Learning Actions from Observations

Primitive-Based Modeling and Grammar

© STOCKBYTE, EYEWIRE, DIGITAL VISION & BRAND X PICTURES

Robot Learning

In the area of imitation learning, one of the important research problems is action representation. There has been a growing interest in expressing actions as a combination of meaningful subparts called action primitives. Action primitives could be thought of as elementary building blocks for action representation. In this article, we present a complete concept of learning action primitives to recognize and synthesize actions. One of the main novelties in this work is the detection of primitives in a unified framework, which takes into account objects and actions being applied to them. As the first major contribution, we propose an unsupervised learning approach for action primitives that make use of the human movements as well as object state changes. As the second major contribution, we propose using parametric hidden Markov models (PHMMs) [1], [2] for representing the discovered action primitives. PHMMs represent movement trajectories as a function of their desired effect on the object, and we will discuss 1) how these PHMMs can be trained in an unsupervised manner, 2) how they can be used for synthesizing movements to achieve a desired effect, and 3) how they can be used to recognize an action primitive and the effect from an observed acting human.

The need and motivation for imitation learning have been widely documented in the area of robotics during the last decade [3]–[6]. The open challenge in imitation learning is to develop a compact and flexible representation that can be used for action planning, action recognition, and action synthesis. Many approaches follow the arguments raised in [7] and [8] that human actions are composed of action primitives similar to human speech, being composed of phonemes, and that the same parts of the human brain seem to be responsible for the generation and recognition of human actions.

Hence, the use of action grammars based on action primitives is a plausible representation. This work is based on the idea that all actions can be described by a finite set of action primitives [9] and that there exists a grammatical description of how these primitives can be combined [9], [10]. It is, however, less obvious how 1) these action primitives and the associated grammars can be extracted automatically from visual observations without any input in addition to the visual observations [10], 2) how they should be represented in terms of data structures, and 3) how they can be used simultaneously for action recognition and action synthesis.

For the learning of primitives and grammars, most of the state-of-the-art approaches employ an off-line, supervised learning stage, where the primitives are labeled by the teacher and are then used in the recognition stage. The problem of on-line continuous learning has been studied recently in [11], where segmentation and classification are performed in an unsupervised manner. In this article, we study the problem of unsupervised detection of action primitives and their subsequent use for action recognition and action synthesis tasks. One novelty in our work is to consider actions, where the teacher is interacting with objects rather than considering only

Digital Object Identifier 10.1109/MRA.2010.936961

BY VOLKER KRÜGER, DENNIS L. HERZOG, SANMOHAN BABY, ALEŠ UDE, AND DANICA KRAGIC

the free body movements. The changes in object state are facilitated directly in the primitive learning phase.

Once the primitives are detected, we can use different techniques to represent them parametrically. A hierarchical representation is necessary when attempting to integrate low-level control and high-level action-planning aspects. An additional contribution to our work is the use of PHMMs for modeling action. Although many different body movements are able to induce the same changes in the object state, PHMMs offer a unified framework that allows to combine the movement trajectories in terms of their effect on the object.

To make the contributions clearer, let us consider the following scenario. A robot is observing a human performing actions on objects such as taking object *A* and placing it at location *B* or inserting object *A* into object *B*. After the demonstration stage, the robot is requested to identify 1) a set of action primitives and 2) the corresponding grammar from the observations. We relate the expected outcome of the system to a graphical model (Figure 1). In Figure 1(a), the actions on the objects may be described with a general approach–act–remove cycle. In Figure 1(b), more intuitive and hand-selected action primitives with the corresponding grammar are shown. The work presented here derives these action primitives and the corresponding grammar automatically from a set of demonstrations. These are then further modeled by a PHMM-based representation and used for generating actions on the robot to achieve a desired effect as well as to allow the robot to recognize the primitives and their effects from human performances.

This article is organized as follows. In the "Background and Motivation" section, we give an overview of the previous research and motivation for our work. In the "Unsupervised Learning of Action Primitive Classes According to the Effect on Objects" section, we discuss our approach for learning

Action primitives could be thought of as elementary building blocks for action representation.

action primitives and the corresponding grammar. In the "Building PHMMs for Each Primitive Class" section, we present our PHMMs for modeling the action primitives for the purpose of synthesis (the "Synthesizing Actions with PHMMs" section) and action recognition (the "Recognition of Humans' Actions" section). The final comments are given in the "Conclusions" section.

Background and Motivation

The derivation of action primitives is not trivial; ideally, the demonstrations are repeated, possibly, even by different teachers and different ways to assure good statistics. When actions are performed on objects, the objects and/or the human teacher may not necessarily be at the same location so that the visual appearance of these performances is likely to vastly differ from each other. One obvious way to look for primitives in demonstrations is to search for statistical cooccurrences, i.e., compare trajectories and identify corresponding and reoccurring parts [13]. However, since the trajectories can vary from each other if the human agent or involved objects are at different locations, such an approach will generate a lot of primitives [13]. Thus, an important challenge is to identify the primitives, independent of the variability in appearance.

To solve this problem, we would like to argue that, for object manipulation actions, one should consider both human movements and the objects to which these actions are applied. Indeed,

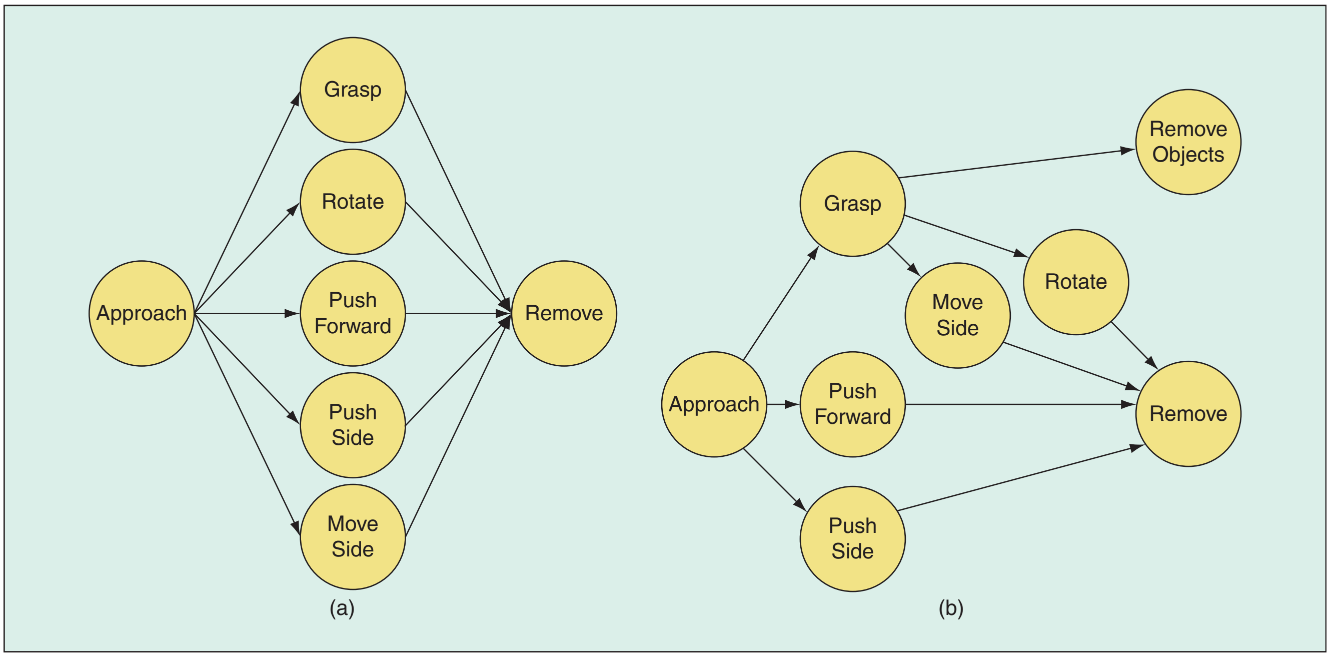


Figure 1. *(a) A set of demonstrated actions. (b) The intuitive and hand-selected action primitives with the corresponding grammar [12].*

An open challenge in imitation learning is to develop a compact and flexible representation that can be used for action planning, action recognition, and action synthesis.

actions and objects seem to be intertwined, and we observe that a human/humanoid action can be seen from two different perspectives: 1) we look at an action as being defined by movements of body parts of the humanoid agent. This is what we need for three-dimensional (3-D) body tracking and movement synthesis; 2) we can also look at actions as being defined by the effect the human movement has on an object, e.g., push object and rotate object. By looking at the state of the object, the effect of a human movement is a change in that object state. Defining the movement space as the space of human movements and the object state space as the space of object states, we define a dual view of the human actions, one from the movement perspective and one from the object perspective. Indeed, one is tempted to identify the effect of movement to be the semantics of that corresponding movement.

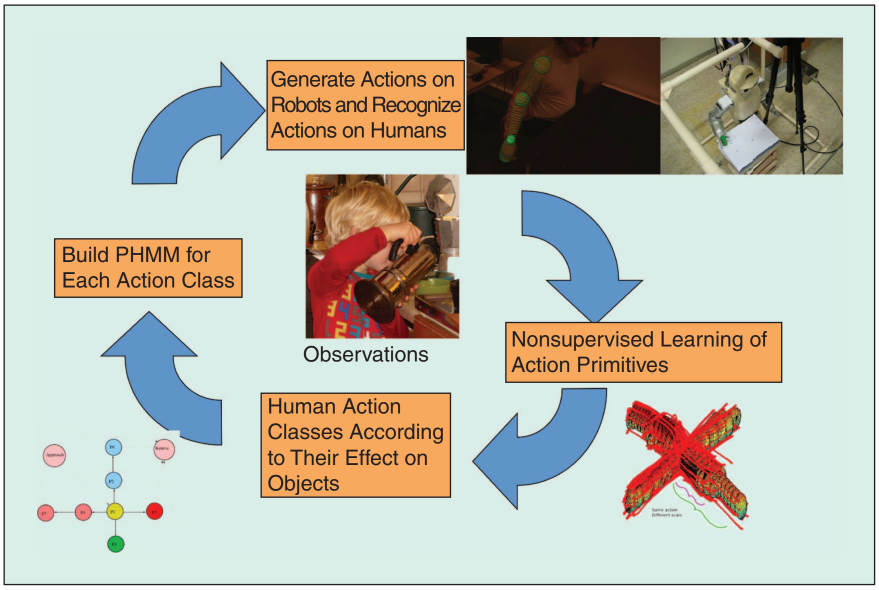
Therefore, instead of analyzing the movement space, as done in [12] and [13], we suggest to approach the detection of action primitives by analyzing the object state space. This way, we are able to identify all human movement trajectories to be instances of the same primitive as long as they induce the same effect on the object.

The concept that objects and actions are intertwined is of course not new to robotics researchers [14]–[20]. However, the focus of many of these works is to apply known motor actions to learn about objects [14], [15], [18], [20], either to segment them or to learn about their affordances. While some other works consider the role of imitation learning within such systems [17], [19], the objects are still not used to support the segmentation of motor experiences. The main difference between these works and our research is that we primarily use objects to gain better action knowledge, e.g., to automatically extract motor primitives from sensorimotor experiences and to parameterize them with respect to the object states. In this sense, our work is complementary to the aforementioned research.

A major difficulty in using the effects of human movements for identifying the primitives instead of trajectories in movement space is that state-of-the-art approaches such as hidden Markov models (HMMs) or HMM networks [21] cannot be easily used anymore for recognizing and synthesizing movements on objects: they only allow to model motor primitives with respect to a specific desired effect. For example, one HMM can only represent one pointing movement in a specific direction. For each new pushing direction, a corresponding HMM needs to be constructed. Therefore, a parametric representation is required that will allow us to recognize and synthesize learned movements regardless of the specific desired effect, e.g., a pointing direction. The PHMM is such a representation, because it takes the effect of movement as a parameter and generates a new trajectory in movement space that would lead to the desired effect. PHMMs are able to collapse an entire HMM network into a single HMM if the variations between the HMMs in the network are due to different parameterizations.

In that respect, the main contributions of our work are as follows:

- the definition of duality between the movement space and object state space
- unsupervised learning procedure for detecting action primitives
- learning of the underlying action grammar
- using PHMMs for modeling trajectories in movement space based on their effect in the object state space
- nonsupervised training of PHMMs based on both the trajectories in the movement space and the effect of the movements on the object state space
- using the PHMMs for action synthesis, where a desired effect is given such that a robot should generate the necessary movement
- using PHMMs for action recognition, where we require to identify the effect and action primitive based on the observed movement.


Generate Actions on Robots and Recognize Actions on Humans
Build PHMM for Each Action Class
Observations
Nonsupervised Learning of Action Primitives
Human Action Classes According to Their Effect on Objects
Figure 2. *The different steps involved in learning actions.*

Our concept is illustrated in Figure 2 and consists of the following steps:

1) Nonsupervised learning of action primitives identifies the selection of necessary action primitives. This will be done by analyzing the

effects of the actions on objects rather than action trajectories directly and store all the different performances of each of the primitive in an action equivalence class. For example, the equivalence class of the primitive reach for object contains all the action trajectories of reach for an object primitive.

2) To build a PHMM for each of these primitive classes based on the movement trajectories and their effects.

3) To generate actions on robots that achieve a desired effect and recognize action primitives and their effects from the observed human actions.

In the following, we will discuss each of the steps in detail. Each section contains its own experimental results.

Unsupervised Learning of Action Primitive Classes According to the Effect on Objects

In this section, we discuss about unsupervised learning of action primitives based on the effect of actions on objects.

In the first two steps in Figure 2, we evaluate the trajectories that are caused by the human actions in the object state space. For this purpose, let an action be represented by a pair $[H_t^i\ O_t^i]$ of hand and arm trajectories H_t^i in movement space $\mathcal{M}$ and object trajectories O_t^i in object state space $\mathcal{O}$. While the trajectories H_t^i in $\mathcal{M}$ are given by the marker locations on the hand and arm and O_t^i are given by the object locations and orientations, i denotes the different demonstrations (the demonstrations are recorded using a VICON system and are then automatically extracted from the continuous flow of movements). We propose analyzing O_t^i to detect joint trajectories across the different action effects, which give rise to a set of primitives. This is an important difference from [22], where human joint data are used to identify action primitives. Having found primitives in $\mathcal{O}$, i.e., a segmentation for each of the O_t^i into these primitives, we are able to segment H_t^i in the same way. If segmentation is done for all training movements, we obtain sets of human trajectories, where each set corresponds to a specific primitive (specific effect) in $\mathcal{O}$. In other words, each set is an equivalence class of human movements modulo the effect these movements have on the object state space (Figure 3).

As movements are considered to be equivalent if their effect is the same, an obvious key question is how the quantization of the object state space should be done. For example, if $\mathcal{O}$ is quantized in terms of object location, then, e.g., two push movements are the same if the complete 3-D movements relative to the initial object locations are the same up to a predefined uncertainty. We have investigated in our work a quantization that is based on the change of object location in terms of Euclidean coordinates, which we call as *Euclidean quantization*.

The approach in this section provides us with 1) sets of movements that have the same effect on the object with respect to the chosen quantization of $\mathcal{O}$ and 2) how precisely the object states were changed by each of the movements. As discussed in the "The PHMM" section, both pieces of information will be used during unsupervised training of a PHMM for each of the detected equivalence classes.

Nonsupervised learning of action primitives identifies the selection of necessary action primitives.

Modeling Object–Action Interactions as HMMs

In the first step, we analyze the trajectories O_t^i to identify the primitives in the object state space $\mathcal{O}$. Here, the trajectories in $\mathcal{O}$ are six-dimensional (6-D), describing the location and orientation of the object. All sequences O_t^i available in the database are processed sequentially, and each one is modeled as an ordered sequence of 6-D Gaussian mixtures. The trajectories are divided into pieces of approximately equal length. Each of the segments is modeled with a Gaussian, where the mean and covariance are given by the mean and covariance of that segment. This way, we represent, initially, each trajectory in the entire database as a left–right HMM [23] (see also "Sequence Generation and Left–Right HMMs" section). We denote the HMM-like model corresponding to the sequence O_t^i by $\lambda_i = (A^i, B^i, \pi^i)$, where the observation densities in B are given by the Gaussians (each one associated with a single hidden state), A models the transitions of a left–right model with an even distribution, and π is given by the first state. To compute the joint HMM λ_F, we start by setting $\lambda_F = \lambda_1$. Then, λ_F is modified and built recursively. For each of the subsequent $\lambda_k, k > 1$, we do the following: compare the states of the HMMs λ_F and λ_k pairwise by computing their distance using

$$D_{\mathrm{KL}}(Q\|P) = \frac{1}{2}\left(\log\frac{|\Sigma_1|}{|\Sigma_0|} + \mathrm{tr}(\Sigma_1^{-1}\Sigma_0) + (\mu_1-\mu_0)^T\Sigma_1^{-1}(\mu_1-\mu_0) - n\right), \qquad (1)$$

where D_{KL} is the Kullback-Leibler divergence of two Gaussians $\mathcal{N}(x, \mu_0, \Sigma_0)$ and $\mathcal{N}(x, \mu_1, \Sigma_1)$. Here, n is the dimension of the data, and all state pairs for which the distance falls below

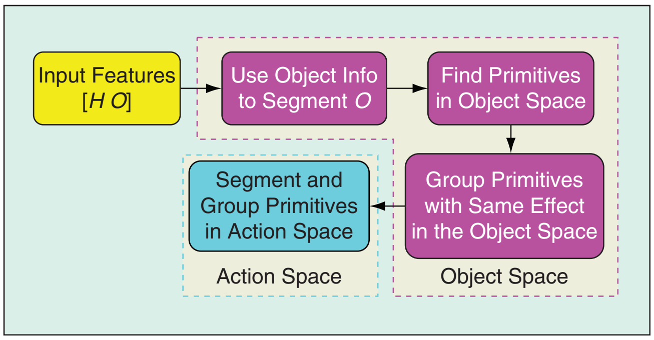

Input Features [H O]
Use Object Info to Segment O
Find Primitives in Object Space
Group Primitives with Same Effect in the Object Space
Segment and Group Primitives in Action Space
Action Space
Object Space

Figure 3. *The inputs H and O denote the action and object features. The object features are first analyzed and segmented. This is then used to extract the primitives in the action space. The magenta boxes correspond to analysis in the object state space, while the cyan box represents the analysis in the action space.*

a predefined threshold are merged into one state. The parameters for the merged state are given by

$$\Sigma^{-1} = \omega\Sigma_0^{-1} + (1-\omega)\Sigma_1^{-1}, \quad (2)$$

$$\mu = \Sigma(\omega\Sigma_0^{-1}\mu_0 + (1-\omega)\Sigma_1^{-1}\mu_1). \quad (3)$$

Here, ω is a weighting parameter that controls the contribution from each of the states. We have used $\omega = 0.5$ to have equal weighting for each of the states. All transitions to the merged state in λ_k are redirected to the modified state, and all transitions from the merged state in λ_k are made to be from the modified state. Each state in λ_k that is not merged with a state is added to λ_F as a new state. At the end of this process, we obtain a single HMM λ_F that models the entire training set, and the hidden states are associated with the observation densities.

After this procedure, we have a single joint HMM λ_F that models all data in the database. In the next step, we identify for each trajectory O_t^i the optimal sequence of HMM states by using the Viterbi algorithm. At this moment, all continuous trajectories are turned into a discrete sequence of HMM states.

This is visualized in Figure 4 by a pushing-action example and 3-D object-location parameters (the rotation parameters were omitted here for clarity). While a human pushes an object on a table, the 3-D object coordinates change. Figure 4 shows the pushes in four directions. To prevent location dependencies, only the differences with respect to the initial object state are considered. One can see that the Gaussians close to the origin appear larger than those that are further away. This reflects that, in the example training set, the variability of short movements was larger.

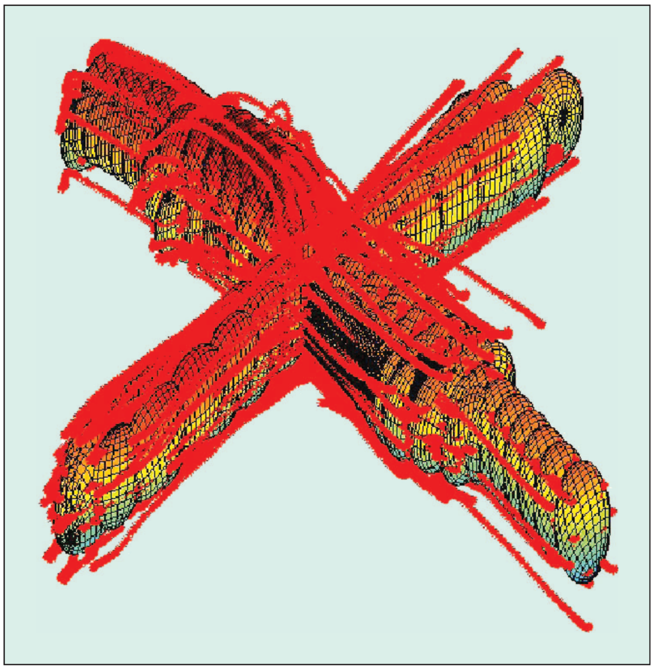


Figure 4. The ellipsoids show the contours of Gaussians used to cover the 3-D location parameters of data. The lengths of the trajectories indicate how much distance the object was moved.

Grouping of HMM States into Primitives in Object Space

The next step aims at detecting the action primitives. After having sampled the trajectories into discrete state sequences, we are able to interpret the discrete state sequences as strings. The action primitives are the substrings that are either common to different state sequence strings or are unique to a single sequence. Common substring selection is done by using a longest common substring (LCS) approach that allows to find for two strings S and T the longest string that is a substring to both S and T. Two steps are required. 1) The HMMs have self-transitions that model the variability in the execution speed of the actions. This means that the discrete state sequences contain not only state changes but also state repetitions. To assure invariance to speed, we discard in the first step the state repetitions so that only the state changes remain. 2) We can now consider the learning of the final primitives from the discrete state sequences as an LCS problem [13], which we solve using a dynamic programming approach [24]; see [13] for a detailed discussion of using the LCS for primitive detection (Figure 5). This results in a small set of primitives in $\mathcal{O}$, and each original continuous trajectory O_t^i can now be represented as a sequence of primitives $p_1^i, p_2^i \ldots p_N^i$.

Segmentation and Grouping in Movement Space

In a final step, we start from a sequence $p_1^i, p_2^i \ldots p_N^i$ of primitives for the object state trajectory O_t^i and propagate the same segmentation to the trajectory H_t^i in $\mathcal{M}$. While doing this for all trajectory pairs $[H_t^i\ O_t^i]$, all pieces of H_t^i that inherit the same (effect) primitive p are combined into one single equivalence class $\hat{p}$. In other words, all movements in the equivalence class $\hat{p}$ have the same effect on the object, while any two sequences from two different equivalence classes $\hat{p}$ and $\hat{p}'$ have different effects in $\mathcal{O}$.

Experimental Results

We have tested our approach for learning action primitives on a rerecorded version of the data set that was previously used by Vicente et al. [12]. In their work, a data set of different human arm actions doing manipulative tasks on objects in a table-top scenario was recorded, and the ground truth was generated. Figure 6(b) shows the experimental setup for the manipulative arm actions on objects on a table top.

In Figure 1(a), one can see the different actions that were recorded. Hand-selected action primitives were taken as ground truth [see Figure 1(b)]. The aim of our experiment is to learn these intuitive action primitives automatically. Since the original data set of [12] was lacking data about the object

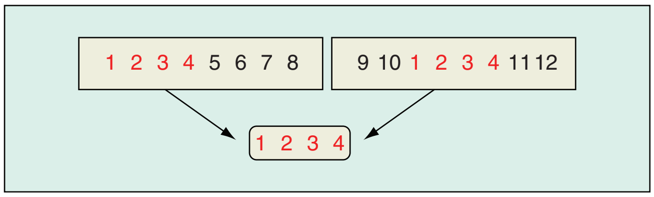


Figure 5. On the top, we have two state sequences. The LCS is (1 2 3 4), as computed by the LCS approach. The final set of primitives is in this case (1 2 3 4), (5 6 7 8), (9 10), and (11 12).

state, we rerecorded the data set using our VICON motion capture system, with markers placed on the human as well as object. In addition, the new data set provides a larger variability in object location as well as four different directions for the push and move movements (up, down, left, and right). The difference between move and push in our data set is that the push movement moves the object on the table while the move action lifts the object from the table to place it at a new location. Figure 6(a) shows the setup for the new recording that includes the object–action interaction. As object states, we use the object locations and object orientation. For the Euclidean quantization of $\mathcal{O}$, the finally recovered action primitives are shown in Figure 7. As can be seen, we recover the move and push primitives along the four different directions available in the training data and the rotate, approach, and remove primitive. The rotation movement was identified correctly, but the grasping movement could not be detected. The reason for this is that the grasp action by itself does not induce any object state change in our training data. The boxes in Figure 7 denote the primitives in the object space and the induced primitive in the movement space. If we consider the box for push right, it has three primitives, each of which corresponds to the repetitions in our training data in which the object was pushed to the right at three different distances. To represent the largest distance moved in this case, we need three primitives, namely, P1, P2, and P3, and for the next shorter distance, we need only P1 and P2, and the shortest distance moved is represented by P1.

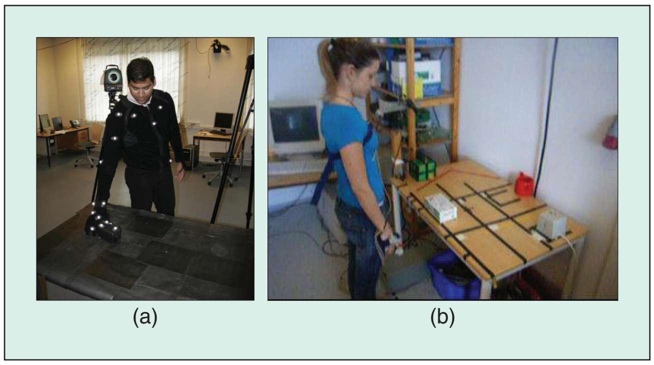


Figure 6. *(a) The markers are attached to both person and object. The object can be moved from any position to any other position on the table. (b) Experimental setup in [12]. The table is marked with locations where the object can be moved around. Object positions are not recorded.*

Building PHMMs for Each Primitive Class

In the previous section, we have presented an approach that can detect a set of action primitives from a set of observed human movements. The result of the learning process was 1) a set of equivalence classes with arm movements that are equivalent in terms of their effect on the object state space and 2) the precise effect of each of these movements on the object state. For example, using polar quantization, all push and move movements were clustered into the same class. Each arm movement, $\boldsymbol{f}(t)$, is described in movement space through trajectories $\boldsymbol{f}_i(t)$ of 3-D locations (i.e., the shoulder, the elbow of the right arm, the wrist, the index finger, its knuckle, and the thumb of the right hand) combined as $\boldsymbol{f}(t) = (\boldsymbol{f}_i(t))_{i=1}^{6}$. Each movement is stored with its precise effect $\boldsymbol{\phi}$, e.g., the initial location x_0 of the object and the final location x_1.

In this section, we introduce PHMMs [1] as a compact representation that allows to model each action primitive (equivalence class) in a way that can be used for synthesizing actions with a given desired effect on an object or for recognizing the primitive and its effect based on the visual and even monocular observation of human movement.

HMMs [23] are a common tool for statistically representing movements as trajectories. However, they do not have the ability to model these trajectories in terms of an effect they should cause. If one needs to use HMM to recognize the pointing direction, one would have to use several HMMs, one HMM for each specific direction. PHMMs, on the other hand, are able to identify the pointing movement as well as direction of pointing.

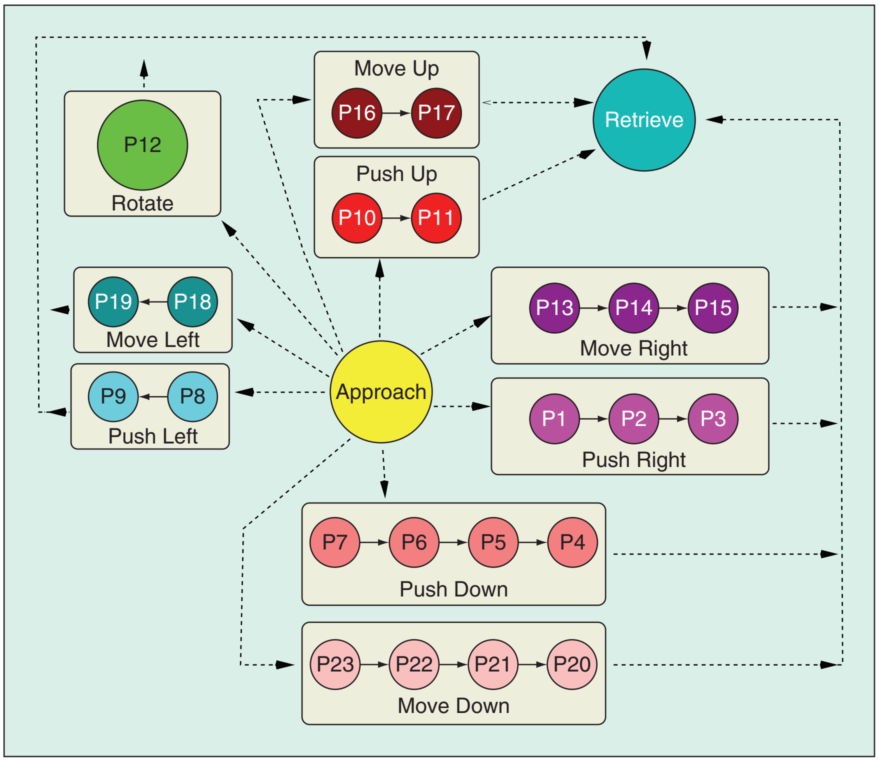


Figure 7. *The results for our data set. Each of the actions (recorded in sequence) starts by approaching the object, then it performs a manipulative primitive and ends by retrieving the hand. Each of the squares represents a primitive in the object state space and the corresponding primitive in $\mathcal{M}$.*

PHMMs are able to synthesize movements that are meant to generate a specific effect on the scene.

In the following sections, we introduce the PHMMs and discuss how they can be used for representing the action primitives such that they can be used for action recognition and synthesis.

The PHMM

PHMMs [1] are an extension of HMMs [23], [25] through latent parameters $\boldsymbol{\phi} = (\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_N)$, which model a systematic variation within each action class. We shortly review the main principles of HMMs and then extend it into PHMMs in the following sections.

The HMM

A HMM is a generative model. It is a finite state machine extended in a probabilistic manner. For a HMM $\boldsymbol{\lambda} = (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi})$, the vector $\boldsymbol{\pi} = (\pi_i)$ and transition matrix $\boldsymbol{A} = (a_{ij})$ define the prior state distribution of the initial states i and the transition probability between the hidden states. In continuous HMMs, the observation densities of each hidden state are described by density functions $b_i(\boldsymbol{x})$, which are in our context multivariate Gaussian densities $b_i(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$. The HMM parameters can be estimated through the Baum–Welch algorithm [23] for a set of training sequences.

Sequence Generation and Left–Right HMMs

An output sequence $\boldsymbol{X} = \boldsymbol{x}_1 \ldots \boldsymbol{x}_T$ can be drawn from the model by generating a step-by-step state sequence $\boldsymbol{Q} = q_1 \ldots q_T$ with respect to the initial probabilities π_i and the transition probabilities a_{ij} and, drawing for each state q_t, the output $\boldsymbol{x}_t$ from the corresponding observation distributions $b_i(\boldsymbol{x})$. Generally, there is no unique correspondence between an output sequence $\boldsymbol{X}$ and a state sequence, as different hidden state sequences can generate the same output sequence $\boldsymbol{X}$. This seems to be a rather poor approach to generate a good prototype movement from the model. To overcome this problem, we use a left–right model [23]. A good prototype can then be generated by taking the sequence means $\boldsymbol{\mu}_1 \ldots \boldsymbol{\mu}_T$ and interpolating between the means with respect to the expected time durations encoded in the state transitions.

The Parametric Extension to HMMs

A PHMM [1] $\boldsymbol{\lambda}^{\boldsymbol{\phi}}$ assumes that the training data can be modeled using observation densities $b_i(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ that are functions of a gesture parameter $\boldsymbol{\phi}$: $b_i^{\boldsymbol{\phi}}(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_i^{\boldsymbol{\phi}}, \boldsymbol{\Sigma}_i)$. In other words, for PHMMs, the means $\boldsymbol{\mu}_i^{\boldsymbol{\phi}} = \boldsymbol{f}_i(\boldsymbol{\phi})$ of the observation probability density functions (pdfs) $b_i^{\boldsymbol{\phi}}$ are functions of the parameter $\boldsymbol{\phi}$, and the functions $\boldsymbol{f}_i(\boldsymbol{\phi})$ are approximated for each state i separately in the training process. The dimensionality of $\boldsymbol{\phi}$ has to match the degree of freedom of the gesture. For example, for an approach action, the parameter $\boldsymbol{\phi}$ is given by the location of the object to be grasped. To model the approach action to location $\boldsymbol{\phi}$, all observation pdfs $b_i^{\boldsymbol{\phi}}$ of the PHMM are adapted appropriately. For a move action that starts at an initial object location A and ends at a final object location B, the parameter $\boldsymbol{\phi}$ contains the initial location A as well as the final location B.

In [1], a linear as well as a more general nonlinear model are used to model $\boldsymbol{f}_i(\boldsymbol{\phi})$. In the linear case, each function $\boldsymbol{f}_i(\boldsymbol{\phi})$ is of the form

$$\boldsymbol{\mu}_i = \bar{\boldsymbol{\mu}}_i + \boldsymbol{W}_i\boldsymbol{\phi}, \tag{4}$$

where the matrices $\boldsymbol{W}_i$ describe the linear variation $\boldsymbol{\phi}$ from the average gesture trajectory $\bar{\boldsymbol{\mu}}_i$. In the more general nonlinear case, a neural network, which is trained to approximate a more general nonlinear dependency on $\boldsymbol{\phi}$, is used for each state i. For both models, the training procedures are generally supervised and are similar to the classical Baum–Welch approach for HMMs, except for, e.g., in the linear case, the additional parameters $\boldsymbol{W}_i$ in (4) that must be estimated, and the values of $\boldsymbol{\phi}$ must be given for each training sequence. In our case, the parameters $\boldsymbol{\phi}$ are automatically provided through our unsupervised primitive learning approach from the "Unsupervised Learning of Action Primitive Classes According to the Effect on Objects" section. Recall that the result of the primitive learning approach was 1) sets of movements that all have the same effect on the object with respect to the chosen quantization of $\mathcal{O}$ and 2) how precisely the object states were changed by each of the movements. This means, while the movements in each equivalence class are used for training the PHMM, the initial and final object states for each of the movement are available and provide the necessary parametric data for PHMM training.

During primitive learning, the approach actions are sufficiently specified by the location of the object to be grasped, because at the beginning of each approach action, the human arm was always in a resting position in which it is simply hanging down from the shoulder. On the other hand, the move action was specified by two parameters given by the initial and final object positions. To use a single learning method for the two types of primitives, we changed the approach movement into a two-parameter movement. To do that, we used the hand location instead of object location: while the move action uses two object locations, we gain an additional location parameter at the beginning of the approach movement and an additional location parameter at the end of the remove movement by considering the hand location. This way, all primitives become biparametric, which allows us to use a single learning approach for all primitives.

To decide between a linear and nonlinear model, one can use the accuracy of the trained PHMM in resynthesizing the demonstrations as an indicator. In this article, we used a linear model to model the parametric variability between the movements, because the linear model was sufficiently precise to model the movements in our database. For learning the PHMM parameters, we used an extended version of Baum-Welch approach. In this work, the number of PHMM states was chosen through cross-validation. One possibility for choosing the number of states automatically is using the same number of states as used during the detection of action primitives.

Synthesis with PHMMs

The procedure for generating a particular sequence with parameter $\boldsymbol{\phi}$ is closely related to the method explained in the "Sequence Generation and Left–Right HMMs" section. The difference here is that one generates a specific movement for a parameterization $\boldsymbol{\phi}$: given $\boldsymbol{\phi}$, one first computes the means $\boldsymbol{\mu}_1^{\boldsymbol{\phi}} \dots \boldsymbol{\mu}_T^{\boldsymbol{\phi}}$ for the observation densities $b_i^{\boldsymbol{\phi}}$. This is done by evaluating the functions $f_i(\boldsymbol{\phi})$ that were learned in the training process ("The Parametric Extension to HMMs" section). Then, as all the observation densities b_i are specified, a good prototype can be synthesized as described in the "Sequence Generation and Left–Right HMMs" section. Once the parameter $\boldsymbol{\phi}$ is defined, the new means are calculated.

Recognition with HMMs and PHMMs

Let us assume that we have a set of action classes $\mathcal{K}$ and, for each class $k \in \mathcal{K}$, we trained a HMM $\boldsymbol{\lambda}_k$. The maximum likelihood approach can then be applied to identify $\boldsymbol{X}$ as the class k that maximizes the likelihood

$$k^{\mathrm{ML}} = \arg\max_i P(\boldsymbol{X}|\boldsymbol{\lambda}_i).$$

The likelihood can be efficiently computed using the forward–backward procedure [23]. In the case of parametric HMMs $\boldsymbol{\lambda}_k^{\boldsymbol{\phi}_k}$, the recognition becomes a two-step procedure. First, one estimates for each model $\boldsymbol{\lambda}_k^{\boldsymbol{\phi}}$ the parameterization $\boldsymbol{\phi}_k$ that explains the movement in the maximum likelihood sense best:

$$\boldsymbol{\phi}_k^{\mathrm{ML}} = \arg\max_{\boldsymbol{\phi}_i} P(\boldsymbol{X}|\boldsymbol{\lambda}_i^{\boldsymbol{\phi}_i}) \quad (\text{for each } k \in \mathcal{K}).$$

This can be done using expectation maximization (EM), as in [1], or with gradient descent, as in [2].

By adapting the parameters of the PHMM, this step reduces the PHMM to a normal HMM, and the next step becomes the same as in the case of general HMMs, i.e.,

$$k^{\mathrm{ML}} = \arg\max_i P(\boldsymbol{X}|\boldsymbol{\lambda}_i^{\boldsymbol{\phi}_i^{\mathrm{ML}}}).$$

Synthesizing Actions with PHMMs

In this section, we evaluate how precisely the PHMMs are able to synthesize movements and discuss how such movements can be concatenated.

Precision of Synthesized Actions from PHMMs

As explained earlier, PHMMs are able to synthesize movements that are meant to generate a specific effect on the scene. For example, to move an object from location L_1 to location L_2, the PHMM $\boldsymbol{\lambda}_{\mathrm{move}}^{\boldsymbol{\phi}}$ that was trained for the move action primitive is parameterized with the parameter $\boldsymbol{\phi} = (L_1, L_2)$. As discussed in the "Synthesis with PHMMs" section, the parameter $\boldsymbol{\phi}$ affects the observation densities $b_i^{\boldsymbol{\phi}}$ of the PHMM $\boldsymbol{\lambda}_{\mathrm{move}}^{\boldsymbol{\phi}}$ in such a way that the first observation pdf $b_1^{\boldsymbol{\phi}}$ assures that the hand is located at location L_1, the last observation pdf $b_{\mathrm{last}}^{\boldsymbol{\phi}}$ positions the hand at the final location L_2, and the observation pdfs between the first and the last one are parameterized appropriately to result in a smooth movement. In other words, the movement trajectory will start at location L_1 and end at location L_2, as these are hard constrained by the parameters.

The locations of the observation pdfs within the movement trajectory and the functions $\boldsymbol{\mu}_i = \bar{\boldsymbol{\mu}}_i + \boldsymbol{W}_i \boldsymbol{\phi}$ for each of the observation densities are specified during the learning process of the PHMM to assure an optimal approximation of the training data in the least-squares sense. The interpolation of the movement trajectory between the observation pdfs is done linearly.

We have investigated how the quality of the action synthesis depends on the number of training movements for the PHMM and also how the synthesis quality depends on the location of the object. The following two experimental results for the approach movement are representative results in our set of experiments: we have hand reduced the equivalence class for the approach movement so that it contained 1) four repetitions of approach movements to each of the four corners of the table (2×2 grid) and 2) four repetitions of approach movements to each of the four table corners and the middle of the table sides (3×3 grid). Based on these reduced equivalence classes, we trained the approach PHMM $\boldsymbol{\lambda}_{\mathrm{approach}}^{\boldsymbol{\phi}}$ with 40 hidden states.

For testing, we synthesized movements for evenly distributed 5×7 locations on the table. For each of these locations, four repetitions of the true human movements were available in our movement data set. The table top has a size of 80×30 cm.

We calculated the error for each of the 5×7 as the distance between the synthesized movement $\boldsymbol{f}(t)$ and the averaged trajectory $\overline{\boldsymbol{f}}(t)$ of the four available human performances.

Each human arm movement in our data set is specified by six 3-D trajectories $\boldsymbol{f}(t) = (\boldsymbol{f}_i(t))_{i=1}^6$, as mentioned in the "Building PHMMs for Each Primitive Class" section. The discrepancy ε between the synthesized movement $\boldsymbol{f}(t)$ and example movement, which was not used for training but started and ended at the same location, is calculated as the root-mean-square error between the synthesized movement and averaged human movement $\overline{\boldsymbol{f}}(t)$

$$\varepsilon = \sqrt{\frac{1}{6}\sum_{i=1}^{6} \int (\boldsymbol{f}_i(\alpha(t)) - \overline{\boldsymbol{f}}_i(\overline{\alpha}(t)))^2 dt \Big/ \int \alpha(t) dt}, \qquad (5)$$

where $\alpha(t)$ and $\overline{\alpha}(t)$ are warping functions.

The results are summarized for the 2×2 training grid in Figure 8. For the 3×3 training grid, the results are very similar to the 2×2 grid. The synthesis errors are approximately 1.8 cm. We have measured the root-mean-square error of the human performances compared to the average human

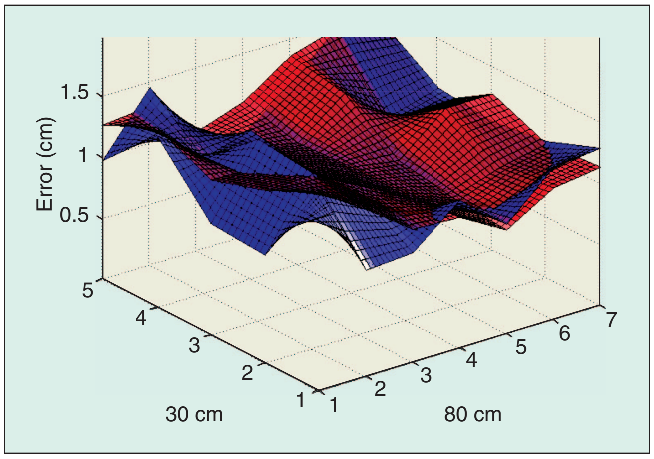


Figure 8. The image shows the root-mean-square error for the synthesized grasping movement (red), where the PHMM was trained on 2×2 training movements, and for the human reference performance (blue).

performance. The result is plotted in Figure 8. By comparing the two plots in Figure 8, one can see that the performance of the PHMM is comparable with the human performance.

Concatenating Action Primitives into Complex Actions Using PHMMs

If actions are defined in terms of action primitives and action grammars, one has to concatenate the primitives to synthesize complex actions, e.g., on a robot. We have investigated the concatenation of primitives in a robot experiment. In this experiment, a humanoid robot was meant to grasp objects on a table and insert these objects into specific holes on the table (see Figure 9). In our experimental setup, the object states were given by their two-dimensional (2-D) locations on a table, and the 2-D locations were identified through a camera above the table. To generate movements on the robot, we used the three action primitives: approach, move, and remove, as learned from our primitive learning approach and represented by our PHMMs. For robot control, the 3-D locations of the synthesized arm trajectories $\boldsymbol{f}(t) = \{\boldsymbol{f}_i(t)\}$ (see the "Building PHMMs for Each Primitive Class" section) are mapped to joint angles. To meet the scale of the robot, the marker locations are rescaled. The joint angles are then calculated as least-squares solution with the constraint that the end effector is at the location that is required to achieve the effect $\boldsymbol{\phi}$.

The robot task was a priori specified as an approach–move–remove movement. During the execution, a human supervisor advised the robot by pointing toward an object and toward the hole into which the object should be dropped. These two positions were estimated by vision. Using the initial robot hand position P, the estimated object position S, and the estimated location E of the hole, the robot generated and executed the corresponding approach PHMM $\boldsymbol{\lambda}^{\phi_1}_{\text{approach}}$ with $\boldsymbol{\phi}_1 = (P, S)$, the move PHMM $\boldsymbol{\lambda}^{\phi_2}_{\text{move}}$ with $\boldsymbol{\phi}_2 = (S, E)$, and finally the remove PHMM $\boldsymbol{\lambda}^{\phi_3}_{\text{remove}}$ with $\boldsymbol{\phi}_3 = (E, P)$. The continuity of the motion was ensured by specifying the end position parameter of the previous PHMM as the starting position parameter of the next PHMM. The results show (see Figure 9) that the generated movements are accurate enough to grasp the object and put it into the hole. Implementational details can be found in [26].

Recognition of Humans' Actions

In this section, we discuss how to recognize an action primitive and its intended effect from a visual monocular observation. In detail, given a monocular visual observation of a human action, we need to identify which PHMM models this observation best and which parameterization it uses. From the PHMMs and its parameterization, we can identify the primitive and, given the action grammar from the "Unsupervised Learning of Action Primitive Classes According to the Effect on Objects," section, we are able to recognize complex actions.

The most common approach to action recognition is to assume that a person can be tracked and that the 3-D postures of the person can be estimated. Once the time sequence of 3-D body postures has been estimated, the sequence is evaluated, commonly with a HMM [5] or possibly even with a PHMM, as discussed in the "The PHMM" section. However, nonintrusive full-body motion capture is neither a trivial nor a solved problem [5], [27], [28]. In this section, we propose a completely novel, context- and object-driven approach, where we acknowledge the fact that tracking, action recognition, and scene context are intertwined, i.e., the objects and context constrain the actions and vice versa.

Recognition Through Tracking in Action Space

Instead of 3-D tracking approach with a subsequent action recognition, we use our PHMMs to estimate directly which action we are observing and which parameterization it has.

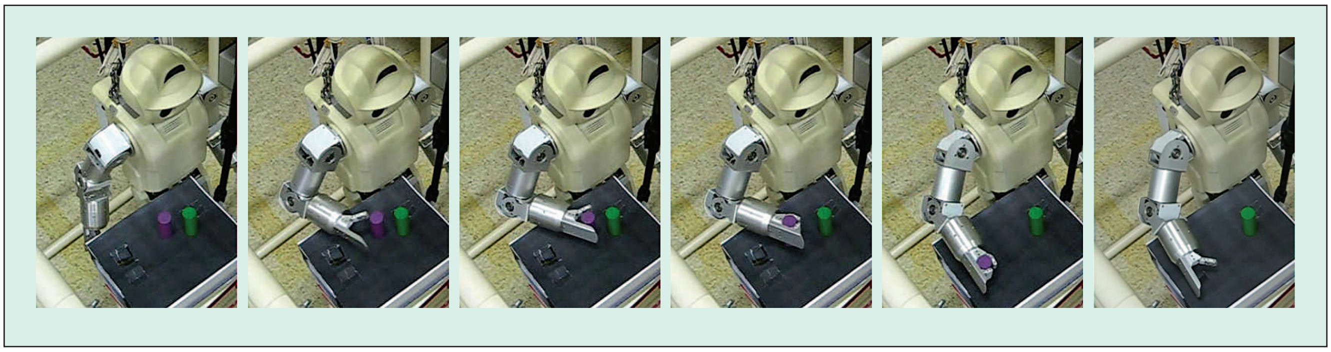


Figure 9. The humanoid HOAP-3 grasps a specified object and drops it into a specified hole on the table.

As a consequence, while general 3-D human body tracking needs to solve the parameter estimation problem in a very high-dimensional parameter space of human joint configuration, our approach to action recognition is concerned with a much smaller parameter space that is defined by the conditional density over the possible actions and parameters, given the context and objects in that context. Since our PHMMs are generative models, we can generate the corresponding 3-D postures from the PHMMs and deduce the 3-D human pose in the scene.

We call this approach tracking in action space, and we define the action space as given by the aforementioned conditional density. For example, in case of an approach action ("The PHMM" section), the action space is essentially described by the object location parameter, thus for tracking the approach action, it is in principle sufficient to test for different location parameters instead of searching in a high-dimensional action space. But even that can be simplified, because we can constrain these parameters if we are able to identify the object locations in the scene.

To explain our framework in detail, let us consider the classical Bayesian propagation over time, as it is often used in the context of general 3-D human body tracking [5]:

$$p_t(\boldsymbol{\omega}_t) \propto \int P(\boldsymbol{I}_t|\boldsymbol{\omega}_t)P(\boldsymbol{\omega}_t|\boldsymbol{\omega}_{t-1})p_{t-1}(\boldsymbol{\omega}_{t-1})d\boldsymbol{\omega}_{t-1}, \qquad (6)$$

where $\boldsymbol{I}_t$ is the current visual observation, $p_t(\boldsymbol{\omega}_t)$ the pdf for the random variable $\boldsymbol{\omega}_t$ at time t, $P(\boldsymbol{\omega}_t|\boldsymbol{\omega}_{t-1})$ the propagation step, and $P(\boldsymbol{I}_t|\boldsymbol{\omega}_t)$ the likelihood measurement of $\boldsymbol{I}_t$, given $\boldsymbol{\omega}_t$. Typically, the random variable $\boldsymbol{\omega}_t$ specifies the body configuration of a human model in joint angles, and the propagation density is used to constrain the random variable $\boldsymbol{\omega}_t$ to the most likely pose values at each time step t [29], [30]. To compute the likelihood $P(\boldsymbol{I}_t|\boldsymbol{\omega}_t)$, a human body model is generated using the pose values from $\boldsymbol{\omega}_t$ and then compared with the input image data $\boldsymbol{I}_t$. For evaluating (6), one commonly uses a particle-based approach [5], [27], [31], [32].

In the tracking of action-space approach, the random variable $\boldsymbol{\omega}$ is given as $\boldsymbol{\omega} = (a, \boldsymbol{\phi}, \tau)$, and it is used to control our PHMMs: the parameter a identifies which PHMM it is, $\boldsymbol{\phi}$ specifies the parameters of $\boldsymbol{\lambda}_a^{\boldsymbol{\phi}}$ that need to be estimated, and τ is the timing parameter that specifies the current hidden state within the PHMM.

The propagation density $P(\boldsymbol{\omega}_t|\boldsymbol{\omega}_{t-1})$ can be considerably simplified. If we assume that a human finishes one action primitive before starting a new one, the action identifier a is constant until an action primitive is finished. The timing parameter τ changes according to the transition matrix of the HMM, and the parameter $\boldsymbol{\phi}$ can also be assumed to be roughly constant until a new action primitive is started.

Finally, the likelihood $P(\boldsymbol{I}_t|\boldsymbol{\omega}_t) = P(\boldsymbol{I}_t|(a, \boldsymbol{\phi}, \tau)_t)$ is computed by first using the ath PHMM to generate the joint angles of the 3-D human body pose for the parameter $\boldsymbol{\phi}$ and HMM state τ. In the second step, the generated joint angles are used together with a 3-D body model to compute a projection of the body onto the image plane, which we then compare with the input image $\boldsymbol{I}_t$. When computing the observation

Tracking in action space is the application of action primitives for tracking.

likelihood, we also make use of the standard deviations of observation densities of the PHMM.

Action Tracking: PHMM-Based

In this section, we discuss the details of using PHMMs to model the actions for action tracking. In our problem scenario, we assume to have a set $\mathcal{A} = \{1, \ldots, \mathcal{M}\}$ of actions, where for each action $a \in \mathcal{A}$, a PHMM $\boldsymbol{\lambda}_a^{\boldsymbol{\phi}}$ was trained.

In the "Sequence Generation and Left–Right HMMs" and "Synthesis with PHMMs" sections, we have discussed how to generate a sequence from a PHMM $\boldsymbol{\lambda}_a^{\boldsymbol{\phi}}$ for action a and parameter $\boldsymbol{\phi}$. Hence, for a given $\boldsymbol{\omega}_t = (a, \boldsymbol{\phi}, \tau)$, $P(\boldsymbol{x}|\boldsymbol{\omega}) = b_{a,\tau}^{\boldsymbol{\phi}}(\boldsymbol{x})$ defines the distribution of joint angles of 3-D body poses for which $b_{a,\tau}^{\boldsymbol{\phi}}(\boldsymbol{x})$ generates a corresponding 3-D human body model [see Figure 10(a)], which is then matched against the input image $\boldsymbol{I}_t$:

$$P(\boldsymbol{I}_t \mid \boldsymbol{\omega}_t) = \int_x P(\boldsymbol{I}_t \mid \boldsymbol{x})P(\boldsymbol{x} \mid \boldsymbol{\omega}_t)dx. \qquad (7)$$

Finally, the propagation density $P(\boldsymbol{\omega}_t|\boldsymbol{\omega}_{t-1})$ is given as follows: τ is propagated as mentioned earlier by the transition matrix $\boldsymbol{A}_a$ of PHMM $\lambda_a^{\boldsymbol{\phi}}$, and we allow $\boldsymbol{\phi}$ to change according to a Gaussian distribution (Brownian motion [33]). The variable a that specifies the action primitive is initially drawn from a context-dependent distribution and is allowed to change according to the grammar computed in the "Unsupervised Learning of Action Primitive Classes According to the Effect on Objects" section.

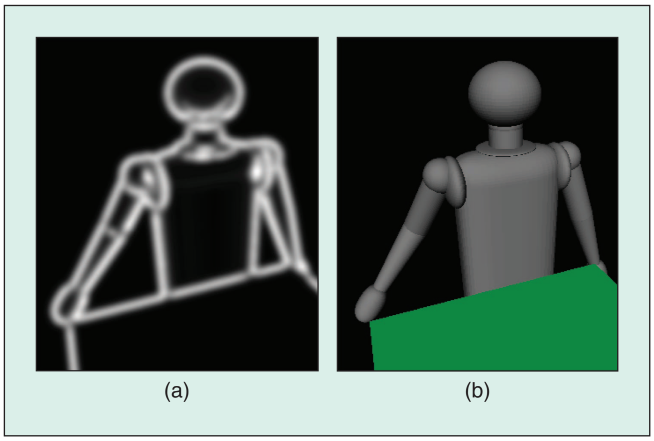


Figure 10. (a) We use an articulated human model, where the skeletal structure is fleshed out by cones and superquadratics. Each shoulder and elbow pair has four degrees of freedom. (b) The edge image (model itself) is a smoothed gradient image, serving as a distance to edge image.

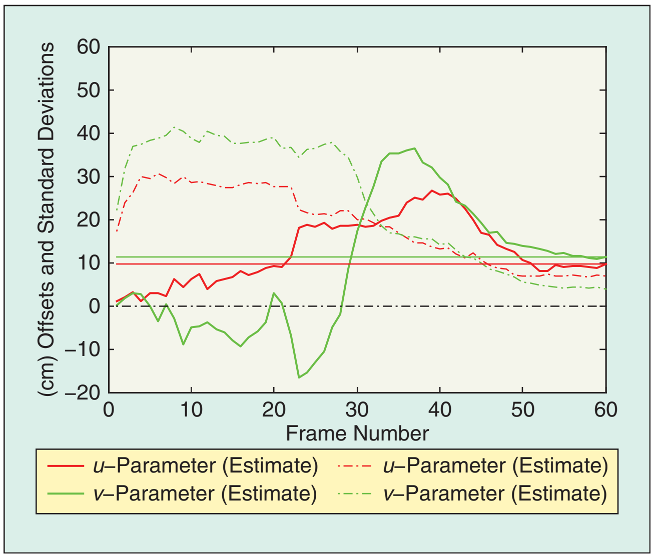


Figure 11. *The figure shows the progression of the current estimate of the action parameters u, v over time, where (u, v) defines the pointed at position on table top and is measured as an offset in centimeter to the center of the active table-top region. In the images of Figure 12, the variables correspond to the horizontal (u) and vertical direction (v). The dotted lines show the standard deviation for u and v.*

It should be noted that, for all action primitives and its corresponding PHMMs, the initial parameters are known and given by the present tracking state. Only the final parameter needs to be estimated. For example, the move primitive starts at the present location of the hand that immediately defines the first parameter for the move PHMM $\boldsymbol{\lambda}^{\phi}_{\text{move}}$ and it must, according to our grammar in "Unsupervised Learning of Action Primitive Classes According to the Effect on Objects" section, coincide with the final parameter of the approach primitive.

It is worth having a close look at the estimation process for $\boldsymbol{\omega}_t$: the entropy of the density p_t reflects the uncertainty of the detected parameters. The entropy usually decreases with every new incoming image, and we use it as a measure of convergence of the parameter estimation process.

Furthermore, by marginalizing over $\boldsymbol{\phi}$ and τ, we can compute the likelihood of each action a, and by marginalizing over a and τ, we can also compute the pdf of the action parameters $\boldsymbol{\phi}$. Figure 11 shows the progression over time for the unknown parameters of the approach action. The red and green lines show the most likely 2-D location parameters u and v [for $\boldsymbol{p} = (u, v)$]. The dotted lines show their corresponding uncertainties. The horizontal thin lines mark the corresponding correct values for u and v. As one can see, the uncertainty decreases with time, and after approximately 60 frames, the correct parameters are recovered. This is about the time when the arm is fully stretched.

The Observation Model

We use an articulated model of the human body [Figure 10(a)]. The computation of the observation likelihood is based on the edge information of the arm silhouette. Therefore, the contour $\mathcal{C}$ of the projected articulated body model is extracted from the rendered view for a pose $\boldsymbol{x}$. We defined the observation function similar to a method described in [32] on a smoothed-edge image [see Figure 10(b)], where the pixel values are interpreted as distances to the nearest edge. The edge distance image is calculated as follows. We calculate a normalized gradient image of the observed image $\boldsymbol{I}$, where the gray values above some threshold are set to 1. The image is finally smoothed with a Gaussian mask and normalized. This edge image is denoted by G. The value of $1 - G(\boldsymbol{c})$ of a contour pixel $\boldsymbol{c}$ can then be interpreted as distance values between 0 and 1, where the value 1 corresponds to a pixel with no edge in the vicinity and 0 corresponds to a pixel on a strong edge. This distance interpretation is in some sense similar to the edge detection along the normal, as used in [31], but faster to evaluate.

The observation function is computed as

$$P(\boldsymbol{I} \mid \boldsymbol{x}) = \exp\left\{ -\frac{1}{2\sigma^2}\frac{1}{|\mathcal{C}|}\sum_{p\in\mathcal{C}} (1 - G(p))^2 \right\}, \tag{8}$$

where $\mathcal{C}$ is the model's contour and G is the edge image. An extension to multiple camera views is straightforward:

$$P(\boldsymbol{I} \mid \boldsymbol{x}) = \exp\left\{ -\frac{1}{2\sigma^2}\sum_i \frac{1}{|\mathcal{C}_i|}\sum_{p\in\mathcal{C}_i} (1 - G_i(p))^2 \right\}, \tag{9}$$

where $\mathcal{C}$ and G_i are the corresponding contour sets and edge images of each view i.

Experiments

We evaluated our approach on a monocular video data of the same arm actions as in our movement data set [12], as described in the "Unsupervised Learning of Action Primitive Classes According to the Effect on Objects" section. Our testing data is, however, different from the training data. The scenario (actor and table top) in which the actions are performed can be seen in the tracked sequence, Figure 12. As an action model, we used linear PHMMs $\boldsymbol{\lambda}^{\phi}_{a}$ with $\boldsymbol{\phi} = (u, v)$, as trained in the "The PHMM" section.

The propagation over time is performed as described in the "Action Tracking: PHMM-Based" section. We decrease the diffusion of the Brownian motion of u and v in dependence of the state number τ. Our argument for the cooling down of the Brownian motion over time is that, for the first frames, the visual evidence for the right position $\boldsymbol{\phi} = (u, v)$ is very weak, which means that we should allow a large variety of possible (u, v). But as visual evidence increases with time, we become more certain about the correct (u, v), and we can successively reduce the variance.

The sampling and normalization of image observations are performed as in [31]. As described in the "The Observation Model" section, the observation function is based on the evaluation of the edge information in the monocular video data and the human body model for state $\boldsymbol{x}_{\omega_{ti}}$.

The images in Figure 12 show that the arm pose is (visually) very accurately estimated. The following three factors emphasize the capabilities of our tracking in action-space approach: all information are gathered from a monocular video based on a single feature type (edge information). The edge images (especially, the first part of the sequence) contain a lot of clutter, and the silhouette of the arm is not segmented accurately. Besides posture estimation, one can see in Figure 12 that the estimation of the action parameters (corresponding to the position indicated through the small red dot) converges in this example slowly to the true parameters of the action when the arm approaches the table top: as one can see in Figure 12, there is only little visual evidence in the middle of the sequence about the pointing location that causes the convergence to be slow in this example.

To evaluate the quality of posture estimation, we recorded the joint positions in parallel with a marker-based motion capture system. The root-mean-square error of the three joints positions (shoulder, elbow, and finger) over the whole sequence, as shown in Figure 12, is 3.3 cm. The component-wise averaged error is only 0.4 cm. This error was within the natural human action variation, as it was observable in the training data. During our experiments, we have used 400 particles, and we have reached a processing speed of 10 Hz on an average PC with an average graphics card.

Conclusions

We have presented a complete framework for learning action primitives and for representing them with PHMMs for synthesizing and recognizing actions. The learning algorithm was tested using a set of actions on objects. We are able to recover a simple primitive structure for the actions that are similar to the natural language description for the actions we have considered. Primitive-based modeling of actions enables us to define a hierarchy of actions by converting continuous observations into discrete symbols. Several authors have represented actions in a hierarchical manner [34]–[36]. These works require the manual modeling of atomic movements/primitives. The contribution of our work is that we perform this segmentation automatically and unsupervised, while the entire learning process is governed by only two parameters, the distance between the Gaussians and the threshold for (1). Even though we have based our discussion on the data from [12], tests with other data sets suggest that our approaches generalize well. Important further investigations will concern the quantization of the object state space and possible different generalization levels.

The experimental results from [37] suggest that action perception and execution of motor primitives are connected through objects. There are also further studies from experimental psychology that confirm the role of objects in action understanding [38], [39]. In this article, we have exploited object information to learn action primitives.

Even though object detection and classification literature are quite large (for overview, see [40]), there are not many attempts to combine it with action modeling [41], [42]. In [41], HMMs are combined with object context to classify hand actions. Image-, object-, and action-based evidence were used to label and summarize activity and also to identify objects. They define a generalized class model to describe objects. Actions associated with each class were represented using trained HMMs. The states of such HMMs were connected to the regions through which the object moved for a particular action. Our approach learns such a model for modeling actions automatically. A Bayesian model was used in [42] for modeling human–object interactions. Some of the conditional probabilities of this model were calculated using trained

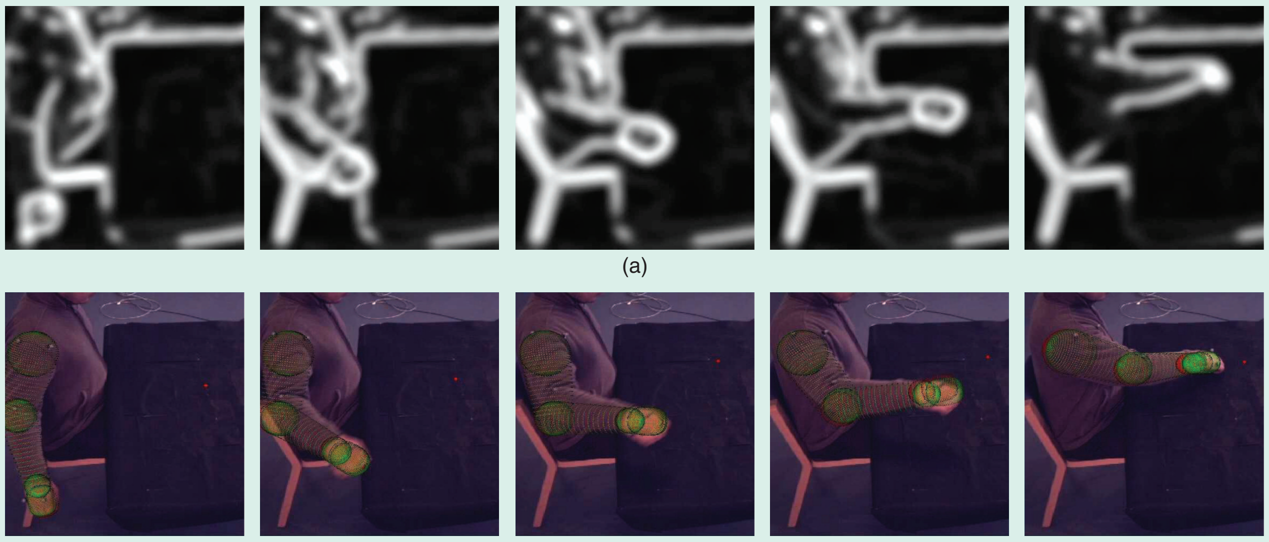


Figure 12. *The images show a person reaching for a certain position on the table. (a) The edge image for the likelihood measurement and (b) the corresponding camera image with the detected arm pose superimposed. The current estimate of the position is indicated by the red point on the table.*

Primitive-based modeling of actions enables us to define a hierarchy of actions by converting continuous observations into discrete symbols.

HMMs. These approaches require a good initial training of action models for later recognition even though a known structure is assumed. Our work goes beyond the state of the art in this area, since it exploits object knowledge in the primitive learning process.

Our work relates to the recent work of [11], where a hierarchical tree structure is incrementally formed representing the motions learned by the robot. One of the issues raised is that each node representing a motion primitive may differ from those segmented in an off-line, supervised process. By integrating the object knowledge in the learning process, the resulting primitives are more similar to the ones generated in an off-line process.

Other robotics researchers used HMMs to represent movement primitives [43]–[45]. Unlike these approaches, we suggest a parametric version of HMMs, which are called PHMMs. PHMMs are essential for the generation of robot movements that can be applied to objects.

One might argue that our approach cannot be used for general action synthesis and recognition, because the space of possible actions will always be too limited.

However, following the arguments in [7], [8], [10], [46], and [47] that human actions are composed of motor primitives similar to human speech being composed out of phonemes, we believe that our limited action space can be generalized to span the space of action primitives. Stochastic action grammars could be used as in [10], [47], and [48] to model more complex actions. Furthermore, [47] explains how a language for human actions can be generated based on grounded concepts, kinetology, morphology, and syntax.

Acknowledgments
This work was partially supported by EU through grants PACO-PLUS and FP6-2004-IST-4-27657.

Keywords
Action grammars, action primitives, human body tracking, motor primitives, parametric hidden Markov models.

References

[1] A. D. Wilson and A. F. Bobick, "Parametric hidden Markov models for gesture recognition," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 21, no. 9, pp. 884–900, 1999.

[2] D. Herzog, V. Krueger, and D. Grest, "Parametric hidden Markov models for recognition and synthesis of movements," in *Proc. British Machine Vision Conf.*, Leeds, U.K., Sept. 1–4, 2008, pp. 163–172.

[3] C. Breazeal and B. Scassellati, "Robots that imitate humans," *Trends Cogn. Sci.*, vol. 6, no. 11, pp. 481–487, 2002.

[4] A. Billard, Y. Epars, S. Calinon, S. Schaal, and G. Cheng, "Discovering optimal imitation strategies," *Robot. Autonom. Syst.*, vol. 47, no. 2–3, pp. 69–77, 2004.

[5] V. Krüger, D. Kragic, A. Ude, and C. Geib, "The meaning of action: A review on action recognition and mapping," *Adv. Robot.*, vol. 21, no. 13, pp. 1473–1501, 2007.

[6] J. Kober, B. Mohler, and J. Peters, "Learning perceptual coupling for motor primitives," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, 2008, pp. 834–839.

[7] G. Rizzolatti, L. Fogassi, and V. Gallese, "Neurophysiological mechanisms underlying the understanding and imitation of action," *Nat. Rev.*, vol. 2, pp. 661–670, Sept. 2001.

[8] G. Rizzolatti, L. Fogassi, and V. Gallese, "Parietal cortex: From sight to action," *Curr. Opin. Neurobiol.*, vol. 7, no. 4, pp. 562–567, 1997.

[9] F. Mussa-Ivaldi and E. Bizzi, "Motor learning through the combination of primitives," *Phil. Trans. R. Soc. London B*, vol. 355, no. 1404, pp. 1755–1769, 2000.

[10] G. Guerra-Filho and Y. Aloimonos, "A sensory-motor language for human activity understanding," in *Proc. 6th IEEE-RAS Int. Conf. Humanoid Robots*, Genova, Italy, Dec. 4–6, 2006.

[11] D. Kulic and Y. Nakamura, "Scaffolding on-line segmentation of full body human motion parameters," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, 2008, pp. 2860–2866.

[12] I. S. Vicente, V. Kyrki, and D. Kragic, "Action recognition and understanding through motor primitives," *Adv. Robot.*, vol. 21, no. 15, pp. 1687–1707, 2007.

[13] S. Baby and V. Krueger, "Primitive based action representation and recognition," in *Proc. Scandinavian Conf. Image Analysis*, Oslo, Norway, June 15–18, 2009, pp. 31–40.

[14] P. Fitzpatrick, G. Metta, L. Natale, S. Rao, and G. Sandini, "Learning about objects through action-initial steps towards artificial cognition," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, Sept. 2003, vol. 3, pp. 3140–3145.

[15] S. Griffith, J. Sinapov, M. Miller, and A. Stoytchev, "Toward interactive learning of object categories by a robot: A case study with container and non-container objects," in *Proc. IEEE 8th Int. Conf. Development and Learning (ICDL)*, June 2009, pp. 1–6.

[16] L. Montesano, M. Lopes, A. Bernardino, and J. Santos-Victor, "Learning object affordances: From sensory-motor coordination to imitation," *IEEE Trans. Robot.*, vol. 24, no. 1, pp. 15–26, Feb. 2008.

[17] H. Kozima, C. Nakagawa, and H. Yano, "Emergence of imitation mediated by objects," in *Proc. 2nd Int. Workshop on Epigenetic Robotics*, 2002, pp. 59–61.

[18] M. Dogar, M. Cakmak, E. Ugur, and E. Sahin, "From primitive behaviors to goal-directed behavior using affordances," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2007, pp. 729–734.

[19] M. Lopes and J. S. Victor, "Visual learning by imitation with motor representations," *IEEE Trans. Syst., Man, Cybern.*, vol. 35, no. 3, pp. 438–449, June 2005.

[20] J. Sinapov and A. Stoytchev, "Detecting the functional similarities between tools using a hierarchical representation of outcomes," in *Proc. 7th IEEE Int. Conf. Development and Learning, (ICDL)*, Aug. 2008, pp. 91–96.

[21] H.-K. Lee and J. Kim, "An HMM-based threshold model approach for gesture recognition," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 21, no. 10, pp. 961–973, Oct. 1999.

[22] D. Kulić, W. Takano, and Y. Nakamura, "Incremental learning, clustering and hierarchy formation of whole body motion patterns using adaptive hidden Markov chains," *Int. J. Robot. Res.*, vol. 27, no. 7, pp. 761–784, 2008.

[23] L. R. Rabiner and B. H. Juang, "An introduction to hidden Markov models," *IEEE ASSP Mag.*, vol. 3, no. 1, pp. 4–15, Jan. 1986.

[24] D. S. Hirschberg, "Algorithms for the longest common subsequence problem," *J. ACM*, vol. 24, no. 4, pp. 664–675, 1977.

[25] X. Huang, Y. Ariki, and M. Jack, *Hidden Markov Models for Speech Recognition*. New York, NY, USA: Columbia Univ. Press, 1990.

[26] D. Herzog, A. Ude, and V. Krueger, "Motion imitation and recognition using parametric hidden Markov models," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots (Humanoids)*, Daejeon, South Korea, Dec. 1–3, 2008, pp. 339–346.

[27] T. Moeslund, A. Hilton, and V. Krueger, "A survey of advances in vision-based human motion capture and analysis," *Comput. Vis. Image Understand.*, vol. 104, no. 2–3, pp. 90–127, 2006.

[28] M. Lee and R. Nevatia, "Human pose tracking in monocular sequences using multilevel structured models," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 31, no. 1, pp. 27–38, 2009.
[29] H. Moon, R. Chellappa, and A. Rosenfeld, "3d object tracking using shape-encoded particle propagation," in *Proc. Int. Conf. Computer Vision*, Vancouver, Canada, July 9–12, 2001, vol. 2, pp. 307–314.
[30] J. Gall, J. Patthoff, C. Schnoerr, B. Rosenhahn, and H.-P. Seidel, "Interacting and annealing particle filters: Mathematics and recipe for applications," *J. Math. Imag. Vis.*, vol. 28, no. 1, pp. 1–18, May 2007.
[31] M. Isard and A. Blake, "Condensation—Conditional density propagation for visual tracking," *Int. J. Comput. Vis.*, vol. 29, no. 1, pp. 5–28, 1998.
[32] J. Deutscher, A. Blake, and I. Reid, "Articulated body motion capture by annealed particle filtering," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2000, vol. 2, pp. 126–133.
[33] D. MacKay, "Introduction to Monte Carlo methods," *Learning in Graphical Models*, M. Jordan, Ed. Cambridge, MA: MIT Press, 1999, pp. 175–204.
[34] A. Bobick, "Movement, activity, and action: The role of knowledge in the perception of motion," *Philos. Trans. Royal Soc. Lond.*, vol. 352, no. 1358, pp. 1257–1265, 1997.
[35] C. Stauffer and W. Grimson, "Learning patterns of activity using real-time tracking," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 22, no. 8, pp. 747–757, 2000.
[36] N. Robertson and I. Reid, "Behaviour understanding in video: A combined method," in *Proc. Int. Conf. Computer Vision*, Beijing, China, Oct. 15–21, 2005, pp. 808–815.
[37] V. Gallese, L. Fadiga, L. Fogassi, and G. Rizzolatti, "Action recognition in the premotor cortex," *Brain*, vol. 119, no. 2, pp. 593–609, 1996.
[38] K. Nelissen, G. Luppino, W. Vanduffel, G. Rizzolatti, and G. A. Orban, "Observing others: Multiple action representation in the frontal lobe," *Science*, vol. 310, no. 5746, pp. 332–336, 2005.
[39] B. N. Daniel and Michael E. J. Masson, "Gestural knowledge evoked by objects as part of conceptual representations," *Aphasiology*, vol. 20, no. 9-11, pp. 1112–1124, Nov. 2006.
[40] S. Ullman, *High-Level Vision: Object Recognition and Visual Cognition*. Cambridge, MA: MIT Press, July 1996.
[41] D. Moore, I. Essa, I. Hayes, and M. H., "Exploiting human actions and object context for recognition tasks," in *Proc. 7th IEEE Int. Conf. Computer Vision*, 1999, vol. 1, pp. 80–86.
[42] A. Gupta and L. Davis, "Objects in action: An approach for combining action understanding and object perception," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition, 2007 (CVPR '07)*, June 2007, pp. 1–8.
[43] T. Inamura, I. Toshima, H. Tanie, and Y. Nakamura, "Embodied symbol emergence based on mimesis theory," *Int. J. Robot. Res.*, vol. 23, no. 4-5, pp. 363–377, 2004.
[44] A. Billard, S. Calinon, and F. Guenter, "Discriminative and adaptive imitation in uni-manual and bi-manual tasks," *Robot. Autonom. Syst.*, vol. 54, no. 5, pp. 370–384, 2006.
[45] T. Asfour, F. Gyarfas, P. Azad, and R. Dilmann, "Imitation learning of dual-arm manipulation tasks in humanoid robots," *Int. J. Humanoid Robot.*, vol. 5, no. 2, pp. 183–202, 2008.
[46] O. Jenkins and M. Mataric, "Deriving action and behavior primitives from human motion data," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, Lausanne, Switzerland, Sept. 30–Oct. 42002, pp. 2551–2556.
[47] G. Guerra-Filho and Y. Aloimonos, "A language for human action," *IEEE Comput. Soc.*, vol. 40, pp. 42–51, May 2007.
[48] Y. Ivanov and A. Bobick, "Recognition of visual activities and interactions by stochastic parsing," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 22, no. 8, pp. 852–872, 2000.

Volker Krüger received his Dipl.-Inf. degree and doctor's degree from Christian-Albrechts-Universität (CAU) Kiel, Germany, in 1997 and 2000, respectively. He was a postdoctoral fellow at the Center for Automation Research at University of Maryland from 2000 to 2002. Since 2002, he has been an associate professor at Aalborg University in Maryland. He is with the Computer Vision and Machine Intelligence Lab (CVMI) at the Copenhagen Institute of Technology (CIT) of Aalborg University. He is a Member of the IEEE. His research focuses on computer vision-based approaches for learning and recognizing human actions.

Dennis L. Herzog received his master's degree in computer science from CAU in 2006. He is currently a Ph.D. student with the CVMI at CIT, Copenhagen. His research activities include the representation of parametric human movements, vision-based tracking of human movements, and the synthesis of human movements for robot control.

Sanmohan Baby received his M.Sc. degree in mathematics from the Indian Institute of Technology, Madras, India. He is currently pursuing his Ph.D. degree in computer science and engineering at Aalborg University, Denmark, in the area of computer vision. His research interests are in the areas of computer and robot vision, especially in finding a grammatical description of human actions for imitation learning in robots.

Aleš Ude studied applied mathematics at the University of Ljubljana, Slovenia, and received his doctoral degree from the Faculty of Informatics, University of Karlsruhe, Germany. He was awarded the STA fellowship for postdoctoral studies in ERATO Kawato Dynamic Brain Project, Japan. He has been a visiting researcher at ATR Computational Neuroscience Laboratories, Kyoto, Japan, for a number of years and is still associated with this group. Currently, he is a senior researcher at the Department of Automatics, Biocybernetics, and Robotics, Jožef Stefan Institute, Ljubljana, Slovenia. He is a Member of the IEEE. His research focuses on imitation and action learning, perception of human activity, humanoid robot vision, and humanoid cognition.

Danica Kragic is a professor of computer science at the School of Computer Science and Communication at KTH and acting director of the interdisciplinary research at Centre for Autonomous Systems. In 2007, she received the 2007 IEEE Robotics and Automation Society (RAS) Early Academic Career Award. In 2008, she received the Swedish Foundation for Strategic Research Future Research Leaders Award. Since 2006, she has been the chair for the RAS Technical Activity Board for Computer and Robot Vision and a member of the IEEE RAS Conference Editorial Board. She is a Member of the IEEE. Her research interests include development of vision systems and vision-based control; human–robot interaction; and collaboration, object grasping, and manipulation.

Address for Correspondence: Volker Krüger, CVMI Lab, Copenhagen Institute of Technology, Aalborg University, Denmark. E-mail: vok@cvmi.aau.dk.