, no. 13, pp. 1521–1544, 2007.

[14] B. Bolder, M. Dunn, M. Gienger, H. Janssen, H. Sugiura, and C. Goerick, "Visually guided whole body interaction," in *IEEE International Conference on Robotics and Automation (ICRA 2007)*, 2007.

[15] M. Gienger, H. Janssen, and C. Goerick, "Exploiting task intervals for whole body robot control," in *Proceedings of the International Conference on Intelligent Robots and Systems*, 2006.

[16] W. Ilg, G. H. Bakir, J. Mezger, and M. A. Giese, "On the representation, learning and transfer of spatio-temporal movement characteristics," *International Journal of Humanoid Robotics*, pp. 613–636, Dec. 2004.

[17] C. A. Ratanamahatana and E. Keogh, "Everything you know about dynamic time warping is wrong," *Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2004.

[18] S. Dasgupta and L. J. Schulman, "A two-round variant of em for gaussian mixtures," in *UAI '00: Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, 2000, pp. 152–159.

[19] G. Schwarz, "Estimating the dimension of a model," *Annals of Statistics*, vol. 6, no. 2, pp. 461–464, 1978.

[20] M. Gienger, H. Janssen, and C. Goerick, "Task-oriented whole body motion for humanoid robots," in *Proceedings of the IEEE-RAS/RSJ International Conference on Humanoid Robots*, Dec. 2005.