

Project no.: 027657

Project full title: Perception, Action & Cognition through learning of Object-Action Complexes

Project Acronym: PACO-PLUS

Deliverable no.: D3.2.4

Title of the deliverable: Human grasping and upper body actions in OAC space

Contractual Date of Delivery to the CEC:	31/01-2009	
Actual Date of Delivery to the CEC:	04/02-2009	
Organisation name of lead contractor for this deliverable:	KTH	
Author(s):	Hedvig Kjellström, Volker Krüger, Danica Kragic, Tamim Asfour	
Participant(s):	KTH, AAU, KIT	
Work package contributing to the deliverable:	WP3.1, WP3.2	
Nature:	R	
Version:	1.0	
Total number of pages:	5	
Start date of project:	1 st Feb. 2006	Duration: 48 month

Project co-funded by the European Commission within the Sixth Framework Programme (2002–2006)
Dissemination Level

PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Abstract:

In this report, we present the work on 1) object-centred representation of human action 2) learning grammars for human manipulation actions from human demonstration 3) 3D articulated hand and upper body tracking, carried out within WP3.1 and WP3.2.

Keyword list: Learning OACs, Recognizing OACs, Learning from Demonstration

Table of Contents

1. OBJECT CENTERED ACTION REPRESENTATION	3
2. LEARNING GRAMMARS FOR MANIPULATION ACTIONS	4
3. 3D ARTICULATED HAND AND UPPER BODY TRACKING	5

This deliverable consists of 8 manuscripts [1, 2, 3, 4, 5, 6, 7, 8], appearing in or submitted to internationally recognized peer-reviewed journals and conferences.

The objective of WP3 is to define representations of human activity involving objects in the surrounding world in terms of *Object-Action Complexes* (OACs). For a formal definition of OACs, see the report¹, found at <http://www.paco-plus.org/>.

The representations should be such that a robot can observe the activity being performed by a human, represent it in terms of a sequence of OACs, map this sequence to its own embodiment, and perform the corresponding sequence of OACs. The application of this in the PACO-PLUS project is visual robot learning from human demonstration.

The following aspects of an OAC are relevant to consider for learning from demonstration purposes:

- An OAC is associated with a set of *attributes* with associated values. The attributes are aspects of the scene in which the action takes place, relevant to the action; the most obvious attribute is the presence/non-presence of a certain object class in the scene. Certain attribute values are prerequisites for the OAC to take place, e.g., the scene presence of a certain object class. Others can be changed during the execution of the OAC, e.g., the filling level of a container during pouring from that container.
- An OAC is associated with a *prediction function* with which it is possible to estimate how the state of the scene – or rather, those aspects of the scene associated with the OAC’s attributes – will change during the execution of the OAC. The prediction function can also encode the range of initial attribute values that make execution of the OAC possible.

Apart from this, the OAC representation¹ comprises functionality for evaluating the actual outcome of the OAC when executed, compared to the belief encoded in the prediction function. This makes it possible for a robot (or an embodied agent in general) to refine its OAC representation through exploration, after learning an initial model from a human (or other agent). However, this aspect of the learning of OACs from demonstration is not discussed in this deliverable.

In the following, the different contributions are described in Sections 1 (relating to Tasks 3.1.1, 3.2.1), 2 (relating to Tasks 3.1.2, 3.2.2), and 3 (relating to Tasks 3.1.3, 3.2.3), respectively.

1. Object Centered Action Representation

In the work [2] we discuss our novel approach *Tracking in Action Space* (TAS). The TAS concept is completely dependent on the OAC concept without which it would not be possible. Many action recognition approaches rely on a functioning 3D human body tracking methods approach. Here, the action recognition is considered as a two step bottom-up process of first tracking the human 3D pose and then recognition the action based on the recovered 3D poses. The TAS concept, on the other hand, is a top-down approach. The computation is embedded into a particle filtering concept: First, the TAS approach makes a guess about an action by sampling from a prior density over the set of possible actions (*action prior*). Then, the *guessed* action gives rise to a pose which can be cross-checked with the input image. The guessed action random variable from the action prior is multi-dimensional and contains information on a) which action it is, b) the timing /phase in that action and c) the parameters of that action (e.g. where to point at or where to reach and

¹N. Krüger, J. Piater, C. Geib, R. Petrick, M. Steedman, F. Wörgötter, A. Ude, T. Asfour, D. Kraft, D. Omrcen, A. Agostini, R. Dillmann, *Object-action complexes: Grounded abstractions of sensorimotor processes*, submitted to Robotics and Autonomous Systems, 2010

grasp). Apart from several computational advantages (such as decreased dimensionality of the parameter space), the TAS concept requires a good action prior. This prior is conditioned by

- the objects on which the actions are being executed: The object affordances constrain which action can be executed on it, and the object state (location, size, orientation, weight, etc.) constraints the parameters of that action
- the action that was executed earlier: As actions are often executed in a certain ordered sequence (see also [4, 8]) the previously executed action and its parameters also conditions the action prior.

The article [3] presents a method to object categorization according to function, i.e., learning the *affordances* of objects from human demonstration. Affordances as a concept is closely linked to OACs, and the method could also be described as a means to learn OACs from human demonstration. Object affordances (functionality) is inferred from observations of humans using the objects in different types of actions. The method is able to simultaneously segment (in time) and classify human hand actions, and detect (in space) and classify the objects involved in the action.

The *scene attributes* considered in the system are the presence/non-presence of a number of object classes in the vicinity of the human. Different OACs, i.e., different combinations of human actions and objects present near the human, have different *prediction functions*. The prediction functions are represented as conditional probabilities over OAC class, conditioned on the human motion and object presence during the execution of the OAC.

2. Learning Grammars for Manipulation Actions

In the works [4, 8] we discuss two different approaches on learning action grammars for manipulative actions on objects. In the earlier work [8] we developed a statistical framework along the line of hidden Markov models that allows to cluster observed trajectories (e.g. as they come from the human body parts) and cluster them into their common parts. This way the common parts become the alphabet of action primitives and the rules that govern how these primitives are used to form the actions are modeled with grammars. One observation we made in [8] was that the analysis of trajectories does not usually lead to the desired results due to the lack of a proper metric between the trajectories. When using an Euclidean distance measure in the space of trajectories (Cartesian space of 3D locations or space of joint angles were investigated) then the observation was that different locations of the human actor or the involved object results in completely different trajectories for the same action. This means that if either the human actor or the object moves, we lose the possibility of comparing the trajectories in a sensible way. As a solution, we propose in [4] to approach the problem of finding the action alphabet in a different space. In [4] we start from observing that human actions can be interpreted from two different (but possibly equivalent) perspectives: a) from the perspective of the space of trajectories of the human body parts (as done in [8]) and b) from the perspective of the *effect* that an action has on the scenario. For example a push action on some object *A* has the effect that the location of object *A* changes. Consequently, in [4] we identify the set of action primitives based on the *effects* that are common across the different training actions. Using the clustering in the *effect space* and propagating it to the space of trajectories, we are able to identify equivalence classes of trajectory pieces where each equivalence class contains trajectories that have the same effect on the objects in the context. In other words, we use the *effect* of the actions trajectories on the scene as the metric between the trajectories. This means, while we failed to identify the action primitives based solely on the observed trajectories, the use of objects and their state provides us with a very good solution for action primitive detection. Indeed, considering the effect of the action primitives, one is tempted identify this effect with the semantics of the corresponding primitive.

3. 3D Articulated Hand and Upper Body Tracking

The work presented in [5, 6, 7] concerns a method for vision based estimation of the pose of human hands in interaction with objects. Despite the fact that most robotics applications of human hand tracking involve grasping and manipulation of objects, the majority of methods in the literature assume a free hand, isolated from the surrounding environment. Grasping hand reconstruction is a more challenging problem than the reconstruction of hands in isolation, since the object often occludes large parts of the hand. However, for our applications where objects are manipulated by the human, it is necessary to handle severe object occlusion. This hand tracking system functions in real time, and is a prerequisite for the OAC recognition system [3] described in Section 1.

The hand tracking system was integrated with the upper body tracking system developed at the University of Karlsruhe; the integration is described in [1]. Experiments demonstrated the possibility to map the grasping actions of a human to the Armar III robot using the two integrated tracking systems. These grasping actions are highly dependent on the type and functionality of the grasped object. They can thus be regarded as OACs with attributes corresponding to object shape.

Attached Papers

- [1] M. Do, J. Romero, H. Kjellström, P. Azad, T. Asfour, D. Kragic, R. Dillmann, Grasp recognition and mapping on humanoid robots, in *IEEE-RAS International Conference on Humanoid Robots*, 2009.
 - [2] D. Herzog, V. Krüger, Tracking in action space, submitted to *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.
 - [3] H. Kjellström, J. Romero, D. Kragic, Visual object-action recognition: Inferring object affordances from human demonstration, submitted to *Computer Vision and Image Understanding*, 2010.
 - [4] V. Krüger, D. Herzog, Sanmohan, A. Ude, D. Kragic, Learning actions from observations, submitted to *Robotics and Automation Magazine*, 2009.
 - [5] J. Romero, H. Kjellström, D. Kragic, Hands in action: Real-time 3D reconstruction of hands in interaction with objects, in *IEEE International Conference on Robotics and Automation*, 2010.
 - [6] J. Romero, H. Kjellström, D. Kragic, Monocular real-time 3D articulated hand pose estimation, in *IEEE-RAS International Conference on Humanoid Robots*, 2009.
 - [7] J. Romero, H. Kjellström, D. Kragic, Modeling and evaluation of human-to-robot mapping of grasps, in *International Conference on Advanced Robotics*, 2009.
 - [8] Sanmohan, V. Krüger, D. Kragic, H. Kjellström, Automatic primitive segmentation and action recognition, submitted to *Advanced Robotics*, 2009.
-

Grasp Recognition and Mapping on Humanoid Robots

Martin Do, Javier Romero, Hedvig Kjellström, Pedram Azad,
Tamim Asfour, Danica Kragic, Rüdiger Dillmann

Abstract—In this paper, we present a system for vision-based grasp recognition, mapping and execution on a humanoid robot to provide an intuitive and natural communication channel between humans and humanoids. This channel enables a human user to teach a robot how to grasp an object. The system comprises three components: human upper body motion capture system which provides the approaching direction towards an object, hand pose estimation and grasp recognition system, which provides the grasp type performed by the human as well as a grasp mapping and execution system for grasp reproduction on a humanoid robot with five-fingered hands. All three components are real-time and markerless. Once an object is reached, the hand posture is estimated, including hand orientation and grasp type. For the execution on a robot, hand posture and approach movement are mapped and optimized according to the kinematic limitations of the robot. Experimental results are performed on the humanoid robot ARMAR-IIIb.

I. INTRODUCTION

A humanoid robot's capability of autonomously adapting and acting in new and unstructured environments is very limited. In the majority of cases, a skilled and experienced user is needed for the programming in order to adapt an existing action to a new situation. To enable teaching of a robot by non-expert users, a natural intuitive interface is needed. Since imitation presents an obvious solution for tackling this problem, this field has received great interest in humanoid robotics. The benefit of exploiting demonstration is clearly revealed in [1], where an anthropomorphic arm is capable of balancing a pole in the first trial after observing a human.

A challenging problem where a robot could greatly benefit from a human demonstration is an object grasping task. Such a task involves the control of several degrees of freedom, visual servoing, tactile feedback, etc., turning it to a highly complex task. About the grasp action, a grasp can be divided in two stages: an approach stage and final grasp stage. Due to high object variety concerning shape, size, and mass, determining an adequate approach movement and selecting a suitable grasp type increase the chances that an object is successfully grasped. Instead of telling the robot explicitly which approach movement and which grasp type shall be

used, it is desirable to have a system which enables the robot to observe a human during grasp execution and to imitate the demonstration. For the implementation of such a system, various problems have to be tackled, like observation of the human performing the grasp, the mapping of the grasp, and the final execution on the robot.

An important part of the grasp imitation system is the block in charge of getting information about the arm and hand movements. In order to provide this information, the approach movement of the arm as well as the hand pose have to be recognized. Aiming towards ease of use, markerless systems seem to be the most obvious solution for the observation of human grasps since, besides vision sensors, additional equipment is avoided and the preparation effort is kept to a minimum. However, markerless 3D motion capturing and reconstruction of hand pose based on image data are extremely difficult problems due to unstructured environments, the large self-occlusion, high dimensionality and non-linear motion of the arm and the fingers.

Besides the perception modules, another crucial part of an imitation system consists of the mapping and the execution of an observed human grasp on a humanoid robot. Due to severe constraints of mechanical systems and differences between the human and the robot's embodiment, a large number of requirements arise, which are difficult to be satisfied at once. Towards enabling a humanoid to imitate a human grasp, our system integrates several subsystems and methods. First, using a stereo camera setup human observation is initiated by capturing upper body motion and scanning the scene for known objects to attain information on the approach stage. Subsequently, grasp classification and hand orientation are provided through the estimation of the full hand pose in a non-parametric fashion. Finally, the motion data is gathered and mapped onto the robot for execution. The mapping is accomplished via a standardized interface and the ensuing execution is achieved by means of non-linear optimization.

II. RELATED WORK

Several approaches have been made to create a markerless human motion capture system for humanoid robots. Especially, image-based approaches have been a major focus of this field. These approaches are either search-based ([2], [3]), utilize an optimization approach based on 2D-3D correspondences [4], [5], or are based on particle filtering. In [6], it was shown that human motion can be successfully tracked with particle filtering, using three cameras positioned around the scene of interest.

J. Romero, H. Kjellström and D. Kragic are with KTH - Royal Institute of Technology, Stockholm, Sweden, as members of the Computer Vision & Active Perception Lab., Centre for Autonomous Systems, e-mail: jrgn, hedvig, dani@kth.se

M. Do, P. Azad, T. Asfour and R. Dillmann are with the University of Karlsruhe (TH), Karlsruhe, Germany, as members of the Institute for Anthropomatics, e-mail: do, azad, asfour, dillmann@ira.uka.de

Towards imitation of human motion by a robot, the mapping and execution of motion capture data are issues whereas possible solutions pursue strategies which either make use of artificial markers and landmarks or which are based on the transfer and post-processing of joint angles. Marker-based approaches are presented in [7] and [8] where methods based on minimization of the mismatch between robot and human markers are introduced. However, in [9] and [10], joint angles of a demonstrators posture are determined and transferred to the robot for execution. Due to joint and velocity constraints, a scaling and transformation process must be performed in order to obtain a feasible joint angle configuration for the robot.

Analysis of human hand pose for the purpose of learning by demonstration (LbD), see [11] has been thoroughly investigated, almost exclusively with the help of markers and/or 3D sensors attached to the human hand. In the work by Oztop [12] motion capture, color segmentation with artificially colored hands, and active-marker capture systems were compared. Magnetic gloves have also been used extensively because of their accuracy [13]. Another input source for LbD systems is the passive joint measurements of the robot itself [14]. However, the methods shown above all use invasive devices. We envision a LbD scenario where the teaching process can be initiated without calibration and where the robot-user interaction is as natural as possible. For this reason, we want to reconstruct the hand posture in a visual markerless fashion.

Methods for hand pose estimation that are not constrained to a limited set of poses can largely be classified into two groups [15]: I) model based tracking and II) single frame pose estimation. Methods of type I) usually employ generative articulated models [16], [17], [18], [19]. Since the state space of a human hand is extremely high-dimensional, they are generally very computationally demanding, which currently makes this approach intractable for a robotics application. Methods of type II) are usually non-parametric [20], [21]. They are less computationally demanding and more suited for a real-time system, but also more brittle and sensitive to image noise, since there is no averaging over time. The method presented here falls into the second approach. However, it takes temporal continuity into account and it can be used for online real-time reconstruction.

III. GRASP OBSERVATION

As mentioned before, we assume that a grasp consists of an approaching stage and a final grasp stage. The observation of the whole grasping process involves recognition of the grasp type, estimation of the approach arm movement and object detection. Following the target of having an intuitive and natural programming interface for robots, we use a markerless human motion capture system for the observation of human motion using the stereo vision system of the robot's head [22]. The head has two eyes and each eye is equipped with two cameras, one with a wide-angle lens for peripheral vision and one with a narrow-angle lens for foveal vision.

First, the robot recognizes known objects in the scene and starts capturing human motion. The hand pose estimation system is triggered as soon as the human hand is in the vicinity of the object. To obtain a close-up of the hand, the foveal cameras are used. The grasp observation is finished with the classification of the observed human grasp.

A. Hand Pose Estimation

The input to the method is a sequence $[\mathbf{I}_t], t = 1, \dots, n$ of monocular images of the human hand [21].

In each frame \mathbf{I}_t , the hand is segmented using skin color segmentation based on color thresholding in HSV space. The result is a segmented hand image \mathbf{H}_t .

The shape information contained in \mathbf{H}_t is represented with a Histogram of Oriented Gradients (HOG). This feature has been frequently used for representation of human and hand shape [23], [24], [25]. It has the advantage of being robust to small differences in spatial location and proportions of the depicted hand, while capturing the shape information effectively.

1) *Non-parametric Pose Reconstruction*: In this section, we omit the time index and regard the problem of reconstructing a *single* pose \mathbf{p} from a *single* HOG \mathbf{x} .

Our goal is to obtain the grasp class and orientation of the human hand. We can infer this information from the pose \mathbf{p} of the hand, since all this information is stored for each entry of the database. Therefore, we want to find the mapping $\hat{\mathbf{p}} = \mathcal{M}(\mathbf{x})$, where $\hat{\mathbf{p}}$ is the estimated 31D hand pose in terms of global orientation (lower arm yaw, pitch, roll) and joint angles (3 wrist joint angles, 5 joint angles per finger), and \mathbf{x} is the observed 512D HOG representation of the hand view, described in Section III-A.

The mapping function \mathcal{M} can be expected to be highly non-linear in the HOG space, with large discontinuities. Following [21], \mathcal{M} is therefore represented non-parametrically, i.e., as a database of example tuples $\{\langle \mathbf{x}_i, \mathbf{p}_i \rangle\}, i \in [1, N]$. Due to the high dimensionality of both the HOG space (512D) and the state space (hereafter denoted JOINT space, 31D), the database needs to be of a considerable size to cover all hand poses to be expected; in our current implementation, $N = 90000$. This has two implications for our mapping method, as outlined in the subsections below.

2) *Generation of Database Examples*: Generating a database of 10^5 examples from real images is intractable.



Fig. 1. Ambiguity in mapping from HOG space to JOINT space. Even though it is visually apparent that $\|\mathbf{p} - \mathbf{p}_2\| \ll \|\mathbf{p} - \mathbf{p}_1\|$ in JOINT space, database instance 1 will be regarded as the nearest neighbor as $\|\mathbf{x} - \mathbf{x}_1\| < \|\mathbf{x} - \mathbf{x}_2\|$. Note that the object in the hand just contributes with occlusion of the hand in HOG extraction, as it is then colored uniformly with background color.

Instead, we used the graphics software Poser 7 to generate synthetic views $\mathbf{H}_i^{\text{synth}}$ of different poses. The database examples are chosen as frames from short sequences of different grasp types from different view points, different grasped objects, and different illuminations.

The grasp types are selected according to the taxonomy developed in the GRASP project¹, which integrates the Cutkosky [26], Kamakura [27], and Kang [28] taxonomies. The whole database is also available at the same place.

From each example view $\mathbf{H}_i^{\text{synth}}$, the tuple $\langle \mathbf{x}_i, \mathbf{p}_i \rangle$ is extracted, where \mathbf{x}_i is generated from $\mathbf{H}_i^{\text{synth}}$ as described in Section III-A, and \mathbf{p}_i is the pose used to generate the view $\mathbf{H}_i^{\text{synth}}$ in Poser 7.

3) *Approximate Nearest Neighbor Extraction*: Given an observed HOG \mathbf{x} , the goal is to find an estimated pose $\hat{\mathbf{p}} = \mathcal{M}(\mathbf{x})$. With the non-parametric mapping approach, the mapping task $\hat{\mathbf{p}} = \mathcal{M}(\mathbf{x})$ is one of searching the database for examples $\langle \mathbf{x}_i, \mathbf{p}_i \rangle$ such that $\mathbf{x}_i \approx \mathbf{x}$. More formally, X_k , the set of k nearest neighbors to \mathbf{x} in terms of Euclidean distance in HOG space, $d_i = \|\mathbf{x} - \mathbf{x}_i\|$ are retrieved.

As an exact k NN search would put serious limitations on the size of the database, an approximate k NN search method, Locality Sensitive Hashing (LSH) [29] is employed. LSH is a method for efficient ϵ -nearest neighbor (ϵ NN) search, i.e. the problem of finding a neighbor $\mathbf{x}_{\epsilon\text{NN}}$ for a query \mathbf{x} such that

$$\|\mathbf{x} - \mathbf{x}_{\epsilon\text{NN}}\| \leq (1 + \epsilon)\|\mathbf{x} - \mathbf{x}_{\text{NN}}\| \quad (1)$$

where \mathbf{x}_{NN} is the true nearest neighbor of \mathbf{x} . The computational complexity of ϵ NN retrieval with LSH [29] is $\mathcal{O}(DN^{\frac{1}{1+\epsilon}})$ which gives sublinear performance for any $\epsilon > 0$.

4) *The Mapping \mathcal{M} is Ambiguous*: The database retrieval described above constitutes an approximation to the true mapping $\hat{\mathbf{p}} = \mathcal{M}(\mathbf{x})$, robust to singularities and discontinuities in the mapping function \mathcal{M} .

However, it can be shown empirically that \mathcal{M} is inherently ambiguous (one-to-many); substantially different poses \mathbf{p} can give rise to the similar HOGs \mathbf{x} [23]. An example of this is shown in Figure 1.

Thus, the true pose \mathbf{p} can not be fully estimated from a single HOG \mathbf{x} (using any regression or mapping method); additional information is needed. In the next section, we describe how temporal continuity assumptions can be employed to disambiguate the mapping from HOG to hand pose.

5) *Time Continuity Enforcement in JOINT Space*: We now describe how temporal smoothness in hand motion can be exploited to disambiguate the mapping \mathcal{M} .

Consider a sequence of hand poses $[\mathbf{p}_t], t = 1, \dots, n$, that have given rise to a sequence of views, represented as HOGs $[\mathbf{x}_t], t = 1, \dots, n$. Since the mapping \mathcal{M} is ambiguous, the k nearest neighbors to \mathbf{x}_t in the database, i.e. the members of the set X_k , are all similar to \mathbf{x}_t but not necessarily corresponding to hand poses similar to \mathbf{p}_t . An important implication of this is that a sequence of hand poses $[\mathbf{p}_t], t = 1, \dots, n$ does not necessarily give rise to a

¹www.grasp-project.eu.

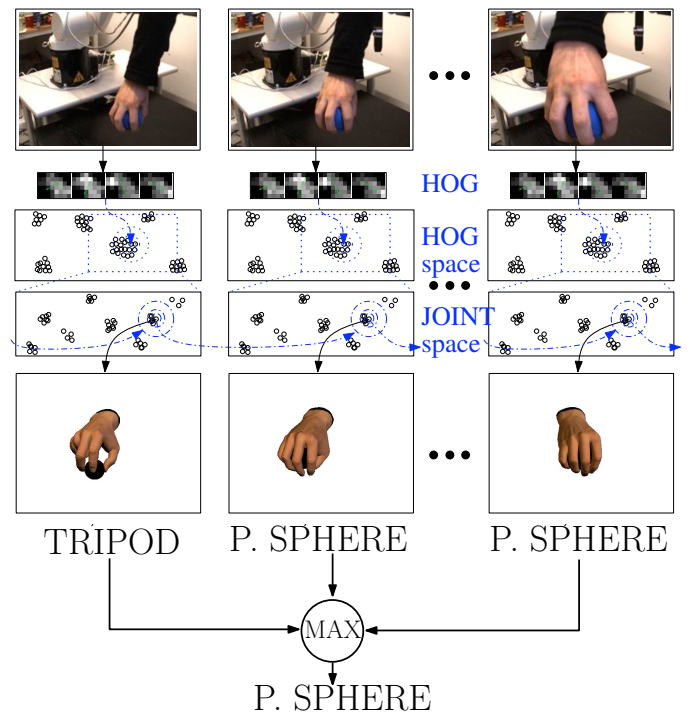


Fig. 2. Grasp Classification with continuity enforcement in JOINT space

sequence of HOGs $[\mathbf{x}_t], t = 1, \dots, n$ continuous in the HOG space.

However, due to the physics of the human body, the speed of the hand articulation change is limited. Thus, the sequence of hand poses $[\mathbf{p}_t], t = 1, \dots, n$, i.e. the *hidden variables*, display a certain continuity in the JOINT space. This is illustrated in Figure 2.

The hand pose recognition for a certain frame t is therefore divided into two stages; I) retrieval of a set of k nearest neighbors X_k using single frame non-parametric mapping, as described in Section III-A.1; II) weighting of the members of X_k according to their time continuity in the JOINT space.

Let P_k be the set of poses corresponding to the k NN set X_k found in stage I). Moreover, let $\hat{\mathbf{p}}_{t-1}$ be the estimated pose in the previous time step. In stage II), the members $\mathbf{p}_j, j \in [1, k]$ of P_k are weighted as

$$\omega_j = e^{-\frac{\|\mathbf{p}_j - \hat{\mathbf{p}}_{t-1}\|}{2\sigma^2}}. \quad (2)$$

where σ^2 is the variance of the distance from each entry pose \mathbf{p}_j to the previous estimated pose $\hat{\mathbf{p}}_{t-1}$.

The pose estimate at time t is computed as the weighted mean of P_k :

$$\hat{\mathbf{p}}_t = \left(\sum_{j=1}^k \omega_j \mathbf{p}_j \right) / \left(\sum_{j=1}^k \omega_j \right). \quad (3)$$

The grasp class estimation G_t is obtained through a majority voting process within the N_p poses with the highest weight ω_j (for our experiments $N_p = 15$). G_t is then smoothed temporally taking the majority vote in a temporal window of N_f frames ($N_f = 10$ in our experiments). This

can be seen in Figure 2. The whole system runs at 10 Hz on a 1.8 GHz single core CPU.

B. Object Recognition

For the robust recognition and accurate 6D pose estimation of single-colored objects, in our previous work, we have developed a model-based approach based on a combination of stereo triangulation, matching of global object views and online projection of a 3D model of the object [30]. The requirement for the approach is global segmentation of the objects, which is accomplished by color segmentation. For training, a 3D model of the object is used to generate views with different object orientations in simulation. Each view is stored along with its corresponding orientation. For recognition, each region candidate obtained by the segmentation routine is matched against the database. An initial orientation estimate is given by the stored orientation information with the matched view. An initial position estimate is given by the stereo triangulation result of the segmented regions in the left and right camera image. The triangulation result of the centroids depends on the view of the object and thus cannot serve as a constant reference point. In order to solve these problems, a pose correction algorithm is applied, which make use of online projection of the 3D model. This pose correction algorithm is an iterative procedure, which in each iteration corrects the position vector by computing the triangulation error in simulation and correcting the orientation estimate on the basis of the updated position estimate.

C. Markerless Motion Capture

In the following, our real-time stereo-based human motion capture system presented in [31] will be summarized briefly. The input to the system is a stereo color image sequence, captured with the built-in wide-angle stereo pair of the humanoid robot ARMAR-IIIb, which can be seen in Figure 5. The input images are preprocessed, generating output for an edge cue and a so-called distance cue, as introduced in [32]. The image processing pipeline for this purpose is illustrated in Figure 3. Based on the output of the image processing pipeline, a particle filter is used for tracking the movements in joint angle space. For tracking the movements, a 3D upper body model with 14 DoF (6 DoF for the base transformation, 2-3 for the shoulders, and 2-1 for the elbows) consisting of rigid body parts is used, which provides a simplified description of the kinematic structure of the human upper body. The model configuration is determined by the body properties like the limbs length of the observed human subject. The core of the particle filter is the likelihood function that evaluates how well a given model configuration matches the current observations, i.e. stereo image pair. For this purpose, an edge cue compares the projected model contours to the edges in the image. On the basis of an additional 3D hand/head tracker, the distance cue evaluates the distance between the measured positions and the corresponding positions inferred by the forward kinematics of the model. Various extensions are necessary for robust real-time application such as a prioritized fusion

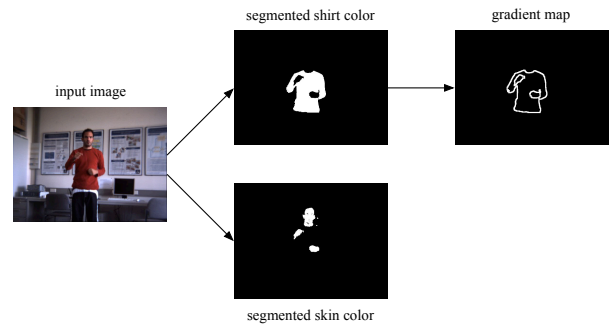


Fig. 3. Illustration of the image processing pipeline.

method, adaptive shoulder positions, and the incorporation of the solutions of the redundant arm kinematics. The system is capable of online tracking of upper body movements with a frame rate of 15 Hz on a 3 GHz single core CPU. Details are given in [31].

IV. GRASP MAPPING

Before the execution on the robot, the approach movement in the form of joint angle configurations and the recognized grasp type are mapped onto the robot. In order to map motion onto the robot, we proposed in our previous work (see [33]) the Master Motor Map (MMM), a standardized interface which features a high level of flexibility and compatibility, since it allows mapping from various motion capture systems to different robot embodiments. The MMM provides a reference kinematic model of the human body by defining the maximum number of DoF, currently 58, that can be used by a human motion capture module and a robot. Trajectories in the MMM file format can be represented in joint angle space as well as in Cartesian space. Concerning movements in Cartesian space, in order to enable grasping and manipulation tasks, the MMM provides mapping of the desired 6D pose and the grasp type on the robot's end effector. A proper connection via the MMM of a motion capture module to a robot requires the implementation of a conversion module which transforms module specific data into the MMM file format and vice versa for overcoming different Euler conventions, active joint sets and orders of the joint angle values between the modules. As depicted in Figure 4, in the current system one conversion module has been implemented for each human motion capture system, converting the motion capture data to the MMM format. A third conversion module is implemented for mapping the MMM data to the kinematics of ARMAR-IIIb.

Along with the approach movement in the form of joint angle values the grasp type and the estimated hand orientation are passed from the hand pose estimation system to the robot through the MMM interface. According this data, from a set of preimplemented grasp the corresponding one is selected to be executed. To complete the grasp mapping, the grasp type to be performed is adjusted regarding the extent of the object shape. For this purpose, a rudimentary grasp type adjustment is implemented, which projects the object

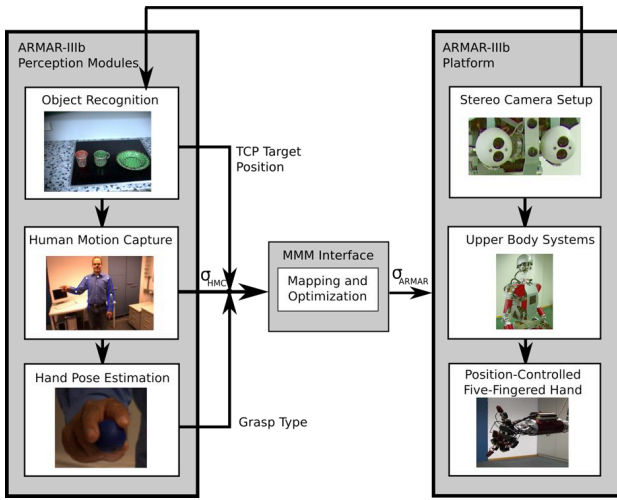


Fig. 4. Structure of the entire framework.

shape onto the thumbs position such that the thumbs tip lies on the shapes margin. The aperture of the fingers is scaled in a way that the positions of the remaining finger tips also approximately meet the margin of the shape. This method works on objects with simple shape properties.

A. Grasp Execution

The grasp reproduction of ARMAR-IIIb is performed in three different stages. The first stage describes the approach movement of the end effector towards the object based on the observed movement, while in the second stage the end effector is placed at the final grasp pose. The reproduction concludes with the execution of the recognized grasp type. Regarding the approach stage, by mapping these joint angle movements onto the robot, through forward kinematics one obtains a trajectory of the TCP in Cartesian space. The resulting trajectory is not sufficient for a goal-directed reproduction due to differences in the kinematic structure between the embodiments of the robot and a human e.g. mechanical joint constraints, differing joints and limb measurements. Therefore, the TCP trajectory for movements such as grasping is stretched and directed towards the object position to be reached. In order to attain a goal-directed reproduction, which additionally should feature a high similarity to the demonstrated human movement, in each frame, joint angles as well as desired TCP position of the modified trajectory have to be considered during execution. In [34], we developed an approach, which supports reproduction of observed human motion on the robot using non-linear optimization methods. In order to formulate an optimization problem which comprises displacements in Cartesian space regarding the TCP position as well as in joint angle space, a similarity measure is defined as follows:

$$S(\sigma) = 2 - \frac{\frac{1}{n} \sum_{i=1}^n (\hat{\sigma}_i^t - \sigma_i)^2}{\pi^2} - \frac{\frac{1}{3} \sum_{k=1}^3 (\hat{p}_k^t - p_k)^2}{(2 \cdot l_{arm})^2} \quad (4)$$

with n representing the number of joints, $\sigma_i, \hat{\sigma}_i^t \in [0, \pi]$ and $p_k, \hat{p}_k^t \in [-l_{arm}, l_{arm}]$, whereas l_{arm} describes the robot's arm length. The reference joint angle configuration is denoted by $\hat{\sigma} \in \mathbb{R}^n$, while $\hat{p} \in \mathbb{R}^3$ stands for the desired TCP position. The current TCP position \mathbf{p} can be determined by applying the forward kinematics of the robot to the joint angle configuration σ . Based on Equation 4 and the joint constraints $\{(C_{min}, C_{max})\}$ of a robot with n joints, one obtains following constrained optimization problem:

$$\min S'(\sigma) = 2 - S(\sigma) \quad (5)$$

$$\text{subject to } C_{i_{min}} \leq \hat{\sigma}_i \leq C_{i_{max}} \quad (6)$$

For solving Equation 5, we apply the Levenberg-Marquardt algorithm, since it features numerical stability and more robust convergence compared to other optimization algorithms such as the Gauss-Newton and the steepest descent method. Following this optimization approach a trade-off is attained, which on the one hand results in an accurate TCP positioning with small displacement error while it provides on the other hand a feasible robot joint angle configuration resembling the observed human configuration. This way goal-directed imitation of the approach movement is achieved. For further details, the reader is referred to [34]. For the execution of the final grasp phase, due to errors and inaccuracies originating from the object localization and the robot's mechanical elements, a displacement error arises between the TCP and the object that has to be diminished. To achieve exact alignment of the end effector and the robot, we make use of visual servoing methods as presented in [35]. Within this approach the hand and object are tracked. The resulting distance between both is reduced and the hand orientation is controlled. The hand orientation estimate coming from the grasp recognition module is used to determine if the grasp should be executed from the top or from the side. Therefore, the hand is placed over the object if the palm orientation was similar to the table plane, or next to the object otherwise.

V. EXPERIMENTS

A. Experimental Setup

The humanoid platform ARMAR-IIIb, a copy the humanoid robot ARMAR-IIIa [36], serves as the experimental platform in this work. From the kinematics point of view, the robot consists of seven subsystems: head, left arm, right arm, left hand, right hand, torso, and a mobile platform. The head has seven DoF and is equipped with two eyes, which have a common tilt and independent pan. Each eye is equipped with two digital color cameras, one with a wide-angle lens for peripheral vision and one with a narrow-angle lens for foveal vision. The upper body of the robot provides 33 DoF: 2.7 DoF for the arms and three DoF for the torso. The arms are designed in an anthropomorphic way: three DoF for each shoulder, two DoF in each elbow and two DoF in each wrist. Each arm is equipped with a five-fingered hand with eight DoF. The locomotion of the robot is realized using a wheel-based holonomic platform.

The proposed approach was integrated on the humanoid platform ARMAR-IIIb and was successfully applied. For

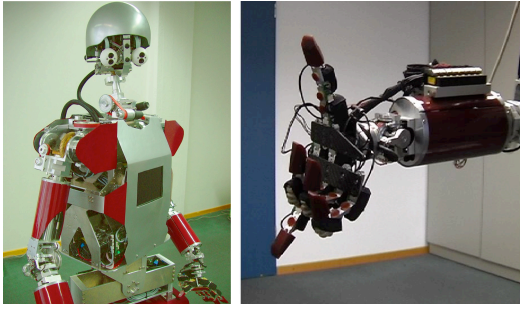


Fig. 5. Left: The humanoid robot ARMAR-IIIb. Right: Position-controlled right hand with 8 DoF.

the experiments, objects were used which can be easily identified such as single-colored cups. The experimental setup stipulates that demonstration of the grasp is performed in front of the robot. Observation is initiated by scanning the scene for known objects. Once an object is found, tracking of the human upper body is triggered leading to the capturing process of movements in the approach stage. This process is finished once the hand is positioned within a tolerated distance to a specific object. At this point, observation is switched to the hand pose estimation whereby its classification and the outgoing orientation complete the motion data of the grasp. As described in Section IV, the data is mapped onto robot, optimized to its embodiment and executed. In the execution phase, the robot searches for the same object which was grasped in the demonstration and approaches it. Based on the classification of the grasp type, an adequate instance is selected from the set of implemented grasp on the robot which is modified to the objects appearance. The hand pose recognition system was running on an external computer, while the rest of the system was running on ARMAR-IIIb. The communication between the two systems was performed through UDP sockets. It is possible to run the whole system on the robot, but this setup was more preferable for debugging purposes. Two sets of experiments were performed: in the first one, the whole system (grasp observation, mapping and execution) was tested with a reduced set of grasps: power grasp from top, power grasp from side, and pinch grasp (see Figure 6). In the second one, the set of grasps was extended to five of them (power sphere, prismatic wrap, parallel extension, tripod, and pinch). However, the execution of the grasp was reduced to the hand pose, keeping the arm still (see Figure 7).

B. Experimental Results

As depicted in Figures 6 and 7 the robot successfully imitated the demonstrated grasp including approach and grasp type. Since a non-linear optimization method is applied during approaching, we attained a trade-off between the similarity of the reproduced movement concerning the demonstration and accuracy in terms of positioning of the end effector regarding goal-directed tasks. Furthermore, the applied method provided a unique solution in terms of joint angles, which standard inverse kinematics methods fail to

do due to singularities and redundancies. Nevertheless, in the approach phase, we experienced a displacement error of max $65mm$ caused by kinematic inaccuracies which varies depending on the cups distance regarding the end effector. The displacement could be recovered by using visual servoing. In order to test the grasp classification module, each grasp was executed 20 times for the Experiment 2. The results are shown in Table I. An overall classification accuracy of 72% was achieved, clearly over the human baseline for grasp recognition with similar grasps [21], with four out of five grasp types with accuracies over 80%. The differences between human model and synthetic had a stronger effect in the parallel extension grasp, lowering the accuracy for that particular grasp. Results of the grasp recognition, mapping and execution on the humanoid robot ARMAR-IIIb are shown in the accompanying video submission, which is also available under www.iam.ira.uka.de/users/do/GraspRecognitionDivx.avi.






Grasp Type	Illustration	Correct Classification Rate
Power Sphere		80 %
Prismatic Wrap		95 %
Parallel extension		50 %
Tripod		85 %
Pinch		80 %

TABLE I
GRASP TYPE CLASSIFICATION RESULTS.

VI. CONCLUSIONS

In this paper, we presented a system for grasp recognition, mapping and execution on a humanoid robot. Human grasping activities are captured using markerless motion capture system and mapped to the humanoid robot ARMAR-IIIb. Human upper body tracking, object tracking and hand pose estimation techniques are applied to perceive human object grasping movements. The recognized grasps are mapped and executed on a humanoid robot with a five-fingered hand.

VII. ACKNOWLEDGMENTS

The work described in this paper was partially conducted within the EU Cognitive Systems projects GRASP (FP7-215821) and PACO-PLUS (FP6-027657) funded by the European Commission.

REFERENCES

- [1] C. G. Atkeson and S. Schaal, "Learning tasks from a single demonstration," in *IEEE International Conference on Robotics and Automation (ICRA97)*, 1997, pp. 1706–1712.
- [2] D. Gavrila and L. Davis, "3-D Model-based tracking of humans in action: a multi-view approach," in *Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, San Francisco, USA, 1996, pp. 73–80.

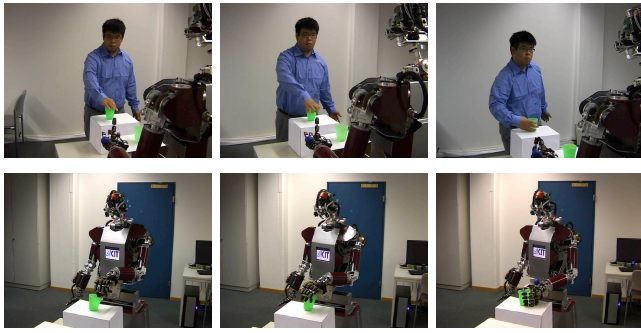


Fig. 6. Image Samples of Experiment 1. Top: human performing pinch grasp, top power grasp, and side power (left to right). Bottom: reproduction of the approach movement and the grasp type to grasp an object.

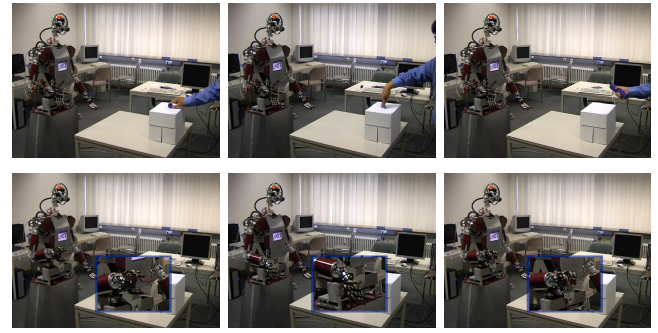


Fig. 7. Image Samples of Experiment 2. Top: human performing tripod grasp, parallel extended finger grasp, and prismatic wrap (left to right). Bottom: reproduction of the grasp type.

- [3] K. Rohr, "Human Movement Analysis based on Explicit Motion Models," in *Motion-Based Recognition*, 1997, pp. 171–198.
- [4] C. Bregler and J. Malik, "Tracking People with Twists and Exponential Maps," in *Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, Santa Barbara, USA, 1998, pp. 8–15.
- [5] D. Grest, D. Herzog, and R. Koch, "Monocular Body Pose Estimation by Color Histograms and Point Tracking," in *DAGM-Symposium*, Berlin, Germany, 2006, pp. 576–586.
- [6] J. Deutscher, A. Blake, and I. Reid, "Articulated Body Motion Capture by Annealed Particle Filtering," in *Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, Hilton Head, USA, 2000, pp. 2126–2133.
- [7] C. Kim, D. Kim, and Y. Oh, "Adaption of Human Motion Capture Data to Humanoid Robots for Motion Imitation using Optimization," *Integrated Computer-Aided Engineering*, vol. 13, no. 4, pp. 377–389, 2006.
- [8] A. Ude, C. Atkeson, and M. Riley, "Programming Full-Body Movements for Humanoid Robots by Observation," *Robotics and Autonomous Systems*, vol. 47, no. 2-3, pp. 93–108, 2004.
- [9] X. Zhao, Q. Huang, Z. Peng, and K. Li, "Humanoid Kinematics Mapping and Similarity Evaluation based on Human Motion Capture," in *IEEE International Conference on Information Acquisition*, Hefei, China, June 2004, pp. 426–431.
- [10] N. Pollard, J. Hodgins, M. Riley, and C. Atkeson, "Adapting Human Motion for the Control of a Humanoid Robot," in *IEEE International Conference on Robotics and Automation*, Washington, DC, USA, May 2002, pp. 1390–1397.
- [11] M. J. Mataric, "Sensory-motor primitives as a basis for imitation: Linking perception to action and biology to robotics," in *Imitation in Animals and Artifacts*, K. Dautenhahn and C. Nehaniv, Eds., 2000.
- [12] E. Oztop, J. Babic, J. G. Hale, G. Cheng, and M. Kawato, "From biologically realistic imitation to robot teaching via human motor learning," in *ICONIP (2)*, vol. 4985, 2007, pp. 214–221.
- [13] M. C. Lopes and J. S. Victor, "Motor representations for hand gesture recognition and imitation," in *IROS Workshop on Robotic Programming by Demonstration*, 2003.
- [14] S. Calinon, A. Billard, and F. Guenter, "Discriminative and adaptive imitation in uni-manual and bi-manual tasks," in *Robotics and Autonomous Systems*, vol. 54, no. 5, 2005, pp. 370–384.
- [15] A. Erol, G. N. Bebis, M. Nicolescu, R. D. Boyle, and X. Twombly, "A review on vision-based full DOF hand motion estimation," in *Vision for Human-Computer Interaction*, 2005, pp. III: 75–75.
- [16] Q. Delamarre and O. D. Faugeras, "3D articulated models and multiview tracking with physical forces," *Computer Vision and Image Understanding*, vol. 81, no. 3, pp. 328–357, 2001.
- [17] B. D. R. Stenger, A. Thayananthan, P. H. S. Torr, and R. Cipolla, "Filtering using a tree-based estimator," in *IEEE International Conference on Computer Vision*, 2003.
- [18] E. Sudderth, M. I. Mandel, W. T. Freeman, and A. S. Willsky, "Visual hand tracking using non-parametric belief propagation," in *IEEE Workshop on Generative Model Based Vision*, 2004.
- [19] M. de la Gorce, N. Paragios, and D. J. Fleet, "Model-based hand tracking with texture, shading and self-occlusions," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–8.
- [20] R. Rosales, V. Athitsos, L. Sigal, and S. Sclaroff, "3D hand pose reconstruction using specialized mappings," in *IEEE International Conference on Computer Vision*, 2001, pp. I: 378–385.
- [21] H. Kjellström, J. Romero, and D. Kragić, "Visual recognition of grasps for human-to-robot mapping," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008.
- [22] T. Asfour, K. Welke, P. Azad, A. Ude, and R. Dillmann, "The karlsruhe humanoid head," in *IEEE/RAS International Conference on Humanoid Robots (Humanoids)*, Daejeon, Korea, December 2008.
- [23] W. T. Freeman and M. Roth, "Orientational histograms for hand gesture recognition," in *IEEE International Conference on Automatic Face and Gesture Recognition*, 1995.
- [24] G. Shakhnarovich, P. Viola, and T. Darrell, "Fast pose estimation with parameter sensitive hashing," in *IEEE International Conference on Computer Vision*, vol. 2, 2003, pp. 750–757.
- [25] H. N. Zhou, D. J. Lin, and T. S. Huang, "Static hand gesture recognition based on local orientation histogram feature distribution model," in *Vision for Human-Computer Interaction*, 2004, p. 161.
- [26] M. Cutkosky, "On grasp choice, grasp models and the design of hands for manufacturing tasks," *IEEE Transactions on Robotics and Automation*, vol. 5, no. 3, pp. 269–279, 1989.
- [27] N. Kamakura, M. Matsuo, H. Ishi, F. Mitsuboshi, and Y. Miura, "Patterns of static prehension in normal hands," *Am J Occup Ther*, vol. 7, no. 34, pp. 437–45, 1980.
- [28] S. B. Kang and K. Ikeuchi, "Grasp recognition using the contact web," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1992, pp. 194–201.
- [29] A. Andoni and P. Indyk, "Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions," *Communications of the ACM*, vol. 51, 2008.
- [30] P. Azad, T. Asfour, and R. Dillmann, "Stereo-based 6d object localization for grasping with humanoid robot systems," in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, 29 2007–Nov. 2 2007, pp. 919–924.
- [31] P. Azad, T. Asfour, and R. Dillmann, "Robust real-time stereo-based markerless human motion capture," in *IEEE International Conference on Humanoid Robots*, Daejeon, Korea, December 2006, pp. 700–707.
- [32] P. Azad, A. Ude, T. Asfour, and R. Dillmann, "Stereo-based Markerless Human Motion Capture for Humanoid Robot Systems," in *IEEE International Conference on Robotics and Automation (ICRA)*, Roma, Italy, 2007, pp. 3951–3956.
- [33] P. Azad, T. Asfour, and R. Dillmann, "Toward a Unified Representation for Imitation of Human Motion on Humanoids," in *IEEE International Conference on Robotics and Automation*, Rome, Italy, April 2007.
- [34] M. Do, P. Azad, T. Asfour, and R. Dillmann, "Imitation of Human Motion on a Humanoid Robot using Non-Linear Optimization," in *IEEE International Conference on Humanoid Robots*, Daejeon, Korea, December 2008, pp. 545–552.
- [35] N. Vahrenkamp, S. Wieland, P. Azad, D. Gonzalez, T. Asfour, and R. Dillmann, "Visual servoing for humanoid grasping and manipulation tasks," in *Humanoid Robots, 2008. Humanoids 2008. 8th IEEE-RAS International Conference on*, Dec. 2008, pp. 406–412.
- [36] T. Asfour, K. Regenstein, P. Azad, J. Schröder, N. Vahrenkamp, and R. Dillmann, "ARMAR-III: An Integrated Humanoid Platform for Sensory-Motor Control," in *IEEE/RAS International Conference on Humanoid Robots*, 2006.

Tracking in Action Space

Dennis Herzog and Volker Krueger
CVMI-CIT, Aalborg University
Ballerup, Denmark
{deh, vok}@cvmi.aau.dk

Abstract

To recognize human actions such as pointing at objects (“Give me that...”), is difficult because they ought to be recognized independent of scene parameters such as viewing direction. Furthermore, the parameters of the action, such as pointing direction, are important pieces of information. One common way to achieve recognition is by using 3D human body tracking followed by action recognition based on the captured tracking data. General 3D body tracking is, however, still a difficult problem. But furthermore, we would like to argue that a) 3D body tracking and action recognition should be seen as an intertwined problem rather than a 2 step process and b) general 3D body tracking for action recognition neglects one important piece of information that can often be used to simplify it: the context in which the action takes place constraints which actions are most likely to happen and which parametrization they will have. In this paper, we are looking at human body tracking for action recognition from a context-driven perspective. Instead of the space of human body poses, we consider the space of possible actions of a given context and argue that 3D body tracking reduces to action tracking in the parameter space in which the actions live. This reduces the high-dimensional problem to a low-dimensional one. In our approach, we use parametric hidden Markov models to represent parametric movements; particle filtering is used to track in the space of action parameters. Our approach is content with monocular video data and we demonstrate its effectiveness on synthetic and on real image sequences. In the experiments we focus on human arm movements.

1. Introduction

Human communicative actions such as pointing (“Give me this”) or object grasping are typical examples of human actions in a human-to-human communication problem [1, 15]. These actions are usually context-dependent and their parametrization defines an important piece of their information [27, 1]. To capture these communicative actions

is challenging because a) the capturing should be independent of scene parameters such as viewing direction or viewing distance and b) one needs complex action models that allow to recover *what* action is performed and *which* parameters it has. The observation that the parameters of an action carry important information about the *meaning* of the action was already earlier pointed out by Wilson and Bobick in [27] using the example “*The Fish was this big*”.

One strategy for recognizing such actions [27, 20] is to first track the human movements using, e.g., a 3D body tracker and to then in a second step feed these tracks into an action recognition engine, such as, e.g., HMMs [17, 28] or even parametric HMMs (PHMMs) [27]. Considering the first ingredient of the above outlined strategy, it was pointed out recently again [16] that 3D tracking and pose estimation, especially from monocular views, is non-trivial. Common approaches are model-based generative ones [3, 23, 24], that compare a synthesized candidate pose with the image data.

In this paper, we would like to argue that such a 2 step approach as the above is un-necessary complication. Human actions are usually goal-directed and are performed within a certain context (*eating, cooking* etc.). Furthermore, actions are often performed on objects [8, 14] which leads to the observation that the objects can prime the actions performed on them (e.g. reaching, pointing, grasping, pushing-forward) [10, 2]. Thus, we would like to suggest to look at 3D human body tracking from an object and context-driven perspective: Instead of asking “*What is the set of joint angles that make a human model fit to the observation*” we ask “*What action causes a pose that fits to the observation*”. By replacing in a particle filter approach the propagation model for joint angles [3] with a propagation model for human actions we become able to re-formulate the 3D tracking problem instead as a problem of recognizing the action itself (incl. its parameters). In other words, instead of having to estimate the high-dimensional parameter vector of the human body model, we sample the action and its parameters in the low-dimensional *action space*. By using a generative model for the action representation we can de-

duce the appropriate 3D body pose from the sampled action parameters and compare it to the input data. We call this approach *Tracking in Action Space*. In our work we use parametric hidden Markov models (PHMMs) [27] as the generative action model. While classical HMMs can recognize essentially only a specific trajectory or movement, PHMMs are able to model different parametrizations of a particular movement. For example, while a common HMMs would be able to recognize only one specific pointing action, PHMMs are able to recognize pointing action into different directions [27]. Furthermore, (P)HMMs are generative models which means that for recognizing an observation they compare it with a model of the expected observation. In the experiments in this paper, our action space spans over the human arm actions *pointing*, *reaching*, *pushing*, the corresponding 2D coordinates of *where* to point at or reach to, plus a timing parameter. One might argue that such an approach cannot be used for general 3D body tracking because the action space will always be too limited. However, following the arguments in [22, 21, 6, 13, 7] that human actions are composed out of motor primitives similarly to human speech being composed out of phonemes, we believe that the limited action space considered here can be generalized to the space spanned by the action primitives. Stochastic action grammars could be used as in [12, 6, 7] to model more complex actions. Furthermore, [7] explains how a language for human actions can be generated based on grounded concepts, kinetology, morphology and syntax. For estimating the action and action parameters during tracking we have used classical Bayesian propagation over time which, as we will discuss below, provides an excellent embedding for tracking in action space, including the use of primitives and action grammars. Our key contribution are:

- introduction of *Tracking in Action space* by posing the 3D human pose estimation problem as one of action and action parameter recognition,
- good recovery of 3D pose and action recognition plus action parameters
- is content with monocular video data
- potentials to run in real-time, our implementation runs with just edge features
- concise formulation in particle filtering framework
- concept of approaching action recognition as a context and object dependent problem.

The paper is structured as follows: In Sec. 2 we will discuss the *Tracking in Action Space* in detail. Sec. 2.1 explains the use of parametric HMMs (PHMMs) for modeling the action space and Sec. 2.2 explains details on using the PHMMs for tracking in action space. In Sec. 2.3, we will discuss the

calculation of the observation likelihood for a given image. In Sec. 3, we provide experimental results for synthetic and for real video data, and we conclude with final comments in Sec. 4.

Related work

As it can be seen in the survey [18], early deterministic methods, as gradient based methods, have been overcome by stochastic methods due to problems as depth disambiguations, occlusions, etc. The methods range from the basic particle filtering, as described in [11], to, e.g., belief propagation [9, 16]. Efficient techniques for particle filtering [3, 24, 5] in combination with (simple) motion models [23] to constrain the particle propagation or the state space [25] are investigated since the number of required particles generally scale exponentially with the degree of freedom. Novel frameworks use for example multistage approaches ([16] considers the stages: coarse tracking of people, body part detection and 2D joint location estimation, and 3D pose inference) or implement various constrains ([9] considers the constrains concerning self-occlusion, kinematic, and appearance and uses belief propagation to infer the pose within a graphical model). Contrary to the simple motion models, which roughly approximate the state space used by certain motions, for example as a linear subspace [25], advanced approaches, as for example locally linear embedding allow to learn [4] the intrinsic low dimensional manifold, or aim at the learning of nonlinear dynamic system (NLDS) on motion data, as it is approached through the Gaussian process model developed in [26] in a more efficient way. Interestingly for our context, the learning of NLDS requires a vast amount of training data [26], whereas “classic” HMMs for example can be easily trained but are limited in their expressiveness for complex motions. In our work, we use the *parametric extension* to HMMs and are interested in both the pose estimation and the uncovering the action parameters.

2. Tracking in action space

In this section we want to discuss our approach for *Tracking in Action Space* in detail.

We are starting our discussion looking at the classical Bayesian propagation over time:

$$p_t(\omega_t) \propto \int P(I_t|\omega_t)P(\omega_t|\omega_{t-1})p_{t-1}(\omega_{t-1})d\omega_{t-1}, \quad (1)$$

where I_t is the current observation, $p_t(\omega_t)$ the probability density function (pdf) for the random variable (RV) ω_t at time t , $P(\omega_t|\omega_{t-1})$ the propagation density, and $P(I_t|\omega_t)$ the likelihood measurement of I_t , given ω_t . If applied for 3D human body tracking, the RV ω usually specifies the set of joint angles for some human body model (see, e.g., [3]).

The propagation density is used to constrain the RV ω_t to the most likely pose values at each time step t [5, 23, 3]. In order to compute the likelihood $P(\mathbf{I}_t|\omega_t)$, a human body model is generated using the pose values from ω_t and is then compared with the input image \mathbf{I}_t .

For *tracking in action space* we use the RV ω to control a generative action model with $\omega = (a, \theta, k)$. Here, the parameter a identifies which action it is, θ is a vector specifying the parameters of action a , and k is a timing parameter which specifies the timing within the action model. In our work, we use parametric hidden Markov models (PHMMs) [27] which we will discuss in detail, below. We train the PHMM for each action a on joint location data captured from human performances of the action a . Using one PHMM for each action, the parameter a refers then to the a -th PHMM, the parameter vector θ specifies the parameters for the PHMM, e.g., *where* we point to or grasp at, and the parameter k is discrete and specifies the PHMM state. Then, the likelihood $P(\mathbf{I}_t|\omega_t) = P(\mathbf{I}_t|(a, \theta, k)_t)$ is computed by first using the a -th PHMM to *generate* the joint location of the 3D human body pose for parameters θ and HMM-state k . We can do this because the PHMM is trained on joint location data for the action a , as discussed above. In the second step, these joint angles of the 3D body pose are translated into the corresponding 3D body model which is then projected onto the image plane and compared to the input image \mathbf{I}_t . For computing the observation likelihood, we also make use of the standard deviations of the observation densities of the PHMM. This second step is in principle the same as the likelihood measurement in Eq. (1). The propagation density $P(\omega_t|\omega_{t-1})$ can be considerably simplified. Assuming that a human finishes one action primitive before starting a new one, the action identifier a is constant until an action (primitive) is finished. If we have an action grammar model for complex human actions as in [7, 6] then the corresponding action grammar can be used to control the progression of a . Likewise, the action parameters θ are approximately constant until the action primitive is completed. The timing parameter k changes according to the transition matrix of the HMM.

In the following subsections we will discuss some details of PHMMs (Sec. 2.1), how we use the PHMMs in our tracking scheme (Sec. 2.2) and how we compute the observation likelihood (Sec. 2.3).

2.1. Parametric hidden Markov models

In this section we give a short introduction to parametric hidden Markov models (PHMMs) [27], which are an extension of the hidden Markov model (HMM) [19] through additional parameters. The additional variables allow to model a systematic variation within a class of modeled sequences. For example, for a pointing, reaching or pushing action, the variation is given by the target location (e.g., a

location pointed at).

A hidden Markov model is a generative model. It is a finite state machine extended in a probabilistic manner. For an continuous (P)HMM $\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$, the vector $\boldsymbol{\pi} = (\pi_i)$ and the transition matrix $\mathbf{A} = (a_{ij})$ define the prior state distribution of the initial states i and the transition probability between hidden states. In continuous HMMs, the output of each hidden state is described by a density function $b_i(\mathbf{x})$, which is in our context a multivariate Gaussian density $b_i(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$. The HMM parameters can be estimated through the Baum-Welch algorithm [19] for a set of training sequences.

An output sequence $\mathbf{x} = \mathbf{x}_1 \dots \mathbf{x}_T$ can be drawn from the model by generating step-by-step a state sequence $\mathbf{Q} = q_1 \dots q_T$ with respect to the initial probabilities π_i and the transition probabilities a_{ij} and drawing for each state q_t the output \mathbf{x}_t from the corresponding observation distribution $b_{q_t}(\mathbf{x})$. Generally, there is no unique correspondence between an output sequence \mathbf{X} and a state sequence \mathbf{Q} as different hidden state sequences can generate the same output sequence \mathbf{X} . Since we are interested in the temporal behavior and correspondence between parts of the sequence and the state, we use a left-right model [19] to model the trajectories of different actions. A left-right model allows only transitions from each state to itself or to a state with a greater state index.

The movement trajectories we are considering generally underlie a systematic variation, e.g., the *pointed at* position. A general HMM can handle this only as noise or with a great number of states. A parametric HMM (PHMM) [27] models the systematic variation as a variation of the means of the observation distributions $b_i^\theta(\mathbf{x})$, where $b_i^\theta(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_i^\theta, \boldsymbol{\Sigma}_i)$. The means are functions $\boldsymbol{\mu}_i^\theta = \mathbf{f}_i(\boldsymbol{\theta})$ that are approximated for each state separately in the training process.

In [27] a linear model and a more general nonlinear model is used to model $\mathbf{f}_i(\boldsymbol{\theta})$. In the linear case, each function $\mathbf{f}_i(\boldsymbol{\theta})$ is of the form $\boldsymbol{\mu}_i = \bar{\boldsymbol{\mu}}_i + \mathbf{W}_i \boldsymbol{\theta}$, where the matrix \mathbf{W}_i describes the linear variation. In the more general nonlinear case, a neural network is used for each state i that is trained to approximate a more general nonlinear dependency. For both models, the training procedures are supervised by providing the parametrization $\boldsymbol{\theta}$ for each training sequence. For training in the linear case, an extension of the Baum-Welch approach is used. For the non-linear case, a generalized EM (GEM) procedure was developed. We will denote a PHMM with parameter $\boldsymbol{\theta}$ by λ^θ .

2.2. Action tracking: PHMM-based

In this section we want to discuss the details of using PHMMs for modeling the actions for the action tracking. In our problem scenario we have a set $\mathcal{A} = \{1, \dots, M\}$ of actions, where we have for each action $a \in \mathcal{A}$ a trained

PHMM λ_a^θ . They define each action through the corresponding sequences of human joint settings. On these sequences of joint settings, the PHMMs are trained. E.g. the PHMM for the pointing action is trained on joint location sequence coming from different pointing actions, including pointing actions into different directions. We consider left-right PHMMs with a single multi-variate Gaussian as the observation density $b_i(\mathbf{x}) = b_{a,i}^\theta(\mathbf{x})$ for each state i of λ_a^θ with a rather small covariance.

Our human action model has the following parameters: the value a identifies the PHMM λ_a . The value k specifies the hidden state, i.e., the progress of the action a . The parametrization θ (e.g., a pointing location) of the PHMM λ_a^θ is used as the parameter for the observation functions $b_{a,i}^\theta$. Hence, we have defined our random space over the tracking parameter $\omega_t = (a, \theta, k)_t$.

In order to *generate* an action using a PHMM one simple way is to sample the PHMM: the first state $k_{t=0}$ is drawn according to the initial distribution π_a . At each time step the state change is governed by the probability mass function $P(k_t|k_{t-1})$ specified by the transition matrix A_a . The actual synthesis is then done by sampling from the observation density $b_{a,k_t}^\theta(\mathbf{x})$, parametrized with θ . In principle the likelihood for an observation I_t and for a given $\omega_t = (a, \theta, k)$ can be computed simply by $P(I_t|\omega_t) = b_{a,k}^\theta(I_t)$ if the observation space is the same as the one $b_{a,k}^\theta$ is defined on. In our case, $P(\mathbf{x}|\omega) = b_{a,k}^\theta(\mathbf{x})$ defines the distribution of joint locations of 3D body poses which generates a corresponding 3D human body model (see Fig. 2, left) which is then matched against the input image I_t :

$$P(I_t|\omega_t) = \int_{\mathbf{x}} P(I_t|\mathbf{x})P(\mathbf{x}|\omega_t)dx. \quad (2)$$

Finally, the propagation density $P(\omega_t|\omega_{t-1})$ is given as follows: k is propagated as mentioned above using A_a^θ , and θ is changed using Brownian motion. The variable a is initially drawn from an even distribution and is then kept in this work constant for each particle. We use a particle-filter approach [11] to estimate $\omega = (a, \theta, k)$. It is worth having a close look at this approach: The entropy of the density $p_t(\omega_t)$ of Eq. 1 reflects the uncertainty of the so far detected parameters. Furthermore, by marginalizing over θ and k , we can compute the posterior probability of each action a (see Fig. 6). And by marginalizing over a and k , we can compute the action parameters, θ , respectively. This is displayed in Fig. 1. The figure shows the progression of the RV ω over time for a pointing action. The red and green lines show the most likely pointing coordinates u and v (for $\theta = (u, v)$). 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 ≈ 60 frames, the correct parameters are recovered. This is about the time when

the arm is fully stretched. In the next section, we will discuss how the observation likelihood $P(I_t|\omega_t)$ is computed.

2.3. Observation model

We use an articulated model of the human body, see Fig. 2 (left), where the kinematic structure is similar to the one used in [3]. Based on this model, we compute the observation function $P(I|\mathbf{x})$ for an arm pose drawn from $P(\mathbf{x}|\omega)$ for a particle ω_i . Here, the arm pose is defined through $\mathbf{x} = (\mathbf{p}, \mathbf{q}, \mathbf{r})$, where \mathbf{p} , \mathbf{q} , and \mathbf{r} are the positions of shoulder, elbow, and finger-tip in \mathbb{R}^3 . The mapping to the model's kinematic is purely trigonometric. However, since the vector \mathbf{x} is drawn from a Gaussian which is an observation density of an HMM, the lengths of the upper arm $|\mathbf{p} - \mathbf{q}|$ and forearm $|\mathbf{q} - \mathbf{r}|$ are not preserved. Generally, we set the finger-tip and shoulder positions of the model as given through \mathbf{r} and \mathbf{p} . The elbow position \mathbf{q} is then corrected with respect to the model's arm lengths through refining \mathbf{q} to the nearest possible point on the plane defined through \mathbf{p} , \mathbf{q} , and \mathbf{r} . The rather unlikely case that $|\mathbf{p} - \mathbf{r}|$ is greater than the overall arm length is mapped on an arm pose, where the stretched arm points from the shoulder position \mathbf{p} in the direction of \mathbf{r} . The computation of the observation function is based on the edge information of the arm silhouette, therefore, the contour \mathcal{C} of the model is extracted from the rendered view for a pose \mathbf{x} . We defined the observation function similar to the method described in [3] on a smoothed edge image (see Fig. 2, right), 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 I , gray values above some threshold are set to 1. The image



Figure 1. The figure shows the progression of the current estimate of the action parameters over time. The estimate parameters u, v are computed as mean of all particles $\omega = (u, v, k)$. The action parameters u, v correspond to the x-, and y-offsets to the center of the active table-top region (x corresponds to the horizontal, y to the deeps direction in Fig. 4). The dotted lines show the standard deviation of the particles.

is finally smoothed with a Gaussian mask, and normalized. This edge image, denoted by G , can be interpreted as a distance to edge image. The value of $1 - G(c)$ of a pixel 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. This distance interpretation is in some sense similar to the edge detection along normals as used in [11], but faster to evaluate.

The observation function is computed by

$$P(\mathbf{I}|\mathbf{x}) = \exp - \frac{1}{2\gamma^2} \frac{1}{|\mathcal{C}|} \sum_{p \in \mathcal{C}} (1 - G(p))^2, \quad (3)$$

where \mathcal{C} is the model's contour and G is the smoothed edge image. A value of $\gamma = 0.15$ turned out to be reasonable in the experiments. An extension to multiple camera views is straight forward:

$$P(\mathbf{I}|\mathbf{x}) = \exp - \frac{1}{2\gamma^2} \sum_i \frac{1}{|\mathcal{C}_i|} \sum_{p \in \mathcal{C}_i} (1 - G_i(p))^2, \quad (4)$$

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

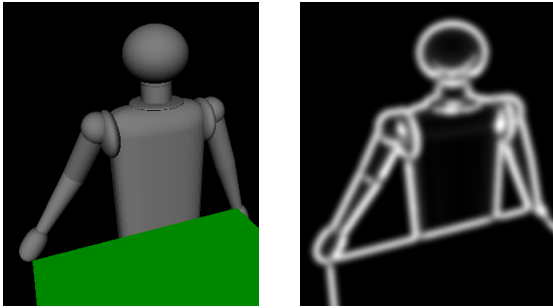


Figure 2. *Left:* We use an articulated human model, where the skeletal structure is modeled by cones and super-quadratics. *Right:* The edge image (here of the model itself) is a smoothed gradient image, serving as a distance to edges image.

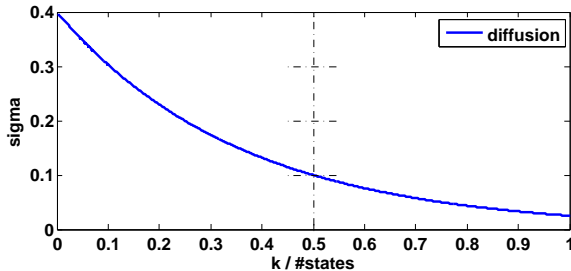


Figure 3. The function $\sigma(k) = 0.4 \cdot \exp(-2 \log_e(1/4) \cdot k / \#states)$ is used to change the diffusion of the parameters (u, v) during the propagation step. The diffusion decreases with increasing PHMM state number k .

3. Experiments

We have evaluated our approach on synthetic data and on monocular video data. For the testing with real data we captured performances of reaching and pointing movements simultaneously with a single video camera and Vicon motion capture system, the video camera was synchronized with the Vicon system. Our experiments were carried out on the monocular video data while the Vicon system provides us with ground truth. In this paper, we focus our attention on human arm pointing, reaching and pushing actions in particular (see tracked sequence, Fig. 4). We call our scenario a *table-top scenario* where the actions are meant to be performed on objects on a table. For each action, we used a linear PHMM $\lambda^{(u,v)}$ trained on 20 demonstration of each action recorded with the Vicon system. The pointing and reaching 2D locations (u, v) at table-top cover a table-

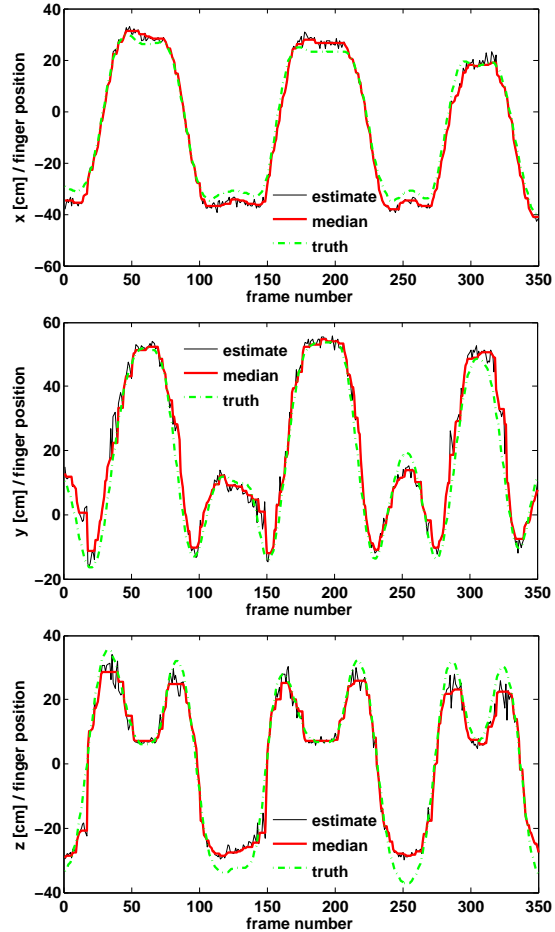


Figure 5. The three plots compare the estimated finger position (red) to the true position (green) over three pointing actions. The position value (red) is the median filtered value of the pose estimates through action tracking (black), the true position is given through marker-based tracking. Here, the x-, y-, and z-position belong to the horizontal, depth, and height directions in Fig. 4.

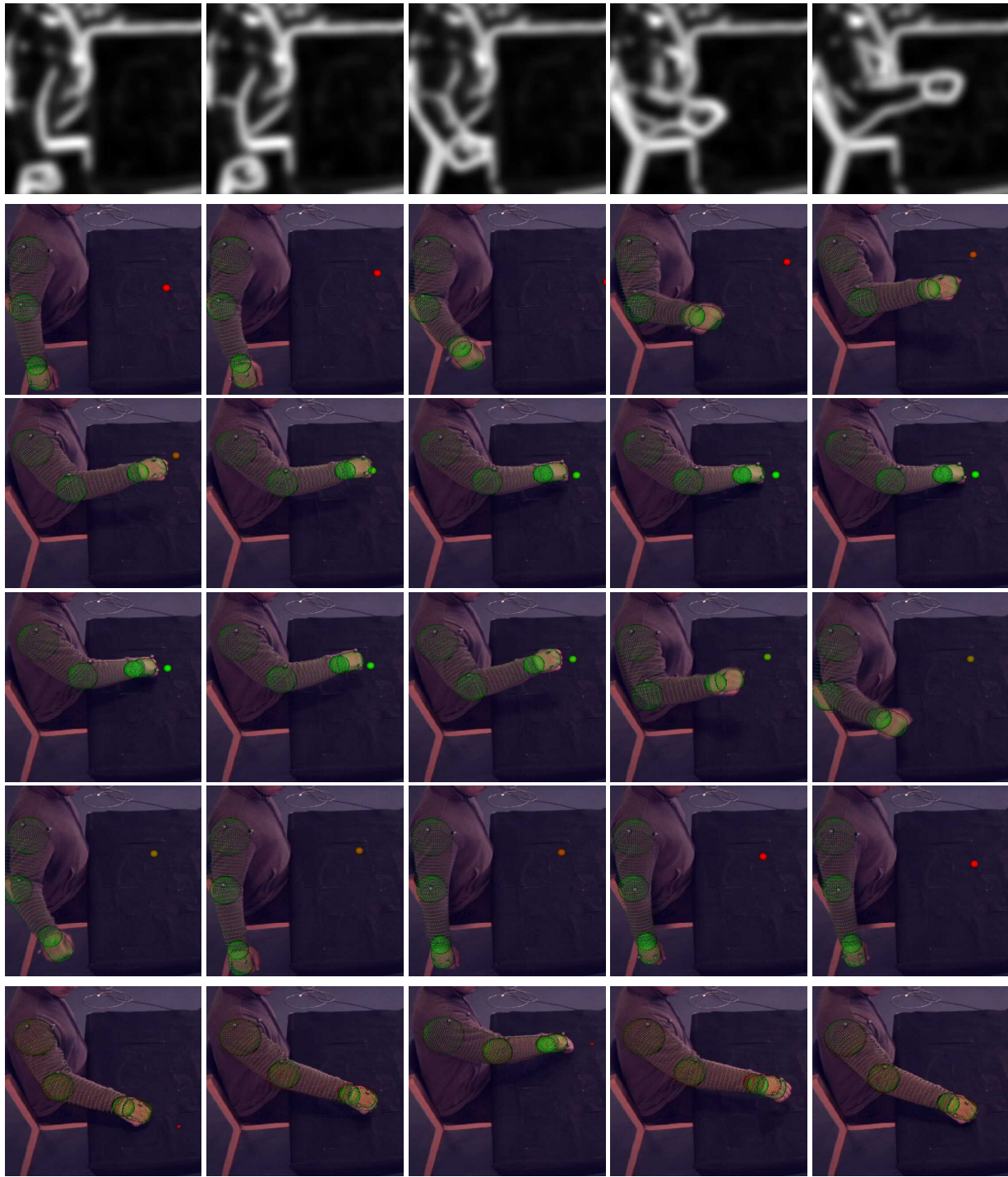


Figure 4. *Pose Estimation through Tracking in Action Space.* In the rows 2–5, a whole pointing action (approaching and withdrawing motion) is shown with the recovered arm pose superimposed. The sequence has about 110 frames, of which every $\approx 6th$ frame is shown. The recovered pose corresponds to the sample/particle which explains the observation in the single monocular view best. The measurement is based only on the edge information in the gradient images (see top row). The estimated action parameters are indicated through the dot on the table: the color of the dot reflects the uncertainty of the indicated location (given through the entropy): a red dot indicates a large uncertainty, a green dot a low one. The last two rows show 10 completed pointing performances where one can see the stretched arm and the recovered pointing position (red dot).

top region of 30cm×80cm; these positions correspond with the parametrization of the corresponding PHMM, and were during the training procedure given by the finger tip location at the time of maximal arm extension. For training we used actions directed approximately to the four outer corners of the table top, with 5 repetitions each. The testing was done using continuous video data of 40 different random locations on the table top. Ground truth was available through the vicon system, the markers on the human were used by the vicon system, but were **not** used for our tracking experiments. We used a large number of HMM states which assured a good predictive model (resolution in time) and small state covariances of the observation densities. The used PHMM is a forward model which allows self-transitions and transitions to the subsequent three states. In order to allow the tracking of repeated performances of the actions, we have defined a very simple action grammar that models a loop. For the particle based tracking in action space we use 400 particles ($\omega_i = (a; u, v; k)$). The propagation over time is done as described in Sec. 2.2. We decrease the diffusion of the u and v during the propagation step in dependence of the HMM state number k and it is governed by $\sigma = \sigma(k)$ as given in Fig. 3. Our argument for the cooling down of the diffusion is that for the first frames the visual evidence for the right u and v is very weak, but as visual evidence increases with time, we inversely reduce the standard deviation.

The sampling and normalization of the image observation are performed as described in [11]; as discussed the observation function is based on the evaluation of the edge information in the monocular video data and the human body model for a particle ω_i .

The images in Fig. 4 show that the arm pose is (visually) very accurately estimated. The following three complicating factors emphasize the capabilities of our *tracking in action space* approach: 1) all information is gathered from a *single monocular* view, 2) we use only a *single feature type* (edge information), and 3) the edge images (especially the first sequence part in Fig. 4 due to the chair) have a lot of clutter, so that the silhouette of the arm is difficult to segment accurately. Besides the pose estimation, one can see in Fig. 4 that the estimation of the action parameters (corresponding to the position indicated by the small colored dot on the table) converges to the true parameters of the action when the arm approaches the table-top.

The quality of the pose estimation over a sequence of several pointing actions is shown in Fig. 5. Here, we compare the positions of the shoulder, elbow, and finger estimated through action tracking to the ground truth positions recorded with the marker-based Vicon system. The route-mean-square error of the three joint positions over the three pointing actions which are plotted in Fig. 5 is 3.3cm, whereas the component-wise average error is just 2.4cm. It

is interesting to note that this errors correlate with the natural variance of the human movements as recorded with the vicon system. This gives a mean error for the recovered table locations of 1.3cm.

frame	Error X	Error Y	Error Z
61	-2.53	-0.53	0.72
195	-3.77	-0.48	-0.36
301	-0.05	-1.70	1.62

Table 1. The table shows the errors in *cm* between the recovered parameters and the ground truth at the specified frames (completed pointing action). Frame numbers here are the same as in Fig. 5.

Despite the good results above, the recognition rate between reaching and grasping was very low. This was due to the fact that these two actions have the same general arm trajectory but differ only in the hand movement. On the other hand, testing on videos showed pointing and pushing actions in random order with at least 40 pointing and 40 pushing actions, the recognition rate was with $\approx 98\%$ as high as expected. Fig. 6 shows the posterior probability for the two actions over time for a test video showing the pointing action: The action label a of a particle $\omega = (a; \theta; k)$ identifies the pointing or pushing action. By marginalizing ω over $\theta = (u, v)$ and k we compute the likelihood of a . The actions are very similar in the beginning. This is also visible in the plot: after 60 frames, the pushing action starts to differ from the observed pointing action and the posterior probability of the pushing action converges.

For the particle filter, we use only 400 particles, the edge features are fast to compute and on a standard workstation with non-threaded code we require presently 3.9s per frame. Ongoing work is to port our approach to CUDA for faster processing on a GPU.

4. Conclusions

We presented a novel concept of *tracking in action space* which combines the aspects of recognition, tracking and prediction based on parametric and time dependent action models. One can argue that this approach is too limited because it is not possible to model all different possible actions each with a PHMM. As our response, the starting point for our approach was (1) the observations that most actions are object and context dependent which means that a) object affordances and b) the scenario and the scene state greatly reduce the set of possible actions, and (2) that according to neuroscientific evidence actions are composed using action primitives and grammars. Thus, even though the number of possible actions at any time is indeed large, only a small number of actions actually *can* appear, with a certain likelihood. Furthermore, all these possibly appearing actions do not need to be modeled each with a PHMM. Instead, it is sufficient to identify the building blocks of these action,

i.e., the action primitives, to model only those with PHMMs and to then compose the complete actions out of these action primitive PHMMs. In the experiments, we have focused on arm actions as these were the ones needed in our human-robot communication scenario. But we believe that our approach should scale well to more body parts and more complex actions. In our future work we are going to consider different actions and the use of stochastic grammars in order to allow proper concatenation of actions as, e.g., reach for an object, move the object, withdraw arm etc. Extension to, e.g., dual arm actions in combination with upper body tracking is also ongoing work.

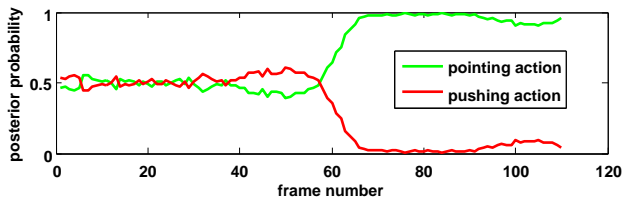


Figure 6. The plot shows the posterior probability of the two actions *point* and *push* over time.

References

- [1] T. Asfour, K. Welke, A. Ude, P. Azad, J. Hoefft, and R. Dillmann. Perceiving objects and movemets to generate actions on a humanoid robot. In *Proc. Workshop: From features to actions – Unifying perspectives in compnaitonal and robot vision, ICRA, Rome, Italy, April 2007*. 1
- [2] D. Bub and M. Masson. Gestural knowledge evoked by objects as part of conceptual representations. *Aphasiology*, 20:11121124, 2006. 1
- [3] J. Deutscher, A. Blake, and I. Reid. Articulated body motion capture by annealed particle filtering. In *CVPR*, volume 2, pages 126–133 vol.2, 2000. 1, 2, 3, 4
- [4] A. Elgammal and C.-S. Lee. Inferring 3d body pose from silhouettes using activity manifold learning. In *CVPR*, 2004. 2
- [5] J. Gall, J. Patthoff, C. Schnoerr, B. Rosenhahn, and H.-P. Seidel. Interacting and annealing particle filters: Mathematics and recipe for applications. *Jounral of Mathematical Imaging and Vision*, 28(1):1–18, May 2007. 2, 3
- [6] G. Guerra-Filho and Y. Aloimonos. A sensory-motor language for human activity understanding. *HUMANOIDS*, 2006. 2, 3
- [7] G. Guerra-Filho and Y. Aloimonos. A language for human action. *Computer*, 40(5):42–51, 2007. 2, 3
- [8] A. Gupta and L. Davis. Objects in action: An approach for combining action understanding and object perception. In *CVPR*, 2007. 1
- [9] A. Gupta, A. Mittal, and L. S. Davis. Constraint integration for efficient multiview pose estimation with self-occlusions. *PAMI*, 30(3):493–506, 2008. 2
- [10] H. B. Helbig, M. Graf, and M. Kiefer. The role of action representation in visual object. *Experimental Brain Research*, 174:221–228, 2006. 1
- [11] M. Isard and A. Blake. Condensation – conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28, 1998. 2, 4, 5, 7
- [12] Y. Ivanov and A. Bobick. Recognition of Visual Activities and Interactions by Stochastic Parsing. *PAMI*, 22(8):852–872, 2000. 2
- [13] O. Jenkins and M. Mataric. Deriving Action and Behavior Primitives from Human Motion Data. In *International Conference on Intelligent Robots and Systems*, pages 2551–2556, Lausanne, Switzerland, Sept.30 – Oct.4, 2002. 2
- [14] H. Kjellstroem, J. Romero, D. Martinez, and D. Kragic. Simultaneous visual recognition of manipulation actions and manipulated objects. In *ECCV*, volume 2, pages 336–349, 2008. 1
- [15] V. Krüger, D. Kragic, A. Ude, and C. Geib. The meaning of action: A review on action recognition and mapping. *Advanced Robotics*, 21(13):1473–1501, 2007. 1
- [16] M. Lee and R. Nevatia. Human pose tracking in monocular sequences using multilevel structured models. *PAMI*, 31:27–38, 2009. 1, 2
- [17] F. Lv and R. Nevatia. Recognition and Segmentation of 3-D Human Action Using HMM and Multi-class AdaBoost. In *ECCV*, Graz, Austria, May 7-13, 2006. 1
- [18] T. Moeslund, A. Hilton, and V. Krueger. A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, 104(2-3):90–127, 2006. 2
- [19] L. R. Rabiner and B. H. Juang. An introduction to hidden Markov models. *IEEE ASSP Magazine*, pages 4–15, January 1986. 3
- [20] H. Ren, G. Xu, and S. Kee. Subject-independent Natural Action Recognition. In *International Conference on Automatic Face and Gesture Recognition*, Seoul, Korea, May 17-19, 2004. 1
- [21] G. Rizzolatti, L. Fogassi, and V. Gallese. Parietal Cortex: from Sight to Action. *Current Opinion in Neurobiology*, 7:562–567, 1997. 2
- [22] G. Rizzolatti, L. Fogassi, and V. Gallese. Neurophysiological Mechanisms Underlying the Understanding and Imitation of Action. *Nature Reviews*, 2:661–670, Sept. 2001. 2
- [23] H. Sidenbladh, M. Black, and L. Sigal. Implicit Probabilistic Models of Human Motion for Synthesis and Tracking. In *ECCV*, Copenhagen, Denmark, 2002. 1, 2, 3
- [24] C. Sminchisescu and B. Triggs. Covariance Scaled Sampling for Monocular 3D Body Tracking. In *CVPR*, Kauai Marriott, Hawaii, December 9-14 2001. 1, 2
- [25] R. Urtasun and P. Fua. 3d human body tracking using deterministic temporal motion models. In *ECCV*, pages 92–106, 2004. 2
- [26] J. M. Wang, D. J. Fleet, and A. Hertzmann. Correction to “gaussian process dynamical models for human motion”. *PAMI*, 30(6):1118, 2008. 2
- [27] A. D. Wilson and A. F. Bobick. Parametric hidden markov models for gesture recognition. *PAMI*, 21(9):884–900, 1999. 1, 2, 3

- [28] T. Xiang and S. Gong. Beyond Tracking: Modelling Action and Understanding Behavior. *International Journal of Computer Vision*, 67(1):21–51, 2006. [1](#)

Visual Object-Action Recognition: Inferring Object Affordances from Human Demonstration[☆]

Hedvig Kjellström*, Javier Romero, Danica Kragić

*Computational Vision and Active Perception Lab
Centre for Autonomous Systems
School of Computer Science and Communication
KTH, SE-100 44 Stockholm, Sweden*

Abstract

This paper investigates object categorization according to function, i.e., learning the *affordances* of objects from human demonstration. Object affordances (functionality) are inferred from observations of humans using the objects in different types of actions. The intended application is learning from demonstration, in which a robot learns to employ objects in household tasks, from observing a human performing the same tasks with the objects. We present a method for categorizing manipulated objects and human manipulation actions in context of each other. The method is able to simultaneously segment and classify human hand actions, and detect and classify the objects involved in the action. Experiments show that the contextual information greatly improves the classification of both objects and actions.

Key words: object recognition, action recognition, contextual recognition, object affordances, learning from demonstration

[☆]An early version of this article appears in (Kjellström et al., 2008).

*Corresponding author. Previously Hedvig Sidenbladh.

Email addresses: hedvig@kth.se (Hedvig Kjellström), jrgn@kth.se (Javier Romero), dani@kth.se (Danica Kragić)

URL: www.csc.kth.se/~hedvig (Hedvig Kjellström), www.csc.kth.se/~jrgn (Javier Romero), www.csc.kth.se/~danik (Danica Kragić)

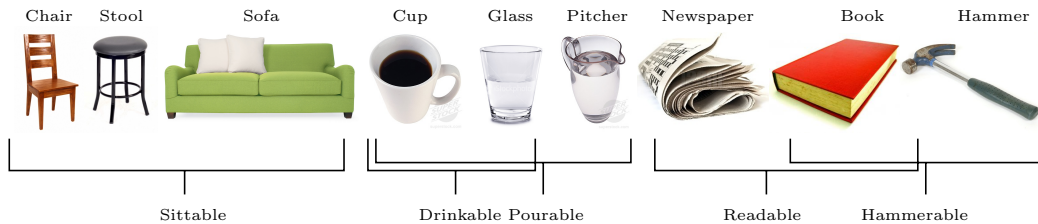


Figure 1: Representing objects in terms of functionality and affordances. Top: Semantic, appearance based categories. Bottom: Functional, affordance based categories.

1. Introduction

In recent years, a tremendous research effort has been made in the area of visual object categorization (Fei-Fei et al., 2009), leading to methods with impressive performance on very difficult images. Object classes are typically semantic and appearance-based; common examples are cups, toys, bikes, cars, and trees.

For certain classes of applications, e.g., in Robotics, it is however more meaningful to categorize objects according to their *function* (Rivlin et al., 1995; Stark and Bowyer, 1996; Stark et al., 2008). Both a chair, a sofa, and a stool might be categorized as "sittable", and a cup might be categorized both as "drinkable" and "pourable" (Figure 1).

To a certain extent, functional object properties can be extracted from visual information. However, there are functional properties that can not be observed visually from a single image, such as temperature, flexibility and weight. We propose to learn functional properties of objects from video sequences where a human perform actions involving the objects.

The application we are focusing on is robot learning from demonstration (Billard et al., 2008), also denoted imitation learning (Schaal, 1999). With imitation we here do not mean blind reproduction of the movements of all body parts; rather, we mean observing an action and its effect on the world, and performing an own action that has the same effect (Montesano et al., 2008).

We here formulate the problem of learning from demonstration as one of learning the *affordances* of objects. Introduced as a concept by Gibson (1979), affordances are properties of the environment that *afford* a certain action to be performed by a human or an animal. Here we study affordances of objects involved in human manipulation actions.

An affordance is an intrinsic property of an object, allowing an action to

be performed with the object. The affordance also depends on the embodiment of the agent performing the action. For example, a human can use a knife to chop an onion, while a dog can not. Hence, the knife *affords* onion chopping to a human but not to a dog.

From this we can conclude that an agent (robot) learning object affordances from a human must have an embodiment similar to a human: Two arms with the same reaching range and at the same height as human arms, and human-like hands, which can manipulate objects in the same way as human hands. In other words, the agent should be humanoid. This paper does not further treat robotic manipulation; we instead concentrate on the learning of object affordances from human demonstration.

Manipulation actions, i.e. hand actions for picking up objects, doing something with them and putting them down again, is an important class of hand activity not well studied in computer vision. An important cue to the class of a manipulation action is the object handled; for example, seeing a human bring a cup towards his/her face brings us to believe that he/she is drinking, without actually seeing the fluid. Similarly, a strong cue to the class of the object involved is the action; for example, a cup is to some extent defined as something you drink from. Therefore, it is beneficial to *simultaneously* recognize manipulation actions and manipulated objects.

Only one-hand actions are considered here, although this is not a limitation to the method in a formal sense. From a video sequence of the action, the human hand position and articulation in 3D are reconstructed and tracked using an example based method (Romero et al., 2010). The action state space in each frame is the hand orientation and velocity as well as the finger joint angles, representing the hand shape.

Objects in this application are "graspable", i.e., fairly rigid, so shape is a good object descriptor. Objects are therefore detected in the neighborhood of the hand using a sliding window approach with a histogram of gradients (HOG) (Dalal and Triggs, 2005; Freeman and Roth, 1995; Shakhnarovich et al., 2003) representation and an SVM classifier (Cristianini and Shawe-Taylor, 2000). Section 3 further describes the feature extraction.

There are implicit, complex interdependencies in the object and action data. The object detection is affected by occlusion and shading from the human hand. Similarly, the hand shape depends on the size, shape and surface structure of the object in the hand. These dependencies are difficult to model, which leads us to use a discriminative sequential classifier, conditional random fields (CRF) (Lafferty et al., 2001), that does not model the data

generation process.

On a semantic level, there are also action-object dependencies of the type `drink-cup`, `drink-glass`, `write-pen`, `draw-pen` and so on, which can be explicitly modeled within the CRF framework. The action-object dependence is modeled on a per-frame basis using a factorial CRF (FCRF) (Sutton et al., 2004). This is detailed in Section 4.

A manipulation action is here thought of as beginning with the picking-up of an object and ending with the putting-down of that object – also referred to as "manipulation segments" (Zöllner et al., 2005). However, two such actions with the same object might also follow each other directly, without any putting-down and picking-up events in between. The FCRF enables simultaneous object-action classification and temporal action segmentation, removing the need for special tags (e.g., grasping or reaching motions) in the beginning and end of each action (Gupta et al., to appear; Serrano Vicente et al., 2007). This is further discussed in Section 5.

The concept of contextual object-action recognition, as well as the recognition method chosen in this paper, are experimentally evaluated in Section 6. From the experiments it can be concluded that both action and object recognition benefit from the contextual information.

2. Related Work

Visual recognition, especially object recognition (Fei-Fei et al., 2009), is a vast area of research and can be regarded as one of the core problems in Computer Vision. We do not make an attempt to review the whole field, but focus on learning of object affordances and contextual recognition.

2.1. Learning of Object Affordances and Learning from Demonstration

The concept of affordances (Gibson, 1979) has come in focus lately within the Cognitive Vision and Robotics communities. While many other papers on affordances, e.g. (Bohg and Kragić, 2009; Saxena et al., 2008; Stark et al., 2008), concentrate on robotic grasping, we here focus on more composite, higher-level actions, which typically involve grasping as a sub-component.

The embodied/cognitive vision approach to affordance learning consists of an agent acting upon objects in the environment and observing the reaction. In (Fitzpatrick et al., 2003), a robot pushes, pokes, pulls and grasps objects with its end-effector, thereby learning about rolling, sliding etc. Montesano et al. (2008) notes that an affordance can be described by the three

interdependent entities of action, object, and effect. A robot first learns a set of affordances by its own exploration of the environment using preprogrammed basic motor skills. It can then imitate a human action, not by mimicking the action itself, but rather observing the effect and then selecting an own action that will have the same effect on the current object. The difference to our imitation learning is that we also learn the object affordances themselves from human demonstration.

To a certain degree, affordances can be observed in images. In three recent works, (Bohg and Kragić, 2009; Saxena et al., 2008; Stark et al., 2008), relations between visual cues and grasping affordances are learned from training data. In (Stark et al., 2008), object grasping areas are extracted from short videos of humans interacting with the objects, while in (Bohg and Kragić, 2009; Saxena et al., 2008) a large set of 2D object views are labeled with grasping points. Early work on functional object recognition (Rivlin et al., 1995; Stark and Bowyer, 1996) can be seen as a first step towards recognizing affordances from images. Objects are there modeled in terms of their functional parts, such as handle, hammer-head etc (Rivlin et al., 1995), or by reasoning about shape in association to function (Stark and Bowyer, 1996).

The robot can also learn through visually observing another agent – for example, a human – making use of object affordances. This is the approach we take in this article. A similar idea is also exploited in (Veloso et al., 2005). However, while they study whole-body activities such as **sitting-on-chair** and **walking-through-door**, we focus on manipulation actions, involving the human hands and arms.

Affordances relate to the concept of *task oriented vision* (Ikeuchi and Hebert, 1992; Miura and Ikeuchi, 1995). According to this notion, a Computer Vision system should be designed with a specific task in mind. This is put in contrast to Marr’s (1982) general purpose vision paradigm. The intended task will affect what aspects of the world are perceived and processed, as well as the design of the whole system. The inspiration comes from human vision; psychophysical experiments (Triesch et al., 2003) indicate that humans indeed only perceive the aspects of the world relevant to the task at hand.

Ikeuchi and Hebert (1992) exemplify task oriented vision by comparing two systems designed to solve two different grasping tasks. Miura and Ikeuchi (1995) point out that knowledge of the task should be used to ensure that only relevant information is extracted. Although the rapid development of

computational power has made this issue less critical today, it is still valid. In our learning from human demonstration method, the robot only includes objects near the human hand in the action-object analysis, rather than trying to model all objects in the scene.

2.2. Contextual Recognition

There has been a large recent interest in contextual recognition within the Computer Vision community.

One type of contextual information for object detection and recognition is text. The caption of an image says something about what objects can be expected in it. When labeling images according to object content, any captions should therefore be taken into account. Caption-guided object detection can be used to segment the image into object regions and associate them with object labels (Carbonetto et al., 2004), or to automatically label or cluster a large set of unlabeled images with captions given a smaller set of labeled images with captions (Quattoni et al., 2007). Prepositions and comparative adjectives can also be used to discover spatial relations between objects in the image (Gupta and Davis, 2008).

In (Torralba, 2003; Murphy et al., 2003; Torralba et al., 2004), the scene itself, the "gist" of the image, is used to guide object recognition. The scene itself is a strong prior cue as to which objects can be expected and where they are most likely to be found. Similarly, in (Marszalek et al., 2009), actions and events are recognized in movies in context of the scene. Events can even be recognized from single images (Li and Fei-Fei, 2007), if object and scene context is exploited.

Object recognition can also be guided by observations of human interaction with the objects. Moore et al. (1999) provide a Bayesian framework for recognizing objects based on contextual information from other objects, human actions being performed on the object, and the scene. In (Peursum et al., 2005), human actions are used to infer object class. Reversely, recognition of manipulation actions can be guided by knowledge about the objects involved. Wu et al. (2007) represent kitchen activity solely in terms of the sequence of objects in contact with the hand during the activity. These approaches all relate to the presently presented, with the addition that we perform simultaneous recognition of actions and objects in context of each other.

The idea of simultaneous object-action recognition has been exploited before. In (Filipovych and Ribeiro, 2008), primitive actor-object interac-

tions such as grasp cup, touch spoon, are learned from video. We model more high-level actions, which might involve grasping, touching etc. Gupta et al. (to appear) use an approach similar to ours to recognize actions and objects in context of each other. The main difference, apart from our affordance framework, is that they segment manipulation action by detecting of reaching motion. We instead incorporate the temporal segmentation into the recognition using a conditional random field. This enables us to recognize actions following each other, without any special delimiter actions such as reaching, putting down or picking up. Furthermore, they focus on upper-body or whole body actions while we study hand manipulation actions.

3. Features for Classification

For our purposes, extraction of object and action features could be done in a variety of ways (Fei-Fei et al., 2009; Moeslund et al., 2006) depending on the purpose of the feature extraction. As opposed to many other action recognition applications, it is here necessary to obtain the location of the human hand to find the object or objects involved. Furthermore, it should be possible to recreate the recognized action with a robot, which means that the hand position, orientation and articulated pose should be retrievable from the action representation. This is further discussed in Section 3.2 below.

3.1. Object Features

Different actions involve different number of objects. For example, the action `pour` involve two containers, one to pour from and one to pour to, while the action `sit` involves one piece of furniture. (We do not here separate between tools and other objects; this is further discussed in Section 7.) The object state o_t therefore encodes both the number and the classes of objects involved in the action at time t .

The object state is approximated by a vector x_t^o where each element is the detection probabilities for each object class respectively. At this pre-processing stage, objects are categorized according to appearance into 6 semantic categories of the type shown at the top row of Figure 1: `book`, `magazine`, `hammer`, `box`, `cup`, and `pitcher`. Section 5 describes how these object classes are grouped according to observed human use.

All objects of the known range of classes in the neighborhood of the human hand are detected. We use sliding window detectors, one for each object class. The detector for a certain object class searches over image

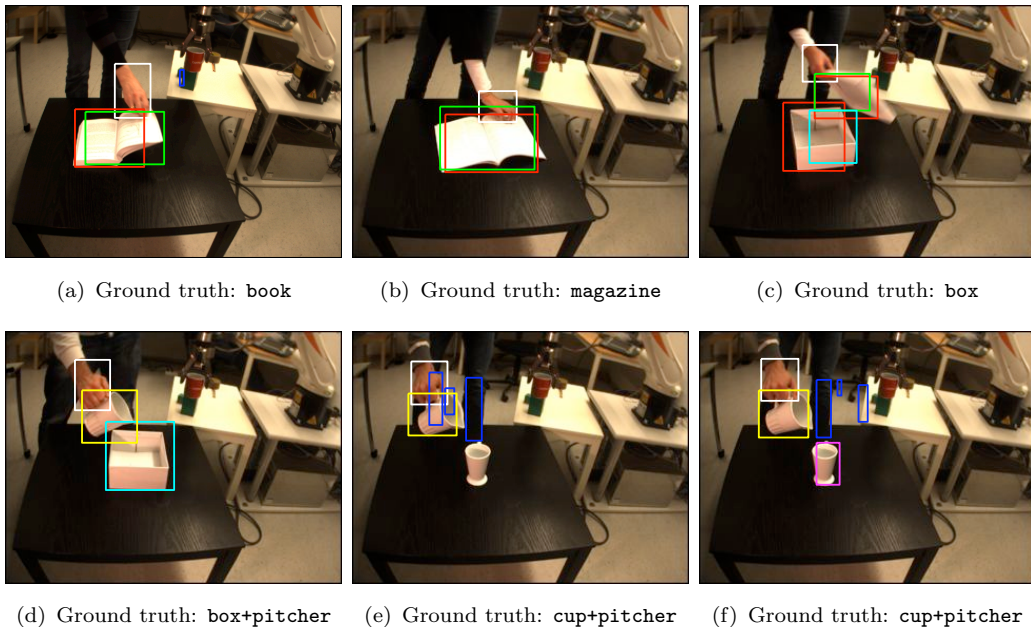


Figure 2: Detection of objects in human action sequences. (a,b) Books and Magazines are difficult to distinguish visually. (c) A Box (and its lid) look similar to a closed Book or a Magazine. (a,e,f) Objects with non-discriminatory appearance (here Hammer) are sometimes "hallucinated". (e) Objects (here Cup) are sometimes missed. (a-f) Color coding: \bullet = book, \bullet = magazine, \bullet = hammer, \bullet = box, \bullet = cup, \bullet = pitcher. *This figure is best viewed in color.*

position, scale and height/width ratio in the image plane, in the vicinity of the human hand (see Section 3.2). Each window is classified as object or background using a two-class support vector machine (SVM) (Cristianini and Shawe-Taylor, 2000). The features used in SVM classification are the histograms of oriented gradients (HOG) (Dalal and Triggs, 2005) extracted from the window. Figure 2 shows example detections of the 6 different object classes.

A sequence of object detections is denoted $\mathbf{x}^o = \{x_t^o\}, t = 1, \dots, T$, where x_t^o is a vector of detection probabilities for the 6 known object classes. Similarly, $\mathbf{o} = \{o_t\}, t = 1, \dots, T$ denotes the corresponding object state, where o_t is an integer value, indicating which combination of objects is involved in the action at time t .

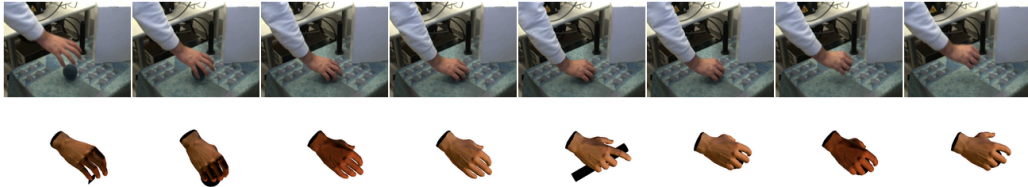


Figure 3: Reconstruction and tracking of 3D articulated hand pose. Top: original frame. Bottom: reconstructed view (object not included in the pose description).

3.2. Action Features

A human manipulation action is to a very large degree described by the articulated motion of the hand. We therefore use the hand pose reconstruction method in (Romero et al., 2010) to reconstruct and track the articulated motion of the hand in 3D.

The method is example based. In each timestep, the hand is first segmented from the image using skin color. The appearance of the hand is compared to a large database (on the order of 10^6 examples) of synthetic hand views tagged with articulated pose and orientation. Temporal pose consistency is enforced in the reconstruction. Moreover, typical occlusions from objects in the hand are modeled by including object occlusions in some examples in the database.

A reconstruction example can be seen in Figure 3. The reconstruction is quite crude with angular errors of 10-20%. However, the method is functioning in real-time and is very robust to temporary tracking failure. Therefore, it is suitable for our purposes, where speed and robustness is more critical than accuracy.

To provide position invariance the global velocity of the hand is encoded, rather than the global position itself. A hand pose is thus defined by global velocity, global orientation, and joint angles. In a manner similar to the object feature extraction, the hand pose at time t is classified as being part of the actions **open**, **hammer**, **pour**, or as not involved in any particular action. The classification of a hand pose as being part of an action or no action is separate from the others, using a two-class SVM for each action.

In the following, a sequence action classifications is denoted $\mathbf{x}^a = \{x_t^a\}, t = 1, \dots, T$, where x_t^a is a vector of classification probabilities for the three action classes. The corresponding sequence of action classes is denoted $\mathbf{a} = \{a_t\}, t = 1, \dots, T$, where a_t is an integer value indicating action class.

3.3. Correlation Between Object and Action Features

The temporal classifier described in Section 4 models explicit semantic dependencies between manipulation actions and the manipulated objects. However, there are also dependencies on the feature level.

The shape of the hand encoded in x_t^a gives cues about the object as well, since humans grasp different types of objects differently, due to object function, shape, weight and surface properties. Similarly, the object detection results encoded in x_t^o is affected by the hand shape since the hand occludes the object in some cases. Furthermore, there are temporal dependencies: x_{t-1}^a and x_t^a are correlated as are x_{t-1}^o and x_t^o . This correlation within the data is *implicit* and difficult to model accurately, but should be taken into account when modeling the simultaneous action-object recognition.

4. Classification of Object-Action Data

Since we can expect complex dependencies within our action data \mathbf{x}^a and object data \mathbf{x}^o over time, a discriminative classifier which does not model the data generation process is preferable over a generative sequential classifier like a hidden Markov model (HMM) (Rabiner, 1989). We thus employ conditional random fields (CRF:s) (Lafferty et al., 2001) which are undirected graphical models that represent a set of state variables \mathbf{y} , distributed according to a graph \mathcal{G} , and conditioned on a set of measurements \mathbf{x} . CRF:s have previously been used to model human activity, e.g. in (Sminchisescu et al., 2006).

Let $C = \{\{\mathbf{y}_c, \mathbf{x}_c\}\}$ be the set of cliques in \mathcal{G} . Then,

$$P(\mathbf{y}|\mathbf{x}; \theta) = \frac{1}{Z(\mathbf{x})} \prod_{c \in C} \Phi(\mathbf{y}_c, \mathbf{x}_c; \theta_c) \quad (1)$$

where Φ is a potential function parameterized by θ as

$$\Phi(\mathbf{y}_c, \mathbf{x}_c; \theta_c) = e^{\sum_k \theta_{c,k} f_k(\mathbf{y}_c, \mathbf{x}_c)} \quad (2)$$

and $Z(\mathbf{x}) = \sum_{\mathbf{y}} \prod_{c \in C} \Phi(\mathbf{y}_c, \mathbf{x}_c; \theta_c)$ is a normalizing factor. The feature functions $\{f_k\}$ are given, and training the CRF means setting the weights θ , e.g., using belief propagation (Lafferty et al., 2001).

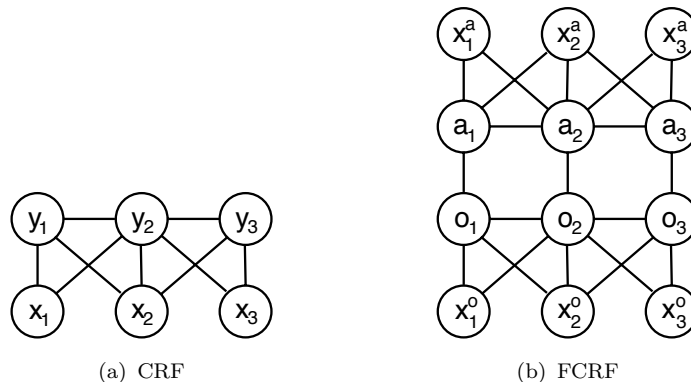


Figure 4: CRF structures. (a) Linear-chain CRF (Lafferty, McCallum and Pereira 2001), used for action or object recognition. (b) Factorial CRF (Sutton, Rohanimanesh and McCallum 2004), used for simultaneous object-action recognition.

4.1. Linear-Chain CRF

For linear-chain data (for example a sequence of object or action features and labels), $\mathbf{y} = \{y_t\}$ and $\mathbf{x} = \{x_t\}$, $t = 1, \dots, T$ as shown in Figure 4a. This means that the cliques are the edges of the model, which gives

$$P(\mathbf{y}|\mathbf{x}; \theta) = \frac{1}{Z(\mathbf{x})} \prod_{t=2}^T \Phi(y_{t-1}, y_t, \mathbf{x}; \theta_t) \quad (3)$$

with a potential function

$$\Phi(y_{t-1}, y_t, \mathbf{x}; \theta_t) = e^{\sum_k \theta_{t,k} f_k(y_{t-1}, y_t, \mathbf{x})}. \quad (4)$$

Each state y_t can depend on the whole observation sequence \mathbf{x} – or any subpart of it, e.g. the sequence $\{x_{t-\mathcal{C}}, \dots, x_{t+\mathcal{C}}\}$, \mathcal{C} being the *connectivity* of the model.

4.2. Factorial CRF

In Section 3.3 we argue that there are correlations between action observations \mathbf{x}^a and object observations \mathbf{x}^o implicit in the data. We make use of this correlation on the data level by not imposing a simplified model on the data generation process and instead using a discriminative classifier, CRF. However, there is also an *explicit*, semantic correlation between actions and objects on the label level, as discussed in the introduction. This correlation can be modeled using a factorial CRF (FCRF) (Sutton et al., 2004). Figure

4b shows an FCRF with two states, action class a_t and object class o_t , for three time-steps $t = 1, 2, 3$. The cliques in this model are the within-chain edges $\{a_{t-1}, a_t\}$ and $\{o_{t-1}, o_t\}$, and the between-chain edges $\{a_t, o_t\}$. The probability of \mathbf{a} and \mathbf{o} is thus defined as

$$P(\mathbf{a}, \mathbf{o} | \mathbf{x}; \theta) = \frac{1}{Z(\mathbf{x})} \prod_{t=1}^T \Phi(a_t, o_t, \mathbf{x}; \theta_t) \prod_{t=2}^T \Phi(a_{t-1}, a_t, \mathbf{x}; \theta_{a,t}) \Phi(o_{t-1}, o_t, \mathbf{x}; \theta_{o,t}) . \quad (5)$$

The weights θ are obtained during training, e.g., using loopy belief propagation (Sutton et al., 2004).

5. Object-Action Recognition using CRF:s

Using the approach described above, the actions and objects in a stereo sequence of human activity can be both temporally segmented and classified, using a CRF in a sliding window manner over time.

An FRCF structure of length $T = 3$ is trained with \mathbf{X} object-action patterns $(\mathbf{o}, \mathbf{x}^o, \mathbf{a}, \mathbf{x}^a)$, also of length $T = 3$, involving \mathbf{Y} different action classes and \mathbf{Z} different object classes.

A new sequence $(\boldsymbol{\chi}^\omega, \boldsymbol{\chi}^\alpha)$ of length τ can now be segmented and classified using this model. For each time step $t = 2, \dots, \tau - 1$, the pattern $(\chi_{t-1}^\omega, \chi_t^\omega, \chi_{t+1}^\omega, \chi_{t-1}^\alpha, \chi_t^\alpha, \chi_{t+1}^\alpha)$ renders the classification ω_t, α_t .

Objects can also be ordered into affordance categories using correlation information extracted from the training data (\mathbf{o}, \mathbf{a}) . This is represented with a correlation matrix \mathbf{C} where element C_{ij} indicates the degree to which object class i can be used to perform action j .

6. Experiments

The feature extraction and classifiers were implemented in Matlab, using the LibSVM toolbox (Chang and Lin, 2001) and the CRF toolbox by Murphy (2004). The object and action feature extraction methods described in Section 3 were first evaluated (Sections 6.1 and 6.2). We then evaluated the temporal object-action segmentation and classification. This is described in Section 6.3.



Figure 5: The 50 instances in the normalized-uniform NORB dataset (LeCun et al., 2004) (for one lighting condition, elevation, azimuth each). Training data left, test data right.

6.1. Evaluation of Object Classifier

HOG has previously been shown (Dalal and Triggs, 2005) to be a good feature representation, since it allows for high intra-class variability (differences between class instances, lighting and pose variation, etc.) while being discriminant with respect to inter-class variability. To verify this, we evaluated the HOG- and SVM-based classifier which is the basis of the sliding window object detector described in Section 3.1.

We first experimented with the NORB dataset (LeCun et al., 2004), which contains 5 different classes of rigid objects; **animals**, **humans**, **airplanes**, **trucks**, and **cars** with 10 instances of each, 5 for test and 5 for training (Figure 5). The database contains stereo views of each object from 18 dif-

Table 1: Results on the normalized-uniform NORB dataset, percent error. Left: Classification error percentage compared to methods presented in LeCun et al. (2004). Right: Generalization test; robustness to different amount of jitter in test data (training data unaltered).

	Mono	Stereo			Brightness			Shift			
					± 0	± 10	± 20	± 30	± 3	± 6	± 9
Hist+SVM	6.4	6.2									
Raw+SVM	12.6 (LC)	—		Hist+SVM	6.4	6.4	7.1	8	10.3	18.1	29.2
Conv Net	—	6.6 (LC)		Raw+SVM	12.6 (LC)	15.8	18	21	20.8	35.1	48.6

ferent azimuths and 9 elevations in 6 different lighting conditions. Only the normalized-uniform part of the dataset, designed to test classification performance, was used.

To evaluate the suitability of the HOG feature representation (Section 3.1) for modeling shape *categories*, a five-class SVM was trained with HOGs extracted from the NORB training images. Table 1 left shows the results compared to others. Our HOG + SVM classifier reached the same classification accuracy as a state-of-the-art method for object categorization (LeCun et al., 2004), which indicates that the HOG representation captures the specifics of a shape class, while allowing a significant variability among instances of that class. In comparison, training an SVM on the raw image downsampled to a size of 32×32 led to twice the classification error (a surprisingly good result, as noted in (LeCun et al., 2004), given that the task is object categorization, not instance recognition). Furthermore, we note that the incorporation of stereo does not add much to the accuracy.

Certain robustness towards differences in color and lighting, as well as small position errors of the object detection window, is also desirable. In (LeCun et al., 2004), this was tested by adding "jitter", i.e. small transformations to both the training and test set. However, this arguably tested how the methods performed with a larger test set, rather than how they could handle noise that was not seen before (not present in the training data). Therefore, we did a variant of this experiment where we added jitter to *only the test set* (Table 1 right). First, the overall brightness of each test image was varied. Our HOG feature representation was very robust to this noise, which is expected since it relies solely on the gradient orientations and not on their value. In comparison, the raw image classification error grew much quicker. Then, the test images were shifted vertically and horizontally in a random manner. The feature representation was more sensitive to this noise, but less so than the raw image representation.

We then proceed to evaluate the the performance of HOG and SVM for classification of the 6 object categories described in Section 3.1. We collected a dataset consisting of 4 to 6 different instances of each class (Figure 6a), with 330 views of each instance. Each object instance was grasped and moved by a human in some views, and the deformable objects were deformed (opened, pages flipped etc.) in other views. In each view, a the quadratic bounding box of the object was manually marked in the image. A HOG representation used for training and classification was then computed over this window.

Training and testing were carried out in a jackknife manner, where one

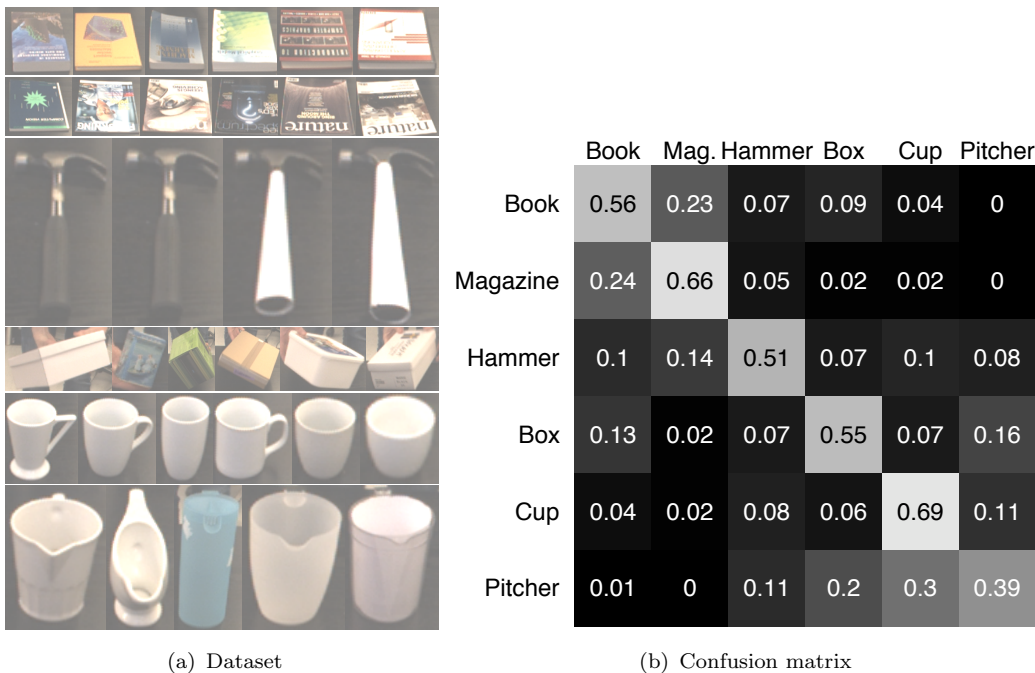


Figure 6: Classification of objects of the 6 classes used in this paper. (a) The 33 instances used in the experiment. 330 views of each instance are provided. Each object is grasped and moved by a human in some views, and the deformable objects are deformed (opened, pages flipped etc.) in other views. (b) Object classification confusion matrix (rows: true classes, columns: classification ratios).

instance at a time of each class were removed from the dataset during training, and used for testing. (Thus, the same instance was never used for both training and testing.) For each training-test data division, a 6-class SVM was trained with the HOG representations. Figure 6b shows the confusion matrix representing the mean classification result.

The experiments with the NORB dataset above indicated that the HOG representation allows certain intra-class variation, while being rich enough to make inter-class discrimination possible. The confusion between **book** and **magazine**, between **book** and **box**, and between **box** and **pitcher** is therefore probably intrinsic to the classes themselves. Books and magazines look very similar both closed and with pages flipped (23% and 24% misclassification) – the main difference is the thickness of the volume. Boxes look like closed books (13% misclassification), but opened books and books with pages flipped do not look like boxes (9% misclassification). Hammers are hard

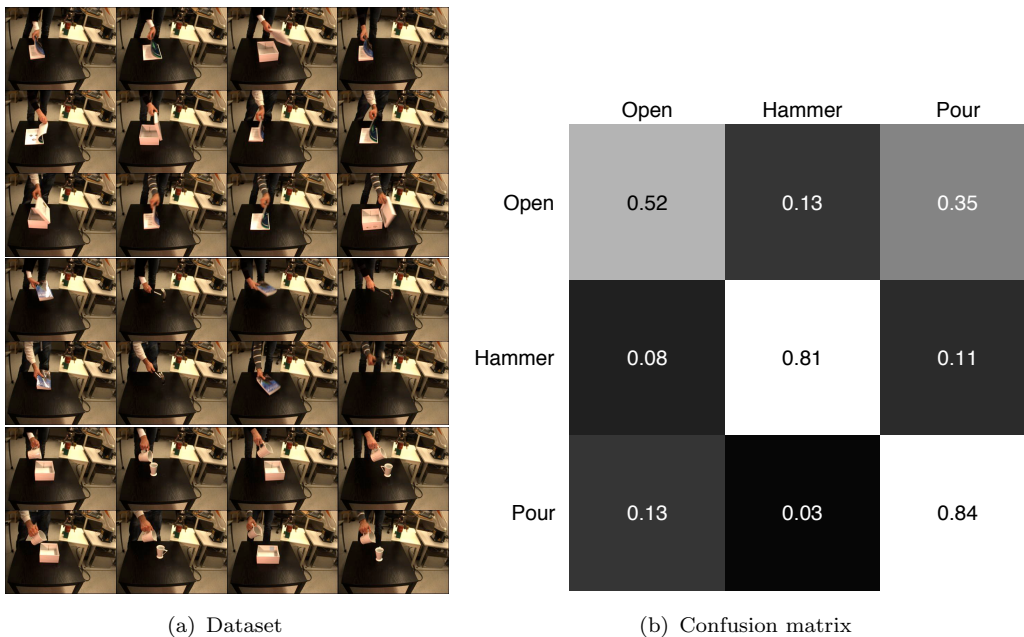


Figure 7: Classification of actions of the three classes used in this paper. (a) The 28 instances used in the experiment. Each instance is a sequence of hand velocities and articulated hand poses, extracted separately from each frame. Each frame is considered as a separate data point. (b) Action classification confusion matrix (rows: true classes, columns: classification ratios).

to model with this classifier (7–14% misclassifications against other classes) since most bounding boxes around hammers contains very much background. Pitchers are often misclassified as boxes (20% misclassification) since many of them look quadratic from a side view, and as cups (30% misclassification) since the shape of those object classes are very similar, with handles and openings at the top.

The classes `book` and `magazine`, and `pitcher` and `cup`, are good examples of object classes that are hard to distinguish from appearance only.

6.2. Evaluation of Action Classifier

The action classification described in Section 3.2 was evaluated in a similar manner. A dataset consisting of 8 to 12 examples of each class of actions, performed with different objects and by four different individuals, was collected (Figure 7a).

Training and testing were done similarly to the above, where the examples from one individual at a time were removed from the training data. Training was done on examples from the three other individuals, and the resulting classifier was evaluated with the examples of the fourth individual. Figure 7b shows the confusion matrix representing the mean classification result.

As with the object classification, there are certain confusions intrinsic to the actions themselves. Most notably, `open` actions are often misclassified as `pour` (35% misclassification), probably since the global hand velocity is similar in these action, and since different individuals configure their hands very differently while opening objects. On the other hand, pouring actions are much more distinct in terms of hand articulation and velocity, and are more seldom misclassified as opening (13% misclassification). Due to its rapid vertical velocity, `hammer` actions are easily recognizable.

The classes `open` and `pour` are good examples of action classes that are hard to distinguish without contextual information, e.g., from objects involved in the action.

6.3. *Classifying Actions and Objects Together*

Experiments with the object and action feature extractors in Sections 6.1 and 6.2 showed certain confusions between classes, which appeared to be intrinsic to the object and action classes themselves. E.g., books and magazines can not always be distinguished by appearance only. However, they afford slightly different ranges of actions, which means that the action observed in connection to the object can be used to constrain the object classification. Similarly, action classification can be constrained by the objects observed in the vicinity of the hand performing the action.

In this experiment, we trained and evaluated an FCRF (Section 5) with the 6 object classes and the three action classes mentioned above, in the 7 following combinations: `open-book`, `open-magazine`, `open-box`, `hammer-with-book`, `hammer-with-hammer`, `pour-from-pitcher-into-box`, and `pour-from-pitcher-into-cup`. Thus, an `open` action can only be observed together with either a `book`, a `magazine` or a `box`; a `hammer` action only together with a `book` or a `hammer`; a `pour` action only together with a `box` and `pitcher` or a `cup` and `pitcher`.

We collected a training set consisting of sequences in which three different individuals performed all 7 object-action combinations, using the first object instance from each class in the evaluation set in Figure 6a. Each frame of the sequences were labeled with object (`none`, `book`, `magazine`, `hammer`, `box`, `cup`,

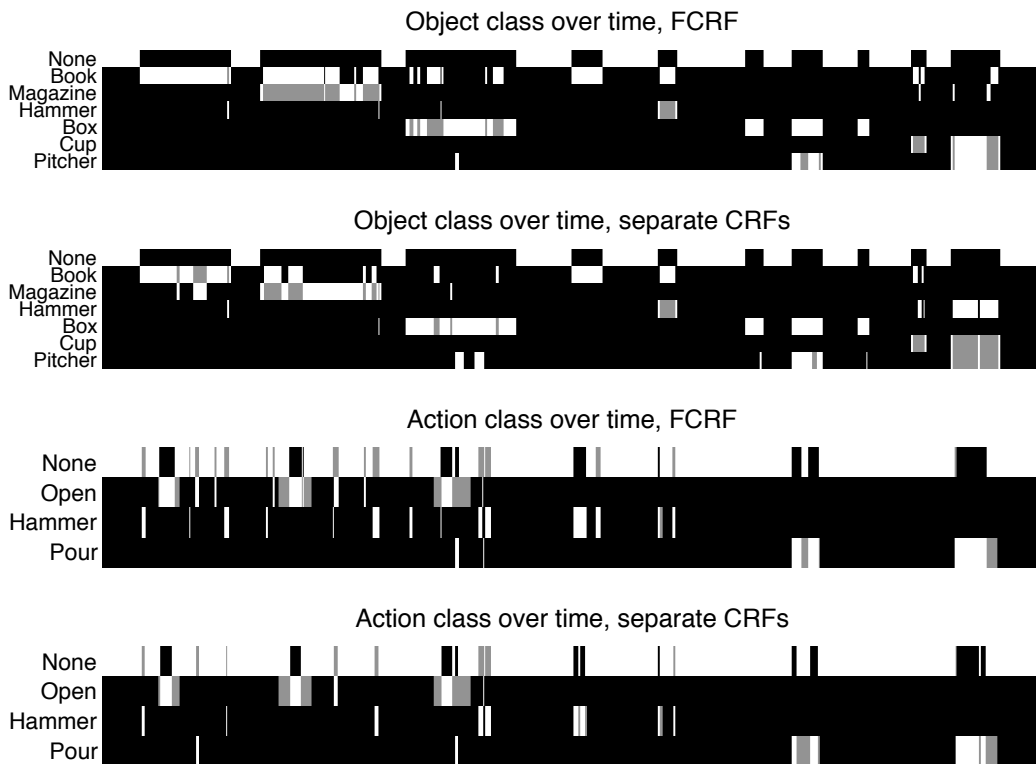


Figure 8: Object-action classification over time. The depicted sequence contains 7 object-action combinations: open-book, open-magazine, open-box, hammer-with-book, hammer-with-hammer, pour-from-pitcher-into-box, pour-from-pitcher-into-cup. Time on x axis, Classification on y axis. White block = (F)CRF classification during this time period. Grey block = classification ground truth during this time period.

pitcher, box+pitcher, or cup+pitcher) and action (none, open, hammer, or pour) ground truth. Only the frames in which a hand is detected were used for training.

The evaluation set consisted of a sequence where a fourth individual performed the same object-action combinations, using the same object instance. Only the frames in which a hand is detected were used for evaluation. (The frames with no hand were automatically labeled as object none, action none.)

From each frame of each of the four sequences, object features were extracted as described in Section 3.1. The 6 background/object classifiers were trained with images of a fifth person handling the object instances in the same way as in the training and evaluation sequences. Action features were

extracted as described in Section 3.2. The three background/action classifiers were trained on hand poses and velocities from the three training sequences.

An FCRF was trained with sequences of $T = 3$ consecutive frames of object and action features from the training data, in all 1471 training examples. This FCRF can be expected to learn

1. allowed object-action combinations
2. allowed temporal action transitions (in this dataset, only **none**-action and action-**none**)
3. typical errors in the per-frame object feature extraction
4. typical errors in the per-frame action feature extraction

To provide a baseline, two individual CRFs were trained with sequences of $T = 3$ consecutive frames of object features and action features, respectively. The object CRF can be expected to learn aspect 3) above, and the action CRF can be expected to learn aspects 2) and 4); none of them capture aspect 1).

The object-action FCRF and the two individual action and object CRFs were used to classify the evaluation sequence. The classification result is shown in Figure 8.

First, it can be noted that many frames of the sequence contains an object but action **none**; in other words, the human is doing something else with the object than opening, hammering or pouring. In these frames, the object classification in the FCRF (Figure 8, row 1) is not supported by more information than the classification in the separate object CRF (Figure 8, row 2), since all combinations of objects and action **none** are present in the training data. In the remainder of the analysis, we therefore focus on the frames where there is an **open**, **hammer** or **pour** action taking place (grey blocks in the Open, Hammer, or Pour lines in Figure 8, rows 3,4).

From an application perspective we are not primarily interested in if an the starting and ending frame of an object-action combination are correctly detected; the main focus is instead on whether the action-object combination is detected and correctly classified at all. From this perspective, both the FCRF and the individual CRFs detected all 7 object-action combinations, i.e., classified some frames of each object-action combination as something else than object **none**, action **none**.

The classification of the detected object-action combination is here defined as the majority vote among the classifications in the frames of the

detection. The FCRF (Figure 8, rows 1,3) detected the 7 following object-action combinations: **open-book** (correct), **open-book** (incorrect but allowed), **open-box** (correct), **hammer-with-book** (correct), **hammer-with-hammer** (correct), **pour-from-pitcher-into-box** (correct), **pour-from-pitcher-into-cup** (correct). This concurs with the findings in the experiments with the object feature extraction above: Books are often detected as magazines and vice versa (see also Figure 2a,b), and they both afford opening, which means that the contextual action information could not guide the object classification in the second object-action combination. The inclination to classify the magazine as **book** in Figure 8, rows 1,2 could be due to coincidences in the training and evaluation data – there were more images of books than magazines in the training data with the same orientation as the magazine in the evaluation data.

The two individual CRFs (Figure 8, rows 2,4) detected the 7 following object-action combinations: **open-book** (correct), **open-book** (incorrect but allowed), **open-box** (correct), **hammer-with-book** (correct), **hammer-with-hammer** (correct), **pour-from-pitcher-into-box** (correct), **pour-from-hammer** (incorrect). In the last combination, the object detection was inadequate by itself, but the contextual action information in the FCRF helped in inferring the correct object classes (Figure 8, row 1). Furthermore, the actions are more accurately detected by the FCRF in the two last combinations (Figure 8, row 3), than by the individual action CRF (Figure 8, row 4). The reason is most certainly the contextual object information provided by the FCRF.

This shows that the FCRF is able to infer information about the object and action present in a frame, not immediately apparent from the present image information, from knowledge about which object-action combinations are commonly observed in other data.

The many spurious detections, particularly of **hammer** actions, would pose problems to a learning from demonstration system employing the classification method. A first measure against this is to train the FCRF and the feature extractors with much larger and more diverse datasets; more individuals, more object instances, more action instances performed by each individual.

7. Conclusions

This paper investigated object categorization according to function, i.e., learning the affordances of objects from human demonstration. More specifi-

cally, we presented a method for classifying objects grasped and manipulated by a human in context of which actions the human was involved in, and at the same time, classifying human actions in context of the object involved in the action. An FCRF was trained with short sequences of simultaneously extracted object and action features, modeling both information about objects and action detection, and the likelihood of observing different object-action combinations.

Experiments with a dataset of combinations of three actions and 6 objects showed that the FCRF captured contextual dependencies which could be used to infer information about both actions and objects not present in the image data. This improved the classification of both actions and objects.

7.1. Future Work

In the applications of interest here, primarily learning from demonstration, the requirement of fully labeled training data is a limiting factor. Our intention is therefore to develop methods for *semi-supervised training* of the FCRF, e.g., using co-training (Blum and Mitchell, 1998) with an object and an action view.

A related avenue of research is the introduction of *grammatical structures* to describe human activity (Shi et al., 2003). These structures can be considered as contextual information, guiding the classification of individual actions and objects. The structures can be learnt in a semi-supervised manner from a combination of the demonstrations themselves and the human demonstrator’s utterances during the demonstration.

A slightly more philosophical question regards the different roles of objects in actions. E.g., the action-object combination `hammer-with-hammer-on-nail` contains two objects, where `hammer` is a tool and `nail` is not. Tools are tricky when reasoning about affordances (Gibson, 1979) – when used, they can be regarded as part of the agent’s body. (A robotic agent can very well have a hammer permanently attached to its body.) Thus, the division between agent, objects and scene is not clear (Wörgötter et al., 2009). At present, our method does not differ between tools and other objects in any principal way. However, this will be addressed in future work.

Acknowledgments

This research has been supported by the EU through PACO-PLUS, FP6-2004-IST-4-27657, and by the Swedish Foundation for Strategic Research.

References

- Billard, A., Calinon, S., Dillman, R., Schaal, S., 2008. Robot programming by demonstration. In: Siciliano, B., Khatib, O. (Eds.), Handbook of Robotics. Springer-Verlag, New York, NY, USA, Ch. 59.
- Blum, A., Mitchell, T., 1998. Combining labeled and unlabeled data with co-training. In: Conference on Computational Learning Theory.
- Bohg, J., Kragić, D., 2009. Grasping familiar objects using shape context. In: International Conference on Advanced Robotics.
- Carbonetto, P., de Freitas, N., Barnard, K., 2004. A statistical model for general contextual object recognition. In: European Conference on Computer Vision. Vol. 1. pp. 350–362.
- Chang, C.-C., Lin, C.-J., 2001. LIBSVM: a library for support vector machines. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
- Cristianini, N., Shawe-Taylor, J., 2000. An Introduction to Support Vector Machines. Cambridge University Press, Cambridge, UK.
- Dalal, N., Triggs, B., 2005. Histograms of oriented gradients for human detection. In: IEEE Conference on Computer Vision and Pattern Recognition. Vol. 2. pp. 886–893.
- Fei-Fei, L., Fergus, R., Torralba, A., 2009. Recognizing and learning object categories: Short course at ICCV. Slides available at <http://people.csail.mit.edu/torralba/shortCourseRLOC>.
- Filipovych, R., Ribeiro, E., 2008. Recognizing primitive interactions by exploring actor-object states. In: IEEE Conference on Computer Vision and Pattern Recognition.
- Fitzpatrick, P., Metta, G., Natale, L., Rao, S., Sandini, G., 2003. Learning about objects through action – initial steps towards artificial cognition. In: IEEE International Conference on Robotics and Automation.
- Freeman, W. T., Roth, M., 1995. Orientational histograms for hand gesture recognition. In: IEEE International Conference on Automatic Face and Gesture Recognition.

- Gibson, J. J., 1979. *The Ecological Approach to Visual Perception*. Lawrence Erlbaum Associates, Hillsdale, NJ, USA.
- Gupta, A., Davis, L. S., 2008. Beyond nouns: Exploiting prepositions and comparative adjectives for learning visual classifiers. In: *European Conference on Computer Vision*.
- Gupta, A., Kembhavi, A., Davis, L. S., to appear. Observing human-object interactions: Using spatial and functional compatibility for recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- Ikeuchi, K., Hebert, M., 1992. Task oriented vision. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. pp. 2187–2194.
- Kjellström, H., Romero, J., Martínez, D., Kragić, D., 2008. Simultaneous visual recognition of manipulation actions and manipulated objects. In: *European Conference on Computer Vision*. Vol. 2. pp. 336–349.
- Lafferty, J., McCallum, A., Pereira, F., 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: *International Conference on Machine Learning*.
- LeCun, Y., Huang, F. J., Bottou, L., 2004. Learning methods for generic object recognition with invariance to pose and lighting. In: *IEEE Conference on Computer Vision and Pattern Recognition*. Vol. 2. pp. 97–104.
- Li, L.-J., Fei-Fei, L., 2007. What, where and who? Classifying events by scene and object recognition. In: *IEEE International Conference on Computer Vision*.
- Marr, D., 1982. *Vision*. Freeman, New York, NY, USA.
- Marszalek, M., Laptev, I., Schmid, C., 2009. Actions in context. In: *IEEE Conference on Computer Vision and Pattern Recognition*.
- Miura, J., Ikeuchi, K., 1995. Task-oriented generation of visual sensing strategies. In: *IEEE International Conference on Computer Vision*.
- Moeslund, T. B., Hilton, A., Krüger, V., 2006. A survey of advances in computer vision-based human motion capture and analysis. *Computer Vision and Image Understanding* 104 (2-3), 90–126.

- Montesano, L., Lopes, M., Bernardino, A., Santos-Victor, J., 2008. Learning object affordances: From sensory motor coordination to imitation. *IEEE Transactions on Robotics* 24 (1), 15–26.
- Moore, D. J., Essa, I. A., Hayes, M. H., 1999. Exploiting human actions and object context for recognition tasks. In: *IEEE International Conference on Computer Vision*.
- Murphy, K., 2004. A CRF implementation for general graphs. Software available at <http://people.cs.ubc.ca/~murphyk/Software/CRF/crf.html>.
- Murphy, K., Torralba, A., Freeman, W. T., 2003. Using the forest to see the trees: A graphical model relating features, objects, and scenes. In: *Neural Information Processing Systems*.
- Peursum, P., West, G., Venkatesh, S., 2005. Combining image regions and human activity for indirect object recognition in indoor wide-angle views. In: *IEEE International Conference on Computer Vision*. Vol. 1. pp. 82–89.
- Quattoni, A., Collins, M., Darrell, T., 2007. Learning visual representations using images with captions. In: *IEEE Conference on Computer Vision and Pattern Recognition*.
- Rabiner, L. R., 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE* 77 (2), 257–286.
- Rivlin, E., Dickinson, S. J., Rosenfeld, A., 1995. Recognition by functional parts. *Computer Vision and Image Understanding* 62 (2), 164–176.
- Romero, J., Kjellström, H., Kragić, D., 2010. Hands in action: Real-time 3D reconstruction of hands in interaction with objects. In: *IEEE International Conference on Robotics and Automation*.
- Saxena, A., Driemeyer, J., Ng, A. Y., 2008. Robotic grasping of novel objects using vision. *International Journal of Robotics Research* 27 (2), 157–173.
- Schaal, S., 1999. Is imitation learning the route to humanoid robots? *Trends in Cognitive Sciences* 3, 233–242.
- Serrano Vicente, I., Kyrki, V., Kragić, J., 2007. Action recognition and understanding through motor primitives. *Advanced Robotics* 21 (13), 1473–1501.

- Shakhnarovich, G., Viola, P., Darrell, T., 2003. Fast pose estimation with parameter sensitive hashing. In: IEEE International Conference on Computer Vision. Vol. 2. pp. 750–757.
- Shi, Y., Bobick, A., Essa, I., 2003. Learning temporal sequence model from partially labeled data. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 1631–1638.
- Sminchisescu, C., Kanujia, A., Metaxas, D., 2006. Conditional models for contextual human motion recognition. *Computer Vision and Image Understanding* 104 (2-3), 210–220.
- Stark, L., Bowyer, K., 1996. *Generic Object Recognition using Form and Function*. World Sci. Ser. Machine Perception and Artificial Intelligence; Vol. 10.
- Stark, M., Lies, P., Zillich, M., Wyatt, J., Schiele, B., 2008. Functional object class detection based on learned affordance cues. In: International Conference on Computer Vision Systems. pp. 435–444.
- Sutton, C., Rohanimanesh, K., McCallum, A., 2004. Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data. In: International Conference on Machine Learning.
- Torralba, A., 2003. Contextual priming for object detection. *International Journal of Computer Vision* 53 (2), 169–191.
- Torralba, A., Murphy, K., Freeman, W. T., 2004. Contextual models for object detection using boosted random fields. In: *Neural Information Processing Systems*.
- Triesch, J., Ballard, D. M., Hayhoe, M. M., Sullivan, B. T., 2003. What you see is what you need. *Journal of Vision* (3), 86–94.
- Veloso, M., von Hundelshausen, F., Rybski, P. E., 2005. Learning visual object definitions by observing human activities. In: IEEE-RAS International Conference on Humanoid Robots.
- Wörgötter, F., Agostini, A., Krüger, N., Shylo, N., Porr, B., 2009. Cognitive agents – a procedural perspective relying on the predictability of object-action-complexes (OACs). *Robotics and Autonomous Systems* 57 (4), 420–432.

Wu, J., Osuntogun, A., Choudhury, T., Philipose, M., Rehg, J. M., 2007. A scalable approach to activity recognition based on object use. In: IEEE International Conference on Computer Vision.

Zöllner, R., Pardowitz, M., Knoop, S., Dillman, R., 2005. Towards cognitive robots: Building hierarchical task representations of manipulations from human demonstration. In: IEEE International Conference on Robotics and Automation. pp. 1535–1540.

Learning Actions from Observations

Volker Krueger, Dennis Herzog, Sanmohan, Ales Ude, and Danica Kragic

Abstract—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 paper we present a complete concept of learning action primitives and for using them to recognize and synthesize actions. The main novelty in our work is the detection of primitives in a unified framework that takes into account objects and actions being applied to them. In many human and robot tasks actions and objects are intertwined; we can interpret actions from their effect on the involved object. While human movements can look vastly different even under minor changes in location, orientation and scale, the use of the object can provide a strong invariant for the detection of motion primitives. As the first major contribution we propose an unsupervised learning approach for action primitives that makes use of the human movements as well as the object state changes. We group actions according to the changes they make to the object state space. This allows us to define action primitives as sets of movements where the movements of each primitive are connected through the object state change they induce. As a statistical representation of the primitives, HMMs, splines, etc. are often used. However, these are strictly trajectory based representations and they are not able to model the relationship between the trajectories and their effects. Thus, as a second major contribution, we propose to use parametric hidden Markov models (PHMMs) for representing the discovered action primitives. PHMMs allow to represent movement trajectories as a function of their desired effect on the object, and we will discuss (a) how these PHMMs can be trained in an unsupervised manner, (b) how they can be used for synthesizing movements to achieve a desired effect and (c) how they can be used to recognize an action primitive and the effect from an observed acting human.

I. INTRODUCTION

The need and motivation for imitation learning have been widely documented in the area of robotics during the last decade, [1], [2], [3], [4]. 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 [5], [6] that human actions are composed of action primitives similarly 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,

This work was partially supported by EU through grant PACO-PLUS, FP6-2004-IST-4-27657.

V. Krueger, D. Herzog and Sanmohan are with the CVMi Lab, Copenhagen Inst. of Technology, Aalborg University, Denmark. {vok, deh}@cvmi.aau.dk

A. Ude is with the Jozef Stefan Institute, Department of Automatics, Biocybernetics, and Robotics, Jamova 39, 1000 Ljubljana, Slovenia ales.ude@ijs.si

D. Kragic is with the School of Computer Science and Communication, Royal Institute of Technology (KTH), Stockholm, Sweden danik@csc.kth.se

the use of action grammars based on action primitives is a plausible representation so that in this work we follow the idea of looking for action primitives out of which all actions can be described and to look for a grammatical description, how the primitives are to be combined. It is, however, less obvious how a) these action primitives and the associated grammars can be extracted automatically from visual observations without any input in addition to the visual observations [7], b) how they should be represented in terms of data structures and c) 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. In a more recent work, the problem of on-line, continuous learning has been studied [8] where segmentation and classification are performed in an unsupervised manner. In this paper 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 the main-stream 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 for representing these parametrically. Hierarchical representation is necessary when attempting to integrate low-level control and high level action planning aspects. An additional contribution of our work is the use of parametric hidden Markov models (PHMMs) for clustering action of primitives. While many different body movements are able to induce the same changes in the object state, PHMMs offer a unified framework that allows to model the movement trajectories in terms of their effect on the object.

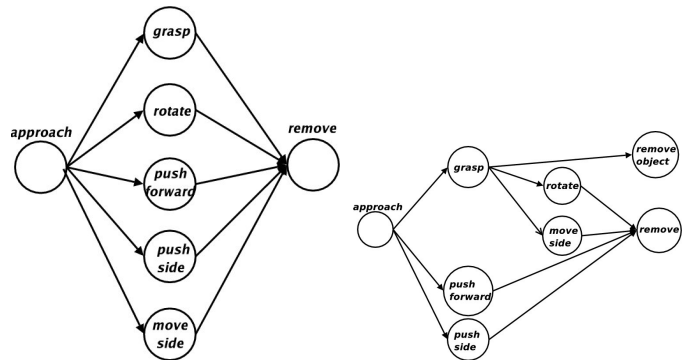


Fig. 1: In the left image, one can see a set of demonstrated actions. In the right image, one can see the *intuitive* and *hand-selected* action primitives with the corresponding grammar [9].

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 a) identify a set of action primitives and b) identify the corresponding grammar from the observations. We relate the expected outcome of the system to the graphical model shown in Fig. 1: In the left image, the actions on objects may be described with a general *approach-act-remove* cycle. In the right image, 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 the set of demonstrations. These are then further modeled by an PHMM-based representation and used for generating actions on the robot in order to achieve a desired effect as well as to allow the robot to recognize the primitives and their effects from human performances.

This paper is organized as follows: In Sec. II we give an overview of previous research and the motivation for our work. In Sec. III we discuss our approach for learning action primitives and the corresponding grammar. In Sec. IV we present our PHMMs for modeling the action primitives for the purpose of synthesis (Sec. V) and action recognition (Sec. VI-A). We will conclude the paper with final comments in Sec. VII.

II. BACKGROUND AND MOTIVATION

The derivation of action primitives is not trivial; ideally, the demonstrations are repeated, possibly even by different teachers and in different ways in order to assure good statistics. When actions are performed on objects, the objects and/or the human teacher may not necessarily be in the same location so that the visual appearance of these performances are likely to vastly differ from each other. One obvious way to look for primitives in demonstrations is to search for statistical co-occurrences, i.e., compare trajectories and identify corresponding and re-occurring parts [10]. However, since the trajectories can be vastly different from each other if the human agent or the involved objects are at different locations, such an approach will generate a lot of primitives [10]. Thus, an important challenge is to identify the primitives independent of the variability in appearance.

In order to solve this problem, we would like to argue that for object manipulation actions, one should consider both, the human movements *and* the objects to which these actions are applied. Indeed, actions and objects seem to be intertwined and we observe that a human/humanoid action can be seen from two different perspectives: a) we look at an action as being defined by movements of body parts of the humanoid agent. This is what we need for 3D body tracking and movement synthesis; b) we can also look at actions as being defined by the *effect* the human movement has on an object, e.g., *push object*, *rotate object*, etc. 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 on the human actions, one from the movement perspective and one from the object perspective. Indeed, one is tempted to identify the effect of a movement to be the “semantic” of that corresponding movement.

Therefore, instead of analyzing the *movement space*, as done in [9], [10], we suggest to approach the detection of action primitives instead 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.

A major disadvantage of using the *effects* of human movements for identifying the primitives instead of the trajectories in *movement space* is that state-of-the-art approaches such as hidden Markov models (HMMs) cannot easily be used anymore for recognizing and synthesizing the movements because they model trajectories in the *movement space* but discard the *effects* as noise.

Therefore, we require a representation that is *effect* dependent. The parametric hidden Markov model is such a representation because it takes the *effect* of a movement as a parameter and generates a new trajectory in *movement space* that would lead to the desired *effect*.

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

- the definition of duality between the movement space and the object state space;
- unsupervised learning procedure for detecting action primitives;
- learning of the underlying action grammar;
- using parametric hidden Markov models (PHMMs) for modeling trajectories in movement space based on their effect in the object state space;
- non-supervised 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 and where a robot should generate the necessary movement;
- using PHMMs for action recognition, where we require to identify the effect and the action primitive based on the observed movement.

Our concept is illustrated in Fig. 2 and it consists of the following steps:

- 1) Non-supervised learning of action primitives: identify the selection of necessary action primitives. This will be done by analyzing the *effects* of the actions on the objects, rather than the 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 action trajectories of the *reach for object* primitive.
- 2) Build a parametric hidden Markov model (PHMM) for each of these primitive classes based on the movement trajectories and their effects.
- 3) 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.

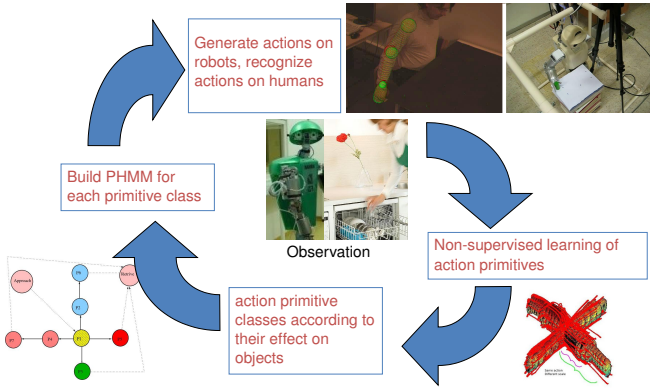


Fig. 2: The figure shows the different steps involved in learning actions.

III. UNSUPERVISED LEARNING OF ACTION PRIMITIVE CLASSES ACCORDING TO THE *Effect* ON OBJECTS

In this section we discuss our approach to unsupervised learning of action primitives based on the *effect* of the actions on objects.

In the first two steps in Fig. 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 trajectories H_t^i in movement space and object trajectories O_t^i in object state space. While the trajectories H_t^i in movement space are given by the hand locations, O_t^i is given by the object locations and orientations. We propose to analyze O_t^i to detect joint trajectories across the different action effects which gives rise to a set of primitives. This is an important difference from, e.g., [11], where human joint data is used to identify action primitives. Having found primitives in the object state space, i. e., a segmentation for each of the O_t^i into these primitives, we become able to segment H_t^i in the same way. If done for all training movements, we obtain sets of human trajectories where each set corresponds to a specific primitive (specific *effect*) in the object state space. In other words, each set is an equivalence class of human movements modulo the effect these movements have on the object state space.

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 the object state space is quantized in terms of the object locations, then, e.g., two *push* movements are the same iff the initial and the final object locations of the two movements are the same. We have investigated in our work two different quantizations: the first one is based on the change of the object location in terms of Euclidean coordinates (Euclidean quantization), the second one is based on the change of object location and orientation in terms of polar coordinates (polar quantization). While the Euclidean quantization is direction selective, the latter is direction invariant.

In the following subsections, we will give a brief explanation of all necessary steps. A more detailed description can

be found in [12]. For better visualization in this section, we chose the Euclidean quantization.

After this section, the learning approach will provide us with a) sets of movements that all have the same effect on the object with respect to the chosen quantization of the object state space and b) how precisely the object states were changed by each of the movements. As we will discuss in Sec. IV-A both pieces of information will be required for unsupervised training a PHMM for each of the detected equivalence class.

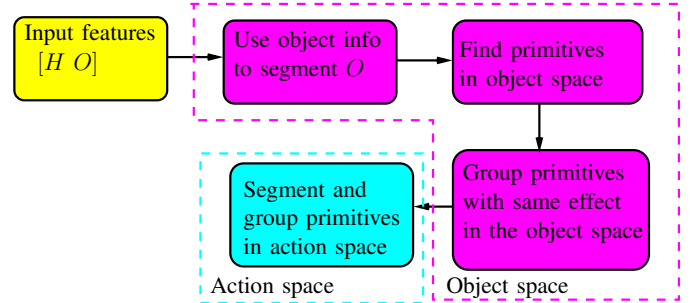


Fig. 3: *Overview of our approach.* The input H and O denote the action and object features. The object features are first analyzed and segmented. This is then used to extract the primitives in the action space. Magenta boxes correspond to analysis in the object state space, while the cyan box represents analysis in the action space.

A. Modeling Object-Action Interactions as HMMs

In the first step we analyze the trajectory O_t^i in order to identify the joint effect trajectories in the object state space. To do this, we represent the set of trajectories as a left-to-right hidden Markov model (HMM) [13] (see also Sec. IV-A2). We build the HMMs recursively: each trajectory O_t^i is modeled as an ordered sequence of 2D Gaussians mixtures where the mean is given by the location on the trajectory, the major axis of each Gaussian points in the direction along the trajectory while the length of the minor axis' is given by the expected noise in the trajectories. A transition matrix is used to capture the temporal order of the Gaussians along the trajectories. If for two sequences two Gaussians strongly overlap, they are fused into a single Gaussian. This keeps the final number of Gaussians small. After the fusion of two Gaussians, the transition matrix is adapted accordingly. At the end of this process, we obtain a single hidden Markov model that models the entire training set and where the HMM states are given by the Gaussians.

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 summarized in Fig. 4 for a pushing action example: While a human pushes an object on a table, the object coordinates change. Fig. 4 shows pushes into 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

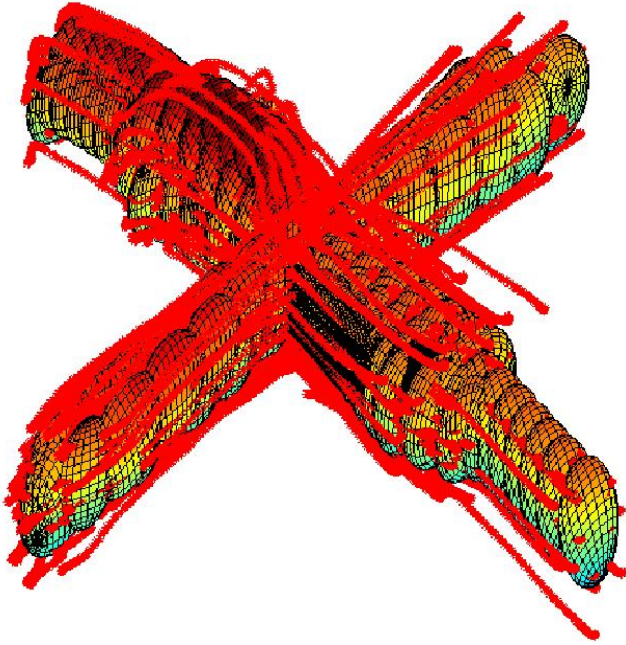


Fig. 4: *Data covered with Gaussians*. Ellipsoids show the contours of Gaussians used to cover the data. Lengths of the trajectories indicate how much distance the object was moved.

in the example training set, the variability of short movements was larger.

B. Grouping of HMM States into Primitives in Object Space

Once we have sampled the continuous trajectories into discrete state sequences, the next step aims at detecting the action primitives. This will be done by using a longest common substring (LCS) approach. Two steps are required: a) The left-to-right HMMs are expected to have self-transitions which means that the discrete state sequences contain not only state changes but also state repetitions. To be able to apply the LCS approach, we discard in the first step the state repetitions such that the discrete state sequence only contain the state changes. b) we now can consider the learning of the final primitives from the discrete state sequences as a Longest Common Substring (LCS) problem [10] which we solve using a dynamic programming approach [14]. This results in a small set of primitives in the object state space and each original continuous trajectory O_t^i can now be represented as a sequence of primitives $p_1^i, p_2^i \dots p_N^i$.

C. Segmentation and Grouping in Movement Space

In the final step we start from a sequence $p_1^i, p_2^i \dots p_N^i$ of primitives for the object state trajectory O_t^i and we now propagate the same segmentation to the trajectory H_t^i in the movement space. While doing this for all trajectory pairs $[H_t^i, O_t^i]$, all pieces of the 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 the object state space.

D. Experimental Results

We have tested our approach for learning action primitives on a re-recorded version of the dataset that was previously used in the work by Vicente *et al.* [9]. In their work, a dataset of different human arm actions doing manipulative tasks on objects in a table-top scenario was recorded and ground truth was generated. Fig. 5, right, shows the experimental setup for the manipulative arm actions on objects on a table top. In Fig. 1, left, one can see the different actions that



Fig. 5: *Left: Our Experimental Setup*. The markers are attached to both, person and object. The object can be moved from any position to any other position on the table. *Right: Experimental Setup in [9]*. The table is marked with locations where the object can be moved around. Object positions are not recorded.

were recorded. Action primitives were hand-selected in an intuitive sense as ground truth (Fig. 1, right). The aim of our experiments is to learn these intuitive action primitives automatically. Since the original dataset of [9] was lacking data about the object state, we re-recorded the dataset using our Vicon motion capture system with markers placed on the human as well as on the object. In addition, the new dataset provides a larger variability in object location as well as four different directions for the *push* and *move* movements (up, down, left, right). Fig. 5, left, shows the setup for the new recording that includes the object-action interaction. As object states we use the object locations and the object orientation. For the Euclidean quantization of the object state space, the finally recovered action primitives are shown in Fig. 6. As it can be seen, we recover object translations (move and push actions) into the four different directions available in the training data, the rotate, the approach and the remove primitive. It is interesting to note that the move and push movements were identified as *one* movement, however. 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. It is important to note in this context that both, the *push* and the *move* movements required the object to be grasped. The difference between *move* and *push* in our dataset is that the *push* movement moves the object on the table while the *move* action lifts the object from the table in order to place it at a

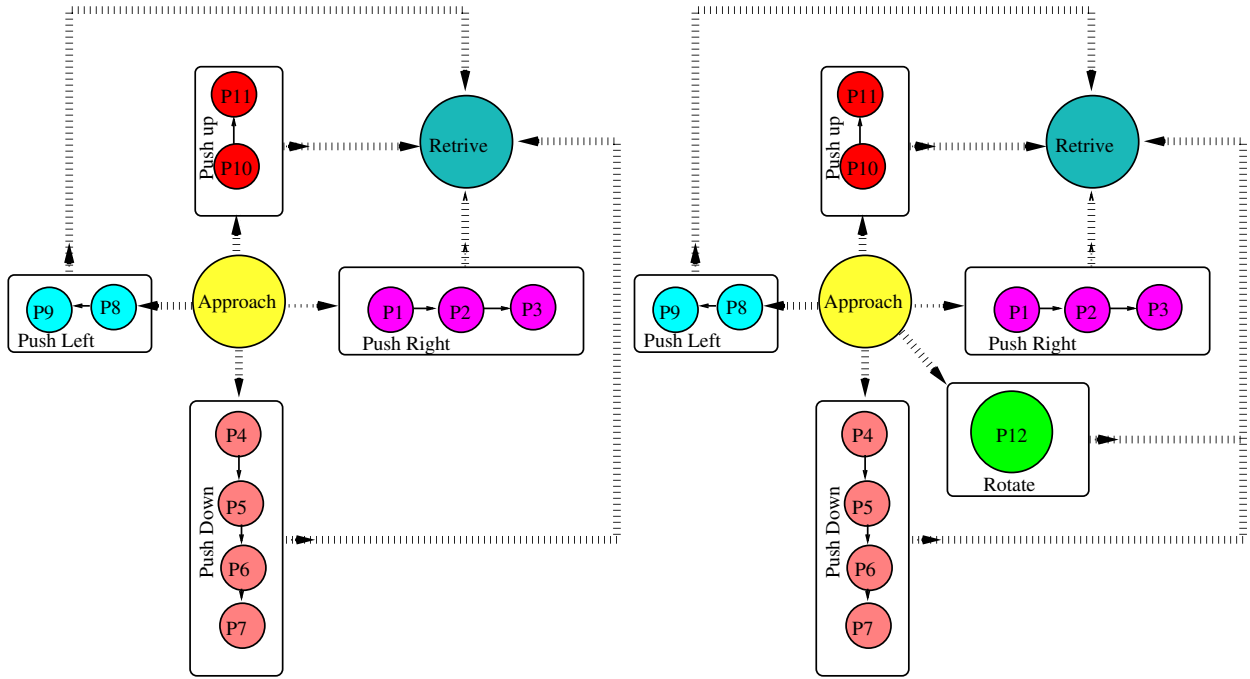


Fig. 6: *Results of Primitive Extraction in Movement Space.* The figures, left and right, show results for slightly different data. Each of the actions (recoded in sequence) starts by approaching the object, then it performs a manipulative primitive and ends by retrieving the hand. Each of the squares represents a primitive in the object state space and the corresponding primitive in the movement space. *Left:* Resulting primitive structure when rotation action was not included. *Right:* Result when rotation is included.

new location. Since both movements require the object to be grasped, the *approach* movements are the same.

When using polar quantization of the object state space, the discovered action primitives become direction invariant, i.e., the direction variant move primitives in the Euclidean quantization are fused into a single, direction-invariant move primitive. In the remainder of this paper, we will use the direction invariant action primitives for learning a PHMM-based representation for each of the primitives.

IV. 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 a) a set of equivalence classes with movements that are equivalent in terms of their effect on the object state space and b) the precise effect of each of these movements on the object state. For example, using the polar quantization, all *push* and *move* movements were clustered into the same class. Furthermore for each movement m we stored with it its precise effect, e.g., the initial location x_0 of the object and the final location x_1 .

In this section we want to introduce parametric hidden Markov models [15] as a compact representation that allows to model each action primitive (equivalence class) in a way that it 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 a human movement.

Hidden Markov models [13], [11] are a common tool for statistically representing movements as trajectories. However,

they do not have the possibility to model these trajectories in terms of an effect they should cause. On the contrary, variations in the movement trajectories that are necessary to cause a certain effect are modeled as noise instead. For example while an HMM can be used to model *pointing* movements (“Look at this object”), the precise pointing direction will be considered as noise. If one wanted to use a HMM to recognize the pointing direction, one would have to use several HMMs, one HMM for each specific direction. This means that the meaning of the pointing movement, i.e., where is being pointing to, is completely lost when using an HMM. Parametric hidden Markov models, on the other hand, are able to identify the pointing movement as well as the direction of pointing.

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

A. The Parametric Hidden Markov Model

Parametric hidden Markov models (PHMMs) [15] are an extension of the hidden Markov models (HMMs) [13], [16] through latent parameters $\phi = (\phi_1, \dots, \phi_N)$, which model a systematic variation within each action class. In the following we shortly review the main principles of HMMs and then extend the HMMs into PHMMs.

1) *The Hidden Markov Model:* A hidden Markov model is a generative model. It is a finite state machine extended in a probabilistic manner. For an HMM $\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$, the vector $\boldsymbol{\pi} = (\pi_i)$ and the transition matrix $\mathbf{A} = (a_{ij})$ 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 $b_i(\mathbf{x})$, which are in our context multivariate Gaussian densities $b_i(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$. The HMM parameters can be estimated through the Baum-Welch algorithm [13] for a set of training sequences.

2) *Sequence Generation and Left-Right HMMs*: An output sequence $\mathbf{X} = \mathbf{x}_1 \dots \mathbf{x}_T$ can be drawn from the model by generating step-by-step a state sequence $\mathbf{Q} = q_1 \dots q_T$ with respect to the initial probabilities π_i and the transition probabilities a_{ij} and drawing for each state q_t the output \mathbf{x}_t from the corresponding observation distributions $b_i(\mathbf{x})$. Generally, there is no unique correspondence between an output sequence \mathbf{X} and a state sequence as different hidden state sequences can generate the same output sequence \mathbf{X} . This seems to be a rather poor approach in order to generate a good prototype movements from the model. To overcome this problem, we use a left-right model [13] as shown in Fig. 7. A good prototype can then be generated by taking the sequence means $\boldsymbol{\mu}_1 \dots \boldsymbol{\mu}_T$ and interpolating between the means with respect to expected time durations encoded in the state transitions.

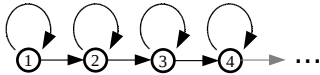


Fig. 7: Transition Structure of a particular Left-Right HMM.

3) *The Parametric Extension to HMMs*: A Parametric hidden Markov model (PHMM) [15] λ^ϕ allows to take into account a systematic variation ϕ in the input data as variations of the means of the observation distributions $b_i^\phi(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_i^\phi, \boldsymbol{\Sigma}_i)$. This means that for PHMMs the means $\boldsymbol{\mu}_i^\phi = \mathbf{f}_i(\phi)$ of the observation pdfs b_i^ϕ are functions of the parameter ϕ and that the functions $\mathbf{f}_i(\phi)$ are approximated for each state i separately in the training process. For example, for an approach action, the parameter ϕ is given by the location of the object to be grasped. In order to model the approach action to location ϕ , all observation pdfs b_i^ϕ 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 ϕ contains the initial location A as well as the final location B .

In [15] a linear as well as a more general nonlinear model are used to model $\mathbf{f}_i(\phi)$. In the linear case, each function $\mathbf{f}_i(\phi)$ is of the form $\boldsymbol{\mu}_i = \bar{\boldsymbol{\mu}}_i + \mathbf{W}_i \phi$, where the matrices \mathbf{W}_i describe the linear variation ϕ . In the more general nonlinear case, a neural network, which is trained to approximate a more general nonlinear dependency on ϕ , is used for each state i . For both models, the training procedures are generally supervised as the parameterization ϕ for each training sequence needs to be provided. In our case, these parameters ϕ are automatically provided through our unsupervised primitive learning approach from Sec. III: Recall that the result of the primitive learning approach was a) sets of movements that all have the same effect on the object with respect to the chosen quantization of the object state space and b) how precisely the object states were changed by each of the movements. That

means while the movements in each equivalence class are used for training the PHMM, the initial and the final object state for each of the movement are available and provide the necessary parametric data for the 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 the final object position. In order to be able 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 the object location: while for example the move action uses two object locations, we gain an additional location parameter at the beginning of the approach movements and an additional location parameter at the end of the remove movement by considering the hand location. This way, all primitives become bi-parametric which allows us to use a single learning approach for all primitives.

In this paper we used a linear model to model the parametric variability between the movements. For learning the PHMM parameters we used an extended version of the Baum-Welch approach.

4) *Synthesis with PHMMs*: The procedure for generating a particular sequence with parameter ϕ is closely related to the method explained in Sec. IV-A2. The difference here is that one generates a specific movement for a parameterization ϕ : given ϕ , one first computes the means $\boldsymbol{\mu}_1^\phi \dots \boldsymbol{\mu}_T^\phi$ for the observation densities b_i^ϕ . This is done by evaluating the functions $\mathbf{f}_i(\phi)$ that were learned in the training process (Sec. IV-A3). Then, as all the observation densities b_i are specified, a good prototype can be synthesized as described in Sec. IV-A2. Once the parameter ϕ is defined the means are fixed.

5) *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 an HMM λ_k . The maximum likelihood approach can then be applied to identify \mathbf{X} as that class k which maximizes the likelihood:

$$k^{\text{ML}} = \arg \max_i P(\mathbf{X}|\lambda_i).$$

The likelihood can be efficiently computed using the forward-backward procedure [13]. In the case of parametric HMMs $\lambda_k^{\phi_k}$ the recognition becomes a two step procedure. First one estimates for each model $\lambda_k^{\phi_k}$ the parameterization ϕ_k that explains the movement in the maximum likelihood sense best:

$$\phi_k^{\text{ML}} = \arg \max_{\phi_i} P(\mathbf{X}|\lambda_i^{\phi_i}) \quad (\text{for each } k \in \mathcal{K}).$$

This can be done using EM, as in [15], or with gradient descent, as in [17].

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.

$$k^{\text{ML}} = \arg \max_i P(\mathbf{X}|\lambda_i^{\phi_i^{\text{ML}}}).$$

V. SYNTHESIZING ACTIONS WITH PARAMETRIC HIDDEN MARKOV MODELS

In the previous sections, we have discussed how action primitives can be detected and how parametric hidden Markov models can be learned in order to represent these action primitives.

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

A. Precision of synthesized actions from PHMMs

As discussed above, PHMMs are able to synthesize movements that are meant to generate a specific effect on the scene. For example in order to move an object from location A to location B , the PHMM λ_{move}^ϕ that was trained for the *move* action primitive is parameterized with the parameter $\phi = (A, B)$. As discussed above in Sec. IV-A4 the parameter ϕ effects the observation densities b_i^ϕ of the PHMM λ_{move}^ϕ in such a way that the first observation pdf b_1^ϕ assures that the hand is located at location A , the last observation pdf b_{last}^ϕ positions the hand at the final location B , 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 by definition start at location A and end at location B as these are hard-constrained by the parameters.

The locations of the observation pdfs within the movement trajectory and the functions $\mu_i = \bar{\mu}_i + \mathbf{W}_i \phi$ 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 can be 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 a) four repetitions of *approach* movements to each of the four corners of the table (2×2 grid) and b) 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 $\lambda_{approach}^\phi$ with 40 states.

For testing, we synthesized movements for evenly distributed 5×7 locations on the table. For each of these locations, 4 repetitions of the true human movements were available in our movement dataset. 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 $f(t)$ and the averaged trajectory $\bar{f}(t)$ of the four available human performances. To be precise, each human movement in our movement dataset is specified by 6 3D trajectories for the shoulder, the elbow of the right arm, the wrist the index finger, its knuckle, and the thumb of the right hand. Thus, the movements $f(t)$ and $\bar{f}(t)$ are 6 3D trajectories $f(t) = (f_i(t))_{i=1}^6$, where the $f_i(t)$, $i = 1, 2, \dots$

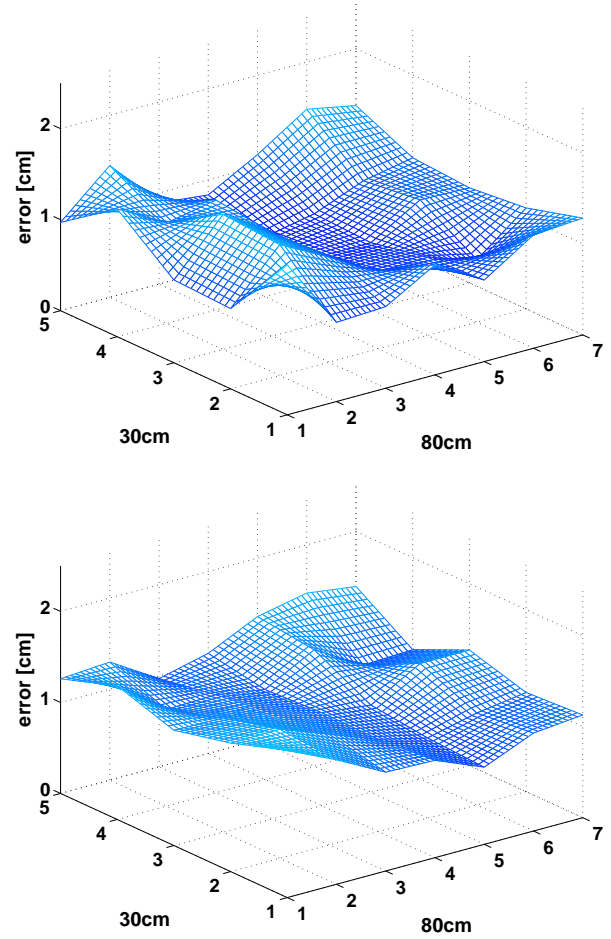


Fig. 8: The top image shows the root-mean-square error for the synthesized grasping movement, where the PHMM was trained on 2×2 training movements. The bottom image shows the route-mean-squared error for the human reference performances. The tabletop has the size 80×30 .

are shoulder, elbow, wrist, etc. The error ε of the synthesized movement $f(t)$ is calculated as the route-mean-square error between the synthesis and the averaged human movement $\bar{f}(t)$:

$$\varepsilon = \sqrt{\frac{1}{6} \sum_{i=1}^6 \int (f_i(\alpha(t)) - \bar{f}_i(\bar{\alpha}(t)))^2 dt / \int \alpha(t) dt}, \quad (1)$$

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

The results are summarized for the 2×2 training grid in Fig. 8,top. For the 3×3 training grid the results are very similar to the 2×2 grid. The synthesis errors are approximately 1.8cm. We have measured the route-mean-error of the human performances compared to the average human performance. The result is plotted in Fig. 8. By comparing the two plots in Fig. 8 one can see that the performance of the PHMM is comparable with the human performance.

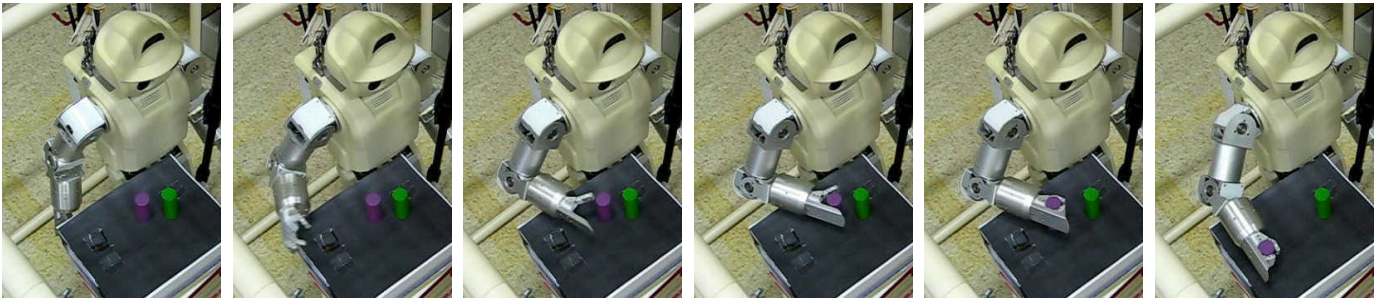


Fig. 9: *Generating movements on a robot.* The humanoid HOAP-3 grasps for an object and puts it somewhere else.

B. Concatenating action primitives into complex actions using PHMMs

If actions are defined in terms of action primitives and action grammars, one has to be able to concatenate the primitives in order to become able to synthesize complex actions, e.g., on a robot. While we are aware that a concatenation of action primitives for animation purposes is very complex [18], [19] as, e.g., smooth transitions between primitives have to be assured, 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 place these objects into specific holes in the table (see Fig. 9). The scenario was inspired by the classic children’s game where specific geometric objects fit only into specific holes in a box. In our experimental setup, the object states were given by their 2D locations on a table, and the 2D locations were identified through a camera above the table. The location of our humanoid robot relative to the camera was calibrated. In order to generate movements on the robot, we used the three action primitives: *approach*, *move*, *remove* as learned from our primitive learning approach and represented by our PHMMs. Extra care was taken for the *move* primitive. As discussed in Sec. III, our learning approach is not able to distinguish between the move and the shift movement because we do the clustering based on the *effect* of a movement, but not the movement itself. Thus, in our robot experiment, we assured that the PHMM λ_{move}^{ϕ} for the *move* movement was trained only on the true *move*-movements in the *move/push* equivalence class while the *push*-movements were deleted from that equivalence class¹.

The robot task was *a-priori* specified as an approach-move-remove movement. During the execution, a human supervisor advised the robot which object to move by pointing towards the object. The robot identified the object closest to the indicated location and estimated its position, which we denote here by A , by vision. Using this information and the training data it was able to generate the appropriate *approach* movement to location A and grasp the object. To generate the movement, the PHMM $\lambda_{approach}^{\phi}$ was used, where ϕ specifies two locations: 1) the initial location of the end-effector in its resting location and 2) the location A . The grasp is modeled as part of the *approach* movement (see Sec. III-D).

Next, the robot executed the *move* movement where the object was moved from location A to the new location B .

¹We are presently investigating the enhancement of the *effect*-based primitive selection to include also hand and grasping information.

This trajectory was executed using λ_{move}^{ϕ} where ϕ specified the locations A and B .

Finally, the robot executed the movement *remove*, where the end effector opens and the arm is removed from the table until it ends again in its resting position alongside the robot body. The trajectory is computed using λ_{remove}^{ϕ} where ϕ is specified 1) by the final object location B as the start location of the *remove* and 2) by rest position of the end-effector as its final location (same location as the initial location of the *approach* movement). Implementational details can be found in [20].

VI. 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 want 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 Sec. III we become able to recognize complex actions.

The most common approach to action recognition is to assume that a person can be tracked and that the 3D postures of the person can be estimated. Once the time sequence of 3D body postures has been estimated, the sequence is evaluated, commonly with an HMM [21] or possibly even with a PHMM, as discussed in Sec. IV-A. However, non-intrusive full-body motion capture is neither a trivial nor a solved problem [22], [21], [23]. In this section we propose a completely novel, context- and object-driven approach where we acknowledge the fact that tracking, action recognition and the scene context are intertwined, i.e., that the objects and the context constrain the actions and vice versa.

A. Recognition through Tracking in Action Space

Instead of posing the action recognition problem as a 3D human tracking problem with a subsequent action recognition step, we suggest to pose the action recognition problem as an *action recognition problem* where we acknowledge that any action is being carried out in a context, and that, e.g., manipulative actions are carried out on objects. Instead of a 3D tracking approach with a subsequent action recognition, we use our PHMMs to estimate directly which action it is and which parameterization it has. As a consequence, while general 3D human body tracking needs to solve the parameter

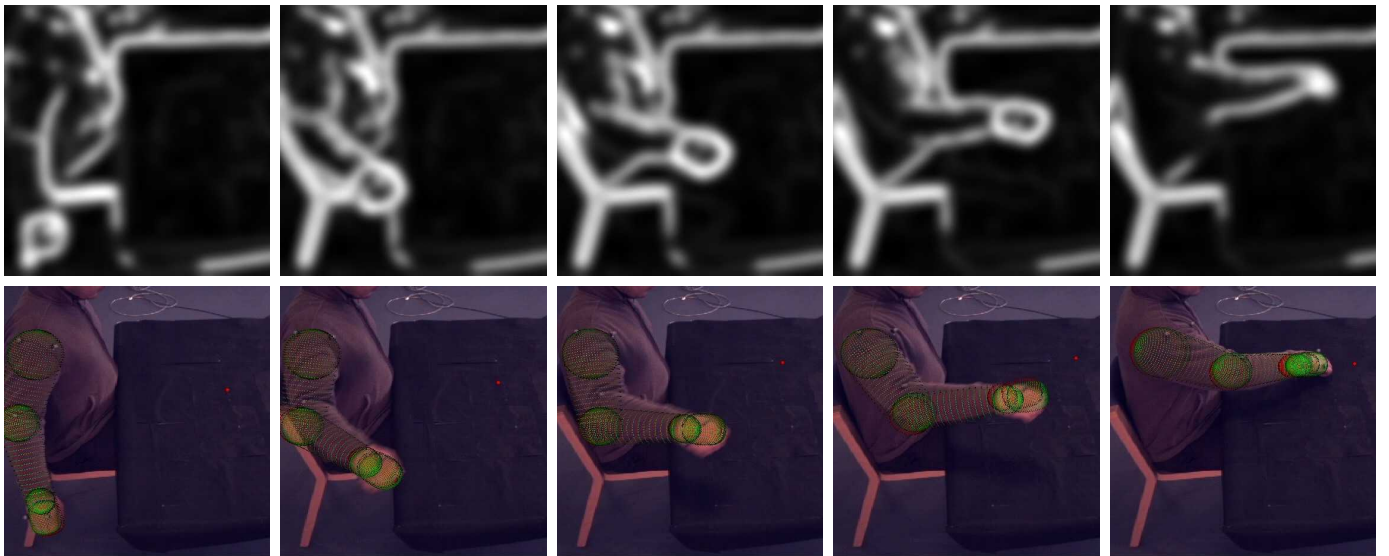


Fig. 10: *Tracking Actions in Action Space*. The images show a person reaching for a certain position on the table. The top row shows the edge image for the likelihood measurement, the second row shows the corresponding camera image with the detected arm pose superimposed. The current estimate of the position is indicated by the red point on the table.

estimation problem in a very high parameter space of human joint configurations, our approach to action recognition is concerned with a much smaller parameter space which is defined by the conditional density over the possible actions and parameters, given the context and objects in that context. Since our parametric hidden Markov models are generative models, we can generate the corresponding 3D postures from the PHMMs and deduce the 3D human pose in the scene.

We call this approach *tracking in action space* and we define the *action space* as given by the above conditional density. For example, in case of an approach action (Sec. IV-A), the action space is essentially described by only 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 more detail, let us consider the classical Bayesian propagation over time as it is often used in the context of general 3D human body tracking [21]:

$$p_t(\omega_t) \propto \int P(\mathbf{I}_t|\omega_t)P(\omega_t|\omega_{t-1})p_{t-1}(\omega_{t-1})d\omega_{t-1}, \quad (2)$$

where \mathbf{I}_t is the current visual observation, $p_t(\omega_t)$ the probability density function (pdf) for the random variable ω_t at time t , $P(\omega_t|\omega_{t-1})$ the propagation step, and $P(\mathbf{I}_t|\omega_t)$ the likelihood measurement of \mathbf{I}_t , given ω_t . Typically, the random variable ω_t specifies the body configuration of a human model in joint angles, and the propagation density is used to constrain the random variable ω_t to the most likely pose values at each time step t [24], [25]. In order to compute the likelihood $P(\mathbf{I}_t|\omega_t)$, a human body model is generated using the pose values from ω_t and then compared with the input image data \mathbf{I}_t . For evaluating the Bayesian propagation, one commonly uses a particle-based approach [22], [21], [26], [27].

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

The propagation density $P(\omega_t|\omega_{t-1})$ 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 ϕ can also be assumed to be roughly constant until a new action primitive is started.

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

B. 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 $\mathcal{A} = \{1, \dots, M\}$ of actions, where for each action $a \in \mathcal{A}$ a PHMM λ_a^ϕ was trained.

In Sec. IV-A2 and Sec. IV-A4 we have discussed how to generate a sequence from a PHMM λ_a^ϕ for action a and parameter ϕ . Hence, for a given $\omega_t = (a, \phi, \tau)$, $P(\mathbf{x}|\omega) = b_{a,\tau}^\phi(\mathbf{x})$ defines the distribution of joint angles of 3D body poses for which $b_{a,\tau}^\phi(\mathbf{x})$ generates a corresponding 3D human body model (see Fig. 11, left) which is then matched against the

input image I_t :

$$P(I_t | \omega_t) = \int_x P(I_t | x) P(x | \omega_t) dx. \quad (3)$$

Finally, the propagation density $P(\omega_t | \omega_{t-1})$ is given as follows: τ is propagated as mentioned above by the transition matrix A_a of PHMM λ_a^ϕ , and we allow ϕ to change according to a Gaussian distribution (Brownian motion[28]). The variable a which specifies the action primitive is initially drawn from a context-dependent distribution and is allowed to change according to the grammar computed in Sec. III.

It should be noted that for all action primitives and the 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 which immediately defines the first parameter for the *move* PHMM λ_{move}^ϕ and it must, according to our grammar in Sec. III coincide with the final parameter of the *approach* primitive.

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

Furthermore, by marginalizing over ϕ 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 ϕ . Fig. 12 shows the progression over time for the unknown parameters of the *approach* action. The red and green lines show the most likely 2D location parameters u and v (for $p = (u, v)$). 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 ≈ 60 frames, the correct parameters are recovered. This is about the time when the arm is fully stretched.

In the next section, we will discuss how the observation likelihood $P(I_t | \omega_t)$ is computed.

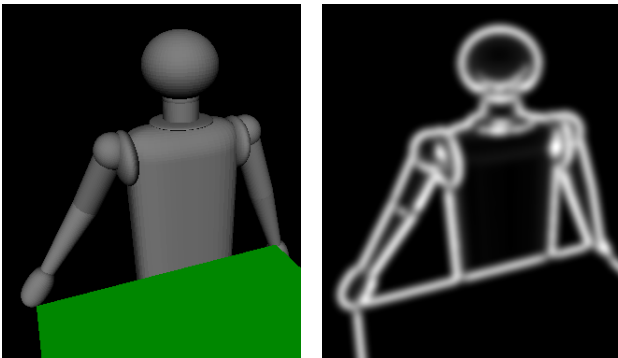


Fig. 11: *Left: Body Model.* We use an articulated human model, where the skeletal structure is fleshed out by cones, and super-quadratics. Each shoulder and elbow pair have 4 degrees of freedom. — *Right: Edge Distance Image.* The edge image (here of the model itself) is a smoothed gradient image, serving as a distance to edges image.

C. The Observation Model

We use an articulated model of the human body, see Fig. 11, left.

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 x . We defined the observation function similar to the method described in [27] on a smoothed edge image (see Fig. 11, right), 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 I , 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 $1 - G(c)$ of a contour pixel 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 normals as used in [29], but faster to evaluate.

The observation function is computed as

$$P(I | x) = \exp \left\{ -\frac{1}{2\sigma^2} \frac{1}{|\mathcal{C}|} \sum_{p \in \mathcal{C}} (1 - G(p))^2 \right\}, \quad (4)$$

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

$$P(I | x) = \exp \left\{ -\frac{1}{2\sigma^2} \sum_i \frac{1}{|\mathcal{C}_i|} \sum_{p \in \mathcal{C}_i} (1 - G_i(p))^2 \right\}, \quad (5)$$

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

D. Experiments

We evaluated our approach on monocular video data of the same arm actions as in our movement dataset [9], as described in Sec. III. The scenario (actor and table-top) in which the actions are performed can be seen in the tracked sequence, Fig. 10. As action model, we used linear PHMMs λ_a^ϕ with $\phi = (u, v)$ as trained in Sec. IV-A.

The propagation over time is performed as described in Sec. VI-B. We decrease the diffusion of the Brownian motion of u and v in dependence of the state number τ . Therefore, we draw the diffusion offset from a Gaussian where the standard deviation $\sigma = \sigma(\tau)$ is a function of τ , as also shown in Fig. 13. 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 $\phi = (u, v)$ 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 like in [29]. As described in Sec. VI-C, the observation function is based on the evaluation of the edge information in the monocular video data and the human body model for state $x_{\omega_{ti}}$.



Fig. 12: *Progression of Action Parameters.* 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 offset in cm to center of the active table-top region, in the images of Fig. 10 the variables correspond to the “horizontal” (u) and “vertical” direction (v). The dotted lines show the standard deviation for u and v .

The images in Fig. 10 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 is 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 the posture estimation, one can see in Fig. 10 that the estimation of the action parameters (corresponding to the position indicated through the small red dot) converges to the true parameters of the action when the arm approaches the table-top.

To evaluate the quality of posture estimation, we recorded the joint positions in parallel with a marker-based motion capture system. The route-mean-square error of the three joints positions (shoulder, elbow, and finger) over the whole sequence as shown in Fig. 10 is 3.3cm. The component-wise averaged error is only 0.4cm. This error was within the natural human action variation as it was observable in the training data.

VII. CONCLUSIONS

We have presented a complete framework for learning action primitives and for representing them with parametric hidden Markov models for synthesizing and recognition actions. The learning algorithm was used for simple actions using object information. We are able to recover a simple primitive structure for the actions that is 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.

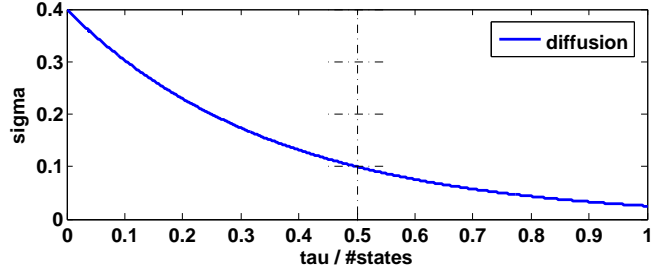


Fig. 13: *Cooling-Down of Brownian Motion.* The plot shows the deviation $\sigma(k) = 0.4 \cdot \exp\{-2 \log_e(1/4) \cdot \tau / \#states\}$ of the diffusion. We decrease the Brownian motion in dependence on the particle’s current state number k .

Several authors have represented actions in a hierarchical manner [30], [31], [32]. These works require the manual modeling of atomic movements/primitives. The contribution of our work is that we perform this segmentation automatically.

The experimental results from [33] suggests that action perception and execution of motor primitives are connected through objects. There are also further studies from experimental psychology which confirms the role of objects in action understanding [34], [35]. In this paper we have exploited object information to learn action primitives. Even though object detection and classification literature is quite large (for overview see [36]), there are not many attempts to combine it with action modeling [37], [38]. In [37] Hidden Markov models are combined with object context to classify hand actions. Image, object and action-based evidence was 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 the particular action. Our approach learns such a model for modeling actions automatically. A graphical Bayesian model was used in [38] for modeling human-object interactions. Some of the conditional probabilities of this model was calculated using trained 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 [8] 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.

Alternatives to PHMMs are for example extensions of HMMs[39] as in [40] where a metric of imitation performance is found and used for optimizing trajectories or the use of dynamic motion primitives [41].

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 [5], [6], [7], [42], [43] about human actions being composed out of motor primitives similarly 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 [44], [7], [43] to model more complex actions. Furthermore, [43] explains how a language for human actions can be generated based on grounded concepts, kinetology, morphology and syntax.

REFERENCES

- [1] C. Breazeal and B. Scassellati, "Robots that imitate humans," *Trends in cognitive sciences*, vol. 6, no. 11, pp. 481–487, 2002.
- [2] A. Billard, Y. Epars, S. Calinon, S. Schaal, and G. Cheng, "Discovering Optimal Imitation Strategies," *Robotics and Autonomous Systems*, vol. 47, pp. 69–77, 2004.
- [3] V. Krüger, D. Kragic, A. Ude, and C. Geib, "The meaning of action: A review on action recognition and mapping," *Advanced Robotics*, vol. 21, no. 13, pp. 1473–1501, 2007.
- [4] J. Kober, B. Mohler, and J. Peters, "Learning perceptual coupling for motor primitives," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008, pp. 834–839.
- [5] G. Rizzolatti, L. Fogassi, and V. Gallese, "Neurophysiological Mechanisms Underlying the Understanding and Imitation of Action," *Nature Reviews*, vol. 2, pp. 661–670, Sept. 2001.
- [6] —, "Parietal cortex: from sight to action," *Current Opinion in Neurobiology*, vol. 7, pp. 562–567, 1997.
- [7] G. Guerra-Filho and Y. Aloimonos, "A sensory-motor language for human activity understanding," *HUMANOIDS*, 2006.
- [8] D. Kulic and Y. Nakamura, "Scaffolding on-line segmentation of full body human motion parameters," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008, pp. 2860–2866.
- [9] I. S. Vicente, V. Kyrki, and D. Kragic, "Action recognition and understanding through motor primitives," *Advanced Robotics*, vol. 21, pp. 1687–1707, 2007.
- [10] Sanmohan and V. Krueger, *Primitive Based Action Representation and Recognition*, ser. Image Analysis, R. J. A.B Salberg, J.Y. Hardeberg, Ed. Springer Berlin / Heidelberg, 2009, vol. LNCS 5575.
- [11] 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. Robotics Research*, vol. 27, no. 7, pp. 761–784, 2008.
- [12] Sanmohan, V. Krueger, and D. Kragic, "Unsupervised learning of action primitives," in *ICRA, ICRA10*, 2010, submitted.
- [13] L. R. Rabiner and B. H. Juang, "An introduction to hidden Markov models," *IEEE ASSP Magazine*, pp. 4–15, January 1986.
- [14] D. S. Hirschberg, "Algorithms for the longest common subsequence problem," *J. ACM*, vol. 24, no. 4, pp. 664–675, 1977.
- [15] A. D. Wilson and A. F. Bobick, "Parametric hidden markov models for gesture recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 9, pp. 884–900, 1999.
- [16] X. Huang, Y. Ariki, and M. Jack, *Hidden Markov Models for Speech Recognition*. Edinburgh University Press, 1990.
- [17] D. Herzog, V. Krueger, and D. Grest, "Parametric hidden markov models for recognition and synthesis of movements," in *Proc. British Machine Vision Conference*, Leeds, UK, Sept. 1–4, 2008, pp. 163–172.
- [18] B. Dariush, "Human Motion Analysis for Biomechanics and Biomedicine," *Machine Vision and Applications*, vol. 14, pp. 202–205, 2003.
- [19] J. Wang and B. Bodenheimer, "An Evaluation of a Cost Metric for Selecting Transitions between Motion Segments," in *SIGGRAPH Symposium on Computer Animation*, 2003.
- [20] D. Herzog, A. Ude, and V. Krueger, "Motion imitation and recognition using parametric hidden markov models," in *Humanoids, IEEE-RAS International Conference on Humanoid Robots*, Daejeon, Korea, South, December 1–3, 2008.
- [21] V. Krüger, D. Kragic, A. Ude, and C. Geib, "The meaning of action: A review on action recognition and mapping," *Advanced Robotics*, vol. 21, no. 13, pp. 1473–1501, 2007.
- [22] T. Moeslund, A. Hilton, and V. Krueger, "A survey of advances in vision-based human motion capture and analysis," *Computer Vision and Image Understanding*, vol. 104, no. 2–3, pp. 90–127, 2006.
- [23] M. Lee and R. Nevatia, "Human pose tracking in monocular sequences using multilevel structured models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, pp. 27–38, 2009.
- [24] H. Moon, R. Chellappa, and A. Rosenfeld, "3d object tracking using shape-encoded particle propagation," in *International Conference on Computer Vision*, Vancouver, Canada, July 9–12, 2001.
- [25] J. Gall, J. Patthoff, C. Schnoerr, B. Rosenhahn, and H.-P. Seidel, "Interacting and annealing particle filters: Mathematics and recipe for applications," *Journal of Mathematical Imaging and Vision*, vol. 28, no. 1, pp. 1–18, May 2007.
- [26] M. Isard and A. Blake, "Condensation – conditional density propagation for visual tracking," *International Journal of Computer Vision*, 1998.
- [27] J. Deutscher, A. Blake, and I. Reid, "Articulated body motion capture by annealed particle filtering," in *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, vol. 2, 2000, pp. 126–133 vol.2.
- [28] D. MacKay, *Learning in Graphical Models*, M.Jordan (ed.). MIT Press, 1999, ch. *Introduction to Monte Carlo Methods*, pp. 175–204.
- [29] M. Isard and A. Blake, "Condensation – conditional density propagation for visual tracking," *International Journal of Computer Vision*, vol. 29, no. 1, pp. 5–28, 1998.
- [30] A. Bobick, "Movement, Activity, and Action: The Role of Knowledge in the Perception of Motion," *Philosophical Trans. Royal Soc. London*, vol. 352, pp. 1257–1265, 1997.
- [31] C. Stauffer and W. Grimson, "Learning Patterns of Activity Using Real-Time Tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 747–757, 2000.
- [32] N. Robertson and I. Reid, "Behaviour Understanding in Video: A Combined Method," in *International Conference on Computer Vision*, Beijing, China, Oct 15–21, 2005.
- [33] V. Gallese, L. Fadiga, L. Fogassi, and G. Rizzolatti, "Action recognition in the premotor cortex," *Brain*, vol. 119, no. 2, pp. 593–609, 1996.
- [34] 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.
- [35] 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, November 2006.
- [36] S. Ullman, "High-level vision: Object recognition and visual cognition," July 1996.
- [37] D. Moore, I. Essa, and I. Hayes, M.H., "Exploiting human actions and object context for recognition tasks," vol. 1, pp. 80–86 vol.1, 1999.
- [38] A. Gupta and L. Davis, "Objects in action: An approach for combining action understanding and object perception," pp. 1–8, June 2007.
- [39] T. Inamura, I. Toshima, H. Tanie, and Y. Nakamura, "Embodied symbol emergence based on mimesis theory," *Int. J. Robotics Research*, vol. 23, no. 4–5, pp. 363–377, 2004.
- [40] A. Billard, S. Calinon, and F. Guenter, "Discriminative and adaptive imitation in uni-manual and bi-manual tasks," *Robotics and Autonomous Systems*, vol. 54, pp. 370–384, 2006.
- [41] T. Asfour, F. Gyarfas, P. Azad, and R. Dilmann, "Imitation learning of dual-arm manipulation tasks in humanoid robots," *Int. J. Humanoid Robotics*, vol. 5, no. 2, pp. 183–202, 2008.
- [42] O. Jenkins and M. Mataric, "Deriving Action and Behavior Primitives from Human Motion Data," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Lausanne, Switzerland, Sept.30 – Oct.4, 2002, pp. 2551–2556.
- [43] GGuerra-Filho and Y. Aloimonos, "A language for human action," *IEEE Computer Society*, May 2007.
- [44] Y. Ivanov and A. Bobick, "Recognition of Visual Activities and Interactions by Stochastic Parsing," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 852–872, 2000.

Hands in Action: Real-Time 3D Reconstruction of Hands in Interaction with Objects

Javier Romero

Hedvig Kjellström

Danica Kragic

Abstract—This paper presents a method for vision based estimation of the pose of human hands in interaction with objects. Despite the fact that most robotics applications of human hand tracking involve grasping and manipulation of objects, the majority of methods in the literature assume a free hand, isolated from the surrounding environment. Our hand tracking method is non-parametric, performing a nearest neighbor search in a large database (100000 entries) of hand poses with and without grasped objects. The system operates in real time, it is robust to self occlusions, object occlusions and segmentation errors, and provides full hand pose reconstruction from markerless video. Temporal consistency in hand pose is taken into account, without explicitly tracking the hand in the high dimensional pose space.

I. INTRODUCTION

Articulated tracking and reconstruction of human hands has received an increased interest within the fields of computer vision, graphics and robotics [1] and applications include learning from demonstration, rehabilitation, prosthesis development, human-computer interaction. Our goal is to equip robots with the capability of observing human hands in interaction with objects based solely on vision data, without markers.

Capturing hand articulation from video without markers is a challenging problem. A realistic articulated hand model has at least 28 degrees of freedom, making the state-space very large. The pose estimation suffers from self-similarity – fingers are hard to distinguish from each other – and a high degree of self-occlusion. Furthermore, hands move fast and non-linearly. Any method is thus computationally costly, making real-time implementation demanding. Although there are hand tracking systems developed for specific purposes such as sign recognition [1], full pose estimation remains an open problem, specially if real-time performance is required, as in virtually all robotics applications.

Hand pose estimation methods can largely be divided into two groups [1]: A) *model based tracking* and B) *single frame pose detection*. Methods of type A) usually employ generative articulated models [2], [3], [4]. Due to the high dimensionality of the human hand, they are facing challenges such as high computational complexity and singularities in the state space. They are thus generally unsuitable for robotics applications. Methods of type B) are usually non-parametric [5], [6]. They are computationally less demanding

This work is supported by EU through the project PACO-PLUS, IST-FP6-IP-027657, and GRASP, IST-FP7-IP-215821 and Swedish Foundation for Strategic Research. The authors are with the Computational Vision and Active Perception Lab, Centre for Autonomous Systems, CSC-KTH, Stockholm, Sweden. jrjn,hedvig,dani@kth.se

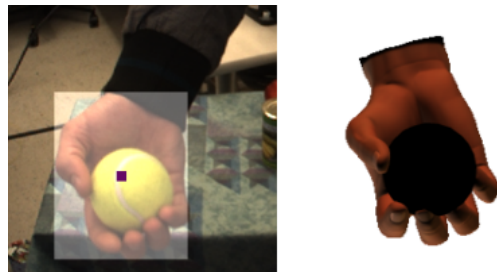


Fig. 1. Left) Original image and Right) Estimated pose.

and more suited for a real-time system, but also more brittle and sensitive to image noise, since there is no averaging over time. In this paper we present a type B) non-parametric pose estimation method (Fig. 1), which takes temporal consistency into account. The probabilistic framework of this method is described in Section II. The method is faster and better at recovering from temporary errors than type A) model-based tracking methods. In an earlier paper [7] we also showed that the time continuity constraint makes the method more accurate and robust than other type B) single frame detection methods.

The method maintains a large database of (synthetic) hand images. Each database instance is labeled with 31 parameters describing the hand articulation and orientation of the hand with respect to the camera. The 31D hand configuration of a new (real) image can then be found using an approximate nearest neighbor approach, taking previous configurations into account. Section II describes the composition of the database. The hand image representation is described in Section IV and the nearest neighbor-based mapping is described in Section V.

In the majority of applications, the human hands are frequently in contact with objects. Despite this, researchers have up to now almost exclusively focused on estimating the pose of hands in isolation from the surrounding scene. A recent notable exception is [8], who describe a type A) model-based tracker that allows for objects in the hand. Our method is also able to reconstruct hands both with and without grasped objects. Reconstruction of a hand grasping an object is in many ways a much more challenging task than reconstruction of a free hand, since the grasped object generally occludes large parts of the hand. The method of [8] allows for hand pose reconstruction *despite* the object occlusion.

On the other hand, knowledge about object shape gives important cues about the configuration of palm and fingers

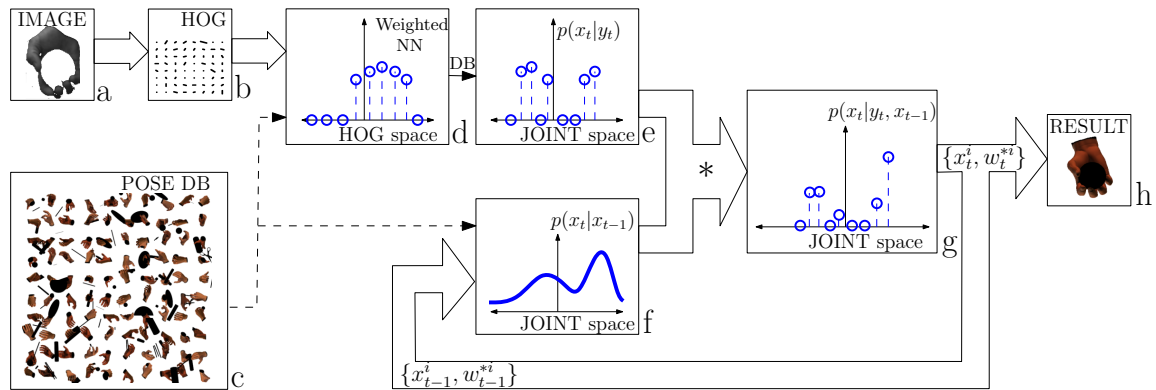


Fig. 2. The non-parametric temporal regression framework.

in contact with the object. Moreover, object shape and functionality give cues as to how this object is generally grasped. The relation between object shape and hand shape is however complex, and this information is hard to exploit in a type A) generative tracking model. In contrast to [8], our method is non-parametric, which means that complex object-hand shape dependencies can be implicitly represented by examples. Hand views in the database depicting grasping hands include occlusion from objects with a shape typical for this kind of grasp (Fig. 1). The occlusion affects the appearance of a hand view, so that hands with similar objects in them will appear similarly. Since the underlying assumption is that appearance similarity implies similarity in hand pose, the object shape *contributes* to the hand pose estimation in our method.

Thus, the main contribution of the paper is a robust non-parametric method for 3D hand reconstruction, operating in real-time, that also takes time continuity constraints into account. The method handles severe occlusions of the hand and also takes the object shape into account in 3D hand reconstruction. Experiments in Section VII also show that the method is robust to segmentation errors, a necessary requirement for the method to be applicable in a realistic setting.

II. PROBABILISTIC FRAMEWORK

The following notation is used throughout the paper. In a specific time instant t , let x_t be the articulated hand pose and y_t the observation. Here, x_t is a 28 dimensional vector of joint angles, and y_t is a 512D histogram of oriented gradients (HOG) [9], see Section IV. The space spanned by x is hereafter called JOINT space, while the space spanned by y is called HOG space. We assume that $p(x_t)$ is uniform over the JOINT space, and that the process is Markovian, i.e., x_t depends on the previous pose x_{t-1} only.

As shown in [7], the view y_t alone is not enough to non-ambiguously estimate the articulated hand pose x_t . Therefore, the pose x_{t-1} at the previous timestep is taken into account in the estimation. This corresponds to sequential estimation of $p(x_t|y_t, x_{t-1})$, the hand pose given the observation and the previous state. The temporal regression problem is decomposed as $p(x_t|y_t, x_{t-1}) \propto p(x_t|y_t)p(x_t|x_{t-1})$. As shown in Fig. 2,

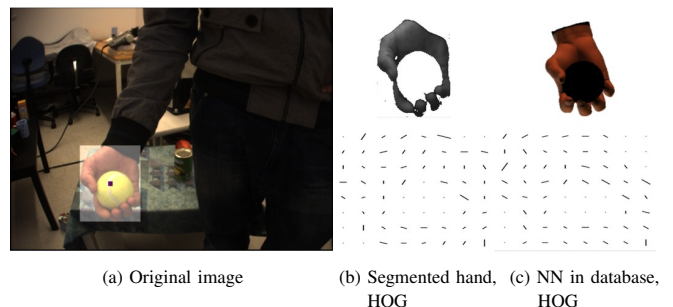


Fig. 3. Data representation.

the method takes as input a monocular image and segments the hand based on skin color segmentation (a). A HOG y_t is then computed as described in Section IV (b).

The HOG y_t is compared to a large database of hand views (c), returning a weighted set of nearest neighbors $\{(y_t^i, x_t^i, w_t^i)\}$, as described in Section V (d). Each neighbor view y_t^i from the database has an associated joint angle configuration x_t^i , which, weighted by w_t^i , constitute a sampled approximation of $p(x_t|y_t)$ (e).

The temporal consistency constraint $p(x_t|x_{t-1})$ is a parametric function of x_t and x_{t-1} , as explained in Section VI (f). This term gives a higher probability to estimates where the hand has moved little over the last time step, thus giving priority to smooth motion estimates. The multiplication with $p(x_t|x_{t-1})$ is approximated by updating the database nearest neighbor weights to $w_t^{*i} \propto w_t^i p(x_t^i|x_{t-1})$ (g).

The expected hand pose value at time t is then estimated as $\hat{x}_t = E(x_t|x_{t-1}, y_t) \approx \arg \max_{x_t^i} w_t^{*i}$, i.e., the database pose with the highest weight (h).

III. DATABASE COMPOSITION

The hand pose x_t could potentially be found by expressing $p(x_t|y_t, x_{t-1})$ parametrically, and finding the maxima of this function using an optimization algorithm. However, this optimization problem is high dimensional and non-convex. To alleviate the dimensionality problem, and constrain the search to commonly observed hand poses, we use a non-parametric approach: we discretize the state space by creating a large database of hand poses with synthetic images.

The composition of the database is motivated by our research aim: understanding human interaction with objects. Our database has more than 10^5 images, consisting of 5 different timesteps of 33 object grasping actions observed from 648 different viewpoints. The grasp types are selected according to the taxonomy presented in [10]. The graphics software Poser 7 is used to generate the synthetic hand views. The synthetic views in the database include basic object shapes that are usually involved in each kind of grasp (see Fig. 3c). The objects are considered background (although colored black for visibility in the figures) and the hand parts occluded by the object do not provide any features to the image observation y_t . This can be seen in Fig. 3c, bottom, where there is a “hole” in the middle of the HOG. As mentioned in the Introduction, the object shape contributes to the hand pose estimation in our method, since the hand pose depends on the shape of the object, which in turn affects the HOG y_t .

It can be argued that this method can only work if the object shape in the real action is the same as in the database. However, firstly, a particular kind of grasp is executed usually to similarly shaped objects and, secondly, the features used in our system (see Section IV) generalizes well over small variations in object shape. As described in Section II, $p(x_t|y_t, x_{t-1})$ is modeled non-parametrically using $\{(y_t^i, x_t^i)\}$, a set of database nearest neighbors to y_t in HOG space, weighted by their distance to y_t in HOG space and x_{t-1} in JOINT space. The weighting is formalized in Sections V and VI.

IV. IMAGE REPRESENTATION

The input to the method are monocular images of the type and quality shown in Figure 3a. In these images, the hand is segmented using skin color thresholding in HSV space [11] (Figure 3b, top). From the segmented hand image a histogram of oriented gradients (HOG) [9] is extracted (Figure 3b, bottom). This is a rich representation of shape, with certain robustness towards segmentation errors and small differences in spatial location and proportions of the segmented hand. The image is partitioned into cells and a histogram of gradient orientation is computed for each cell.

The size of the cells and the granularity of the histograms affect the generalization capabilities of the feature. With smaller cells and detailed histograms, the feature is richer but less capable of generalize over small differences. For our purposes, 8×8 cells and histograms with 8 bins provide good generalization with a sufficient level of details. The observation y_t equals the concatenation of the 8×8 histograms corresponding to each cell of the image. The dimensionality of y_t is thus $8 \times 8 \times 8 = 512$. A more detailed discussion on how different parameters of the HOG affect human detection can be found in [9].

V. NON-PARAMETRIC MAPPING

The probability density function $p(x_t|y_t)$ is approximated by indexing into the database of hand poses using the image

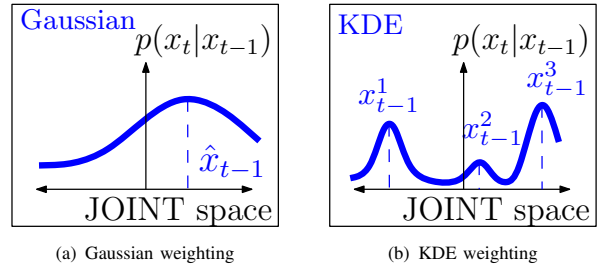


Fig. 4. Two different methods for modeling temporal consistency.

representation y_t , and retrieving the k nearest neighbors (k NN) in the space spanned by y .

As an exact k NN search would put serious limitations on the size of the database, an approximate k NN search method, Locality Sensitive Hashing (LSH) [12] is employed. LSH is a method for efficient ϵ nearest neighbor (ϵ NN) search. It is particularly suited for high dimensional data, since its online complexity does not depend explicitly on the set size or the dimensionality [12].

Each retrieved ϵ NN y_t^i is given a weight $w_t^i = \mathcal{N}(y_t^i|y_t, \sigma_y)$, drawn from a 512D Gaussian density centered in y_t with standard deviation σ_y . This gives higher weight to database ϵ NN that look similar to the observed hand.

In the database, each HOG y^j is associated with a pose x^j . The poses corresponding to the ϵ NN $\{y_t^i\}$ can thus be retrieved. Together with the weights, they form the set $\{(x_t^i, w_t^i)\}$ which is a sampled non-parametric approximation of the density $p(x_t|y_t)$.

The pose vector x is composed of the rotation matrix of the wrist wrt the camera and the sines of the joint angles of the hand (which takes values between $[-\frac{\pi}{2}, \frac{\pi}{2}]$). Each component of x therefore lie in the domain $[-1, 1]$, which makes scaling unnecessary. The advantage of using a rotation matrix to represent the wrist rotation is that rotation matrices can be compared in a Euclidean fashion, as opposed to Euler angles and quaternions. Euclidean comparison of poses is used in the temporal consistency modeling (Section VI) and the experimental evaluation (Section VII-A).

VI. TEMPORAL CONSISTENCY MODELING

As described in Section II, the temporal consistency constraint $p(x_t|x_{t-1})$ is modeled as a parametric function. It is used to reweight the sampled distribution $\{(x_t^i, w_t^i)\}$, approximating $p(x_t|y_t)$. We propose two ways to model the temporal consistency constraint, outlined in the two subsections below.

A. Single Hypothesis Gaussian Weighting

The simplest way of modeling temporal consistency is to assume that poses similar to the previous estimated pose \hat{x}_{t-1} are more likely than poses that are very different from the previous one. Hence, $p(x_t|x_{t-1}) = \mathcal{N}(x_t|\hat{x}_{t-1}, \sigma_x)$, a 28D Gaussian density centered in \hat{x}_{t-1} with standard deviation σ_x . This approach was used in [7] and is depicted in Figure 4a.

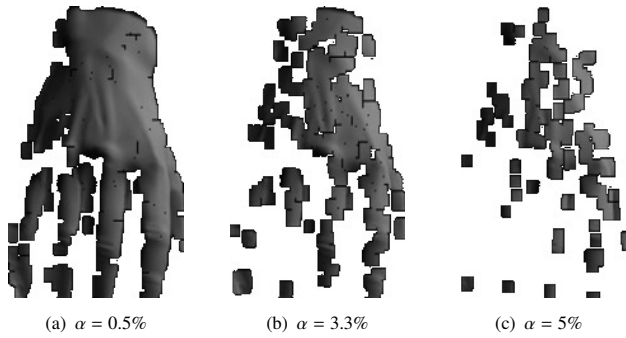


Fig. 5. Artificial segmentation corruption α added to synthetic sequences.

B. Multiple Hypothesis Kernel Density Estimation Weighting

A drawback of the single hypothesis approach is that all the “second best” nearest neighbor hypotheses at $t - 1$ are thrown away before temporal propagation. A logical improvement is to consider the full weighted set of hypotheses $\{(x_{t-1}^i, w_{t-1}^{*i})\}$ instead of the most likely hypothesis \hat{x}_{t-1} in the estimation of $p(x_t|x_{t-1})$. This is illustrated in Figure 4b.

Following this idea, we use kernel density estimation (KDE) [13] over the weighted set of poses of the previous frame $\{(x_{t-1}^i, w_{t-1}^{*i})\}$ to estimate $p(x_t|x_{t-1})$. The system can then recover from an erroneous estimation of x_{t-1} .

As shown in the experiments in Section VII, KDE leads to a more robust sequential estimation than Gaussian weighting in many cases. Furthermore, even though KDE increases the computational load with a factor corresponding to the number of nearest neighbors $|\{x_{t-1}\}|$, the computational load of computing the temporal consistency weights is negligible compared to, e.g., the database ϵ NN lookup. A drawback of KDE compared to Gaussian weighting is however the necessity of tuning more parameters, most importantly, the bandwidth of the kernels.

VII. EXPERIMENTS

We first experimentally compare the two temporal consistency models detailed in Section VI, using synthetic sequences with hand pose ground truth. Then, the method is evaluated on real sequences featuring three different subjects and three object shapes. The sequences were captured at 10 frames/sec with a Point Grey Dragonfly camera with a resolution of 640×480 pixels. The method was implemented in C++ and runs at 10 frames/sec on one of the cores of a four core 2.66GHz Intel processor.

A. Comparison of Temporal Consistency Models

The single hypothesis and multiple hypothesis temporal consistency models are first compared in terms of pose reconstruction accuracy. This quantitative analysis of our method is done with synthetic sequences, where the hand pose ground truth is available. To make experimental conditions as realistic as possible, none of the hand poses or the objects in the synthetic sequences are present in the database. Moreover, the poses are corrupted with a variable amounts of segmentation noise (see Fig. 5), to simulate

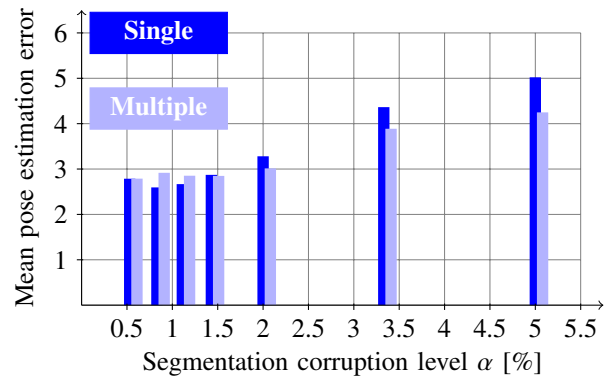


Fig. 8. Pose error with increasing segmentation corruption in sequence 1.

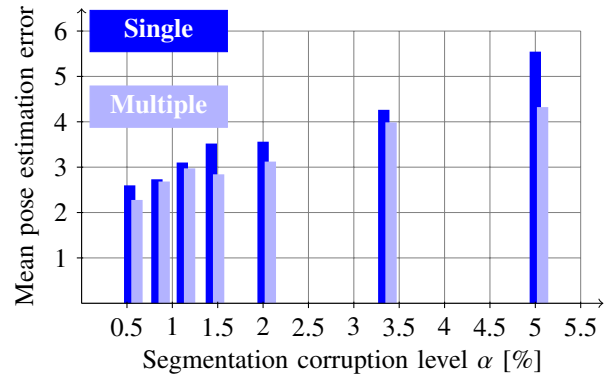


Fig. 9. Pose error with increasing segmentation corruption in sequence 2.

segmentation errors that occur with real sequences. The segmentation corruption is performed in the following way: The segmentation mask is first assigned as the full hand view (without noise). A fraction α of the pixels in the segmentation mask are set to zero. The error is then propagated through an erosion followed by dilation. In each frame t , the error of the estimated hand pose \hat{x}_t relative to the ground truth pose x_t^{gt} is estimated as $\|\hat{x}_t - x_t^{\text{gt}}\|$, the Euclidean distance in the pose space explained in Section V. Figures 6 and 7 show the hand pose estimation of synthetic sequences 1 and 2 respectively, with segmentation corruption $\alpha = 0.5\%$.

As shown in Fig. 8-9, the multiple hypothesis temporal consistency model almost consistently gives a better accuracy. The effect is more visible with higher segmentation corruption levels α . The reason for this is that the single-frame pose estimate $p(x_t|y_t)$ is more ambiguous for higher α , which means that there is a higher uncertainty about which sample x_t^i is the best pose estimate at time t . With higher α it is thus increasingly better to let all samples $\{(x_{t-1}^i, w_{t-1}^{*i})\}$ influence the temporal model. It can also be seen that the pose estimation performance is largely unaffected by segmentation corruption levels up to $\alpha = 2\%$.

B. Real Sequences with Subjects Not in Database

To show the performance of the method on real data, it was evaluated with sequences of the first author and two un instructed persons (one man and one woman) grasping



Fig. 6. Synthetic sequence 1. Top: original synthetic image. Middle: segmentation image with $\alpha = 0.5\%$. Bottom: estimated pose. (The objects in the database are colored black for visibility here, but do not contribute to the HOGs.) Video at www.csc.kth.se/~jrjn/VideosICRA2010/synthetic1.mp4

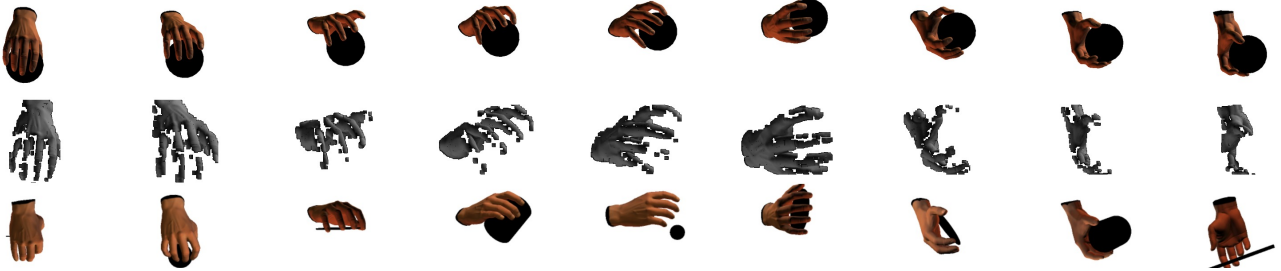


Fig. 7. Synthetic sequence 2. Top: original synthetic image. Middle: segmentation image with $\alpha = 0.5\%$. Bottom: estimated pose. (The objects in the database are colored black for visibility here, but do not contribute to the HOGs.) Video at www.csc.kth.se/~jrjn/VideosICRA2010/synthetic2.mp4

three different objects: A cup (with no equivalence in the database), a tennis ball (similar to a ball in the database), and a pair of pliers (with no equivalence in the database). The actions are not required to start from any specific pose. Naturally, the grasps in the sequences do not have exact correspondences in the database. Furthermore, the subjects' hands are of different sizes and shapes.

The multiple hypothesis temporal consistency modeling, shown above to be consistently better than the single hypothesis alternative, was used throughout the real image experiments. Fig. 10, 11, and 12, show the result of pose estimation for the three subjects respectively.

One conclusion that can be drawn is that the method is robust to individual variations in hand shape and proportions. The hand model used to generate the database view is designed to be male. However, the method is successful in recovering the poses of the considerably more slender female hand (Fig. 12), as well as of the hand with a larger proportion of the lower arm uncovered (Fig. 11); this affects skin segmentation, which in turn affects the HOG y_t used for database lookup.

The results also show that the method generalizes over grasps and objects that are not exactly represented in the database. It should be taken into account that two of the subjects have no previous experience with the method or the database, and thus can be expected to grasp the objects in a natural way. The cup and the ball are well represented by other objects present in the database. However, the pliers pose a slightly larger challenge for the method. There are two possible reasons for this. Firstly, the layout of the pliers, with two separated legs, makes the occlusion of the hand appear differently than any example in the database. Secondly,

the functionality of the pliers makes the subjects grasp it differently than other rod-like structures in the database. Fig. 13 shows the pose estimation of a sequence where large parts of the hand is occluded by the grasped object showing the method is robust to large object occlusion.

The pose estimation in Fig. 14 points to an avenue for improvement of the method. In our current temporal continuity approaches we assume that the most probable current pose is similar to the most probable previous pose. With this we are making an implicit assumption of static hand pose. However, this assumption is frequently violated; fast hand motions like the one shown at the end of the sequence in Figure 14 are not uncommon. With the assumption of being static in the temporal consistency model, all poses x_t^i selected by the ϵ NN sampling will be equally unlikely according to the temporal consistency model. Ambiguities in the HOG signature, e.g., between the front and back part of the hand, will then cause estimation errors as the one in the leftmost frame of Fig. 14. This issue can be addressed by including a dynamic model of pose over time.

VIII. CONCLUSIONS

A non-parametric method for 3D sequential pose estimation of hands in interaction with objects was presented. The contributions of this paper are the development of a method that not only handles severe occlusion from objects in the hand, but also takes the object shape into account in 3D hand reconstruction. In addition, the method is non-parametric and provides 3D hand reconstruction, operating in real-time, taking time continuity constraints into account.

Experiments showed that the method estimates hand pose in real time robustly against segmentation errors and large occlusion of the hand from objects. It was also shown that

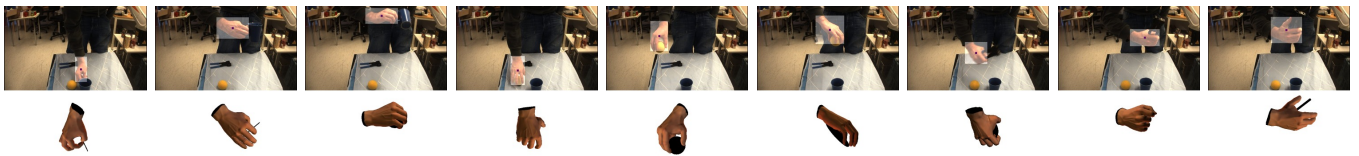


Fig. 10. Real sequence 1 (male subject 1). Top: image with skin segmentation window highlighted. Bottom: estimated pose. (The objects in the database are colored black for visibility here, but do not contribute to the HOGs.)
Video at www.csc.kth.se/~jrgn/VideosICRA2010/real1.mp4



Fig. 11. Real sequence 2 (male subject 2). Top: image with skin segmentation window highlighted. Bottom: estimated pose. (The objects in the database are colored black for visibility here, but do not contribute to the HOGs.)
Video at www.csc.kth.se/~jrgn/VideosICRA2010/real2.mp4



Fig. 12. Real sequence 3 (female subject 3). Top: image with skin segmentation window highlighted. Bottom: estimated pose. (The objects in the database are colored black for visibility here, but do not contribute to the HOGs.)
Video at www.csc.kth.se/~jrgn/VideosICRA2010/real3.mp4

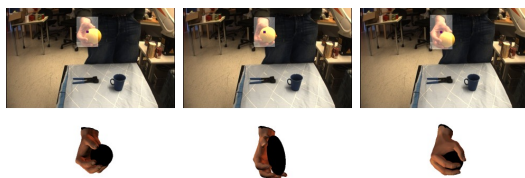


Fig. 13. Real sequence 4 (male subject 1) with large hand occlusion. Top: image with skin segmentation window highlighted. Bottom: estimated pose.
Video at www.csc.kth.se/~jrgn/VideosICRA2010/real4.mp4

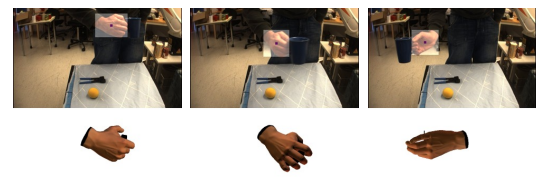


Fig. 14. Real sequence 5 (male subject 1) with fast non-linear motion. Top: image with skin segmentation window highlighted. Bottom: estimated pose.
Video at www.csc.kth.se/~jrgn/VideosICRA2010/real5.mp4

the robustness to temporary estimation errors is improved by taking multiple hypotheses of previous hand pose into account.

Future work includes improving the motion model; currently, a static temporal model is implicitly assumed. This can be done in several ways, e.g., by learning low-dimensional models of hand motion from motion capture training data. Furthermore, we will enlarge the database to represent poses of differently shaped hands, grasping a wider range of objects under different illumination conditions. The approximate database lookup has a highly sub-linear time complexity, which allows for a significantly larger database with a moderate increase in computational load.

REFERENCES

- [1] A. Erol, G. N. Bebis, M. Nicolescu, R. D. Boyle, and X. Twombly, "Vision-based hand pose estimation: A review," *Computer Vision and Image Understanding*, vol. 108, pp. 52–73, 2007.
- [2] B. D. R. Stenger, A. Thayananthan, P. H. S. Torr, and R. Cipolla, "Filtering using a tree-based estimator," in *IEEE International Conference on Computer Vision*, 2003.
- [3] E. Sudderth, M. I. Mandel, W. T. Freeman, and A. S. Willsky, "Visual hand tracking using non-parametric belief propagation," in *IEEE Workshop on Generative Model Based Vision*, 2004.
- [4] M. de la Gorce, N. Paragios, and D. J. Fleet, "Model-based hand tracking with texture, shading and self-occlusions," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–8.
- [5] V. Athitsos and S. Sclaroff, "Estimating 3D hand pose from a cluttered image," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2003, pp. 432–439.
- [6] H. Kjellström, J. Romero, and D. Kragić, "Visual recognition of grasps for human-to-robot mapping," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008.
- [7] J. Romero, H. Kjellström, and D. Kragić, "Monocular real-time 3D articulated hand pose estimation," in *IEEE-RAS International Conference on Humanoid Robots*, 2009.
- [8] H. Hamer, K. Schindler, E. Koller-Meier, and L. Van Gool, "Tracking a hand manipulating an object," in *IEEE International Conference on Computer Vision*, 2009.
- [9] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2005, pp. I: 886–893.
- [10] T. Feix, R. Pawlik, H. Schmiedmayer, J. Romero, and D. Kragic, "A comprehensive grasp taxonomy," in *Robotics, Science and Systems Conference: Workshop on Understanding the Human Hand for Advancing Robotic Manipulation*, June 2009.
- [11] A. A. Argyros and M. I. A. Lourakis, "Real time tracking of multiple skin-colored objects with a possibly moving camera," in *European Conference on Computer Vision*, vol. 3, 2004, pp. 368–379.
- [12] W. Dong, Z. Wang, M. Charikar, and K. Li, "Efficiently matching sets of features with random histograms," in *ACM Multimedia*, 2008.
- [13] V. Morariu, B. Srinivasan, V. Raykar, R. Duraiswami, and L. Davis, "Automatic online tuning for fast gaussian summation," in *Neural Information Processing Systems*, 2008, pp. 1113–1120.

Monocular Real-Time 3D Articulated Hand Pose Estimation

Javier Romero Hedvig Kjellström Danica Kragic
Computational Vision and Active Perception Lab
Centre for Autonomous Systems
School of Computer Science and Communication
KTH, SE-100 44 Stockholm, Sweden
jrgn,hedvig,dani@kth.se

Abstract—Markerless, vision based estimation of human hand pose over time is a prerequisite for a number of robotics applications, such as Learning by Demonstration (LbD), health monitoring, teleoperation, human-robot interaction. It has special interest in humanoid platforms, where the number of degrees of freedom makes conventional programming challenging. Our primary application is LbD in natural environments where the humanoid robot learns how to grasp and manipulate objects by observing a human performing a task. This paper presents a method for continuous vision based estimation of human hand pose. The method is non-parametric, performing a nearest neighbor search in a large database (100000 entries) of hand pose examples. The main contribution is a real time system, robust to partial occlusions and segmentation errors, that provides full hand pose recognition from markerless data. An additional contribution is the modeling of constraints based on temporal consistency in hand pose, without explicitly tracking the hand in the high dimensional pose space. The pose representation is rich enough to enable a descriptive human-to-robot mapping. Experiments show the pose estimation to be more robust and accurate than a non-parametric method without temporal constraints.

I. INTRODUCTION

Vision based, markerless human hand tracking in natural environments with and without interaction with objects is an important building block for various human-machine interaction and robot learning tasks. An important aspect considered in our work is enabling robots to learn how to grasp and manipulate objects just by observing humans. Another aspect is monitoring of humans in everyday environments for designing hand prosthesis able of performing most common human grasps. However, capturing hand articulation is a challenging problem. Using the joint angle representation of hand pose requires 28-dimensional configuration space. In addition, self-occlusions of fingers introduce uncertainty for the occluded parts. Although there have been examples of systems that can track hands for very specific purposes such as sign recognition, full pose estimation remains an open problem, specially if real-time performance is required.

In robotic applications, an important aspect of task modeling is how different objects involved in the task should be grasped and manipulated. Humanoid robots are equipped with more and more dexterous humanoid hands, capable of perform human-like grasps. However, the control of these hands is far from trivial; therefore LbD is an attractive way of teaching the robot how to grasp [1]. While observing the human, the robot must estimate the human hand pose

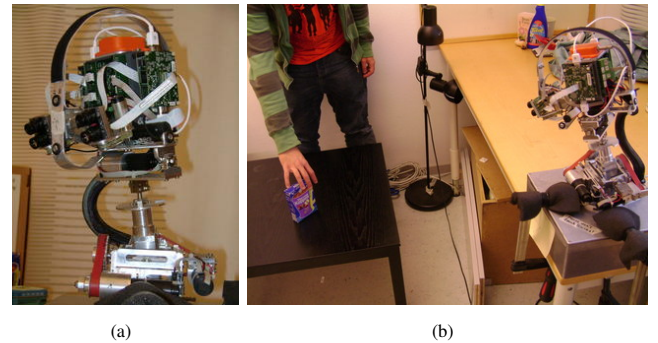


Fig. 1. a) ARMAR head, b) ARMAR head observing human grasp demonstration

over time, and then map the hand pose to its own hands or grippers. In this paper we focus on visual estimation of human hand motion during object manipulation. While hand motion can be robustly extracted using 3D magnetic sensors or datagloves [2], the usability of a home service robot is compromised if the user is required to carry special markers during task instruction. The visual hand pose estimation is therefore required to be markerless.

Humanoid heads are constraint to have small baseline, lightweight stereo vision systems (see Figure 1). This makes the stereo-matching problem difficult and sometimes inaccurate, specially for textureless surfaces as human hands. For this reason visual hand pose estimation based on monocular images can be an attractive field for humanoid robot research.

Markerless 3D reconstruction of hand pose based on a single image is an extremely difficult problem due to the large self-occlusion, high dimensionality and non-linear motion of the fingers. There are different ways of addressing these difficulties. Hand pose estimation method can largely be divided into two groups [3]: *model based tracking* and *single frame pose estimation*. Due to the high dimensionality of the human hand, articulated 3D model based trackers are facing challenges such as high computational complexity and singularities in the state space [4]. Single frame pose estimation is usually more computationally efficient than model based tracking, but lacks the notion of temporal consistency, which is an important cue to hand pose [5], [6].

In earlier work [6], we presented a method for non-parametric estimation of grasp type and hand orientation

from a single monocular image. The method maintained a large database of (synthetic) hand images. Each database instance was labeled with the grasp type and the orientation of the hand with respect to the camera. The grasp type and orientation of a new (real) image could then be found using a nearest neighbor approach. For completeness, the hand image representation is described in Section III and the nearest neighbor-based mapping is described in Section IV.

In the current work, we have further developed the initial approach in two ways; I) by including temporal consistency in the distance measure used for database retrieval. This greatly enhances the robustness of the hand pose estimation, as it will be shown in Section VI; II) by extending the state space to a full joint angle representation, allowing a full 3D reconstruction of hand pose. This facilitates the learning of rich human-to-robot hand pose mapping. Development II) is the main contribution of this paper, described in more detail in Section IV and it is possible in part because of Development I), which is a secondary contribution, detailed in Section V.

Experiments in Section VI show that we can reconstruct the hand pose in real time and that our method is considerably robust to segmentation errors, a necessary requirement for the method to be applicable in a realistic setting. Additionally, it is shown that the temporal consistency constraint has a profound effect on the pose estimation accuracy and robustness.

II. RELATED WORK

Analysis of human hand pose for the purpose of LbD [7] has been thoroughly investigated, almost exclusively with the help of markers and/or 3D sensors attached to the human hand [2]. However, we envision a LbD scenario where the teaching process can be initiated without calibration and where the robot-user interaction is as natural as possible. For this reason we want to reconstruct the hand posture in a visual markerless fashion.

The field of markerless visual hand pose estimation has been mainly devoted to hand gesture or sign language recognition [8]. A common approach is to estimate the hand pose from a single frame and use this pose as the input to a recognition module [5], [9], [10]. The pose estimation is made easier by the fact that the range of poses can be constrained to the discrete set of specific gestures.

Methods for hand pose estimation that are not constrained to a limited set of poses can largely be classified into two groups [3]: I) model based tracking and II) single frame pose estimation. Methods of type I) usually employ generative articulated models [11], [4]. Since the state space of a human hand is extremely high-dimensional, they are generally very computationally demanding, which currently makes this approach intractable for a robotics application. Methods of type II) are usually non-parametric [6]. They are less computationally demanding and more suited for a real-time system, but also more brittle and sensitive to image noise, since there is no averaging over time. The method presented here falls into the second approach. However, it

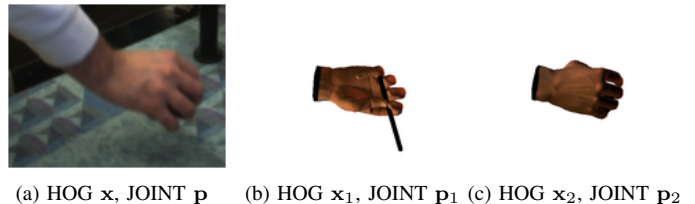


Fig. 2. Ambiguity in mapping from HOG space to JOINT space. Even though it is visually apparent that $\|\mathbf{p} - \mathbf{p}_2\| \ll \|\mathbf{p} - \mathbf{p}_1\|$ in JOINT space, database instance 1 will be regarded as the nearest neighbor as $\|\mathbf{x} - \mathbf{x}_1\| < \|\mathbf{x} - \mathbf{x}_2\|$. Note that the object in the hand just contributes with occlusion of the hand in HOG extraction, as it is then colored uniformly with background color.

takes temporal continuity into account and it can be used for on-line real-time reconstruction.

For LbD purposes, it is relevant to investigate what hand pose information the robot needs in order to perform a successful human-to-robot mapping of the hand motion. In [12], [13] the control of a grasping hand was performed from a low dimensional space thanks to dimensionality reduction techniques.

III. IMAGE REPRESENTATION

The input to the method is a sequence $\{\mathbf{I}_t\}, t = 1, \dots, n$ of monocular images of the human hand. The same image representation was used in our previous work [6], where a more elaborate description can be found.

In each frame \mathbf{I}_t , the hand is segmented using skin color segmentation based on color thresholding in HSV space. The result is a segmented hand image \mathbf{H}_t . Due to a number of factors such as image noise, skin color in the background and non-skin colored areas on the hand (e.g. jewellery), the segmentation is more or less erroneous.

The shape information contained in \mathbf{H}_t is represented with a Histogram of Oriented Gradients (HOG). This feature has been frequently used for representation of human and hand shape [14], [15]. It has the advantage of being robust to small differences in spatial location and proportions of the depicted hand, while capturing the shape information effectively.

Gradient orientation $\Phi_t \in [0, \pi)$ is computed from the segmented hand image \mathbf{H}_t as $\Phi_t = \arctan(\frac{\partial \mathbf{H}_t}{\partial y} / \frac{\partial \mathbf{H}_t}{\partial x})$.

From Φ_t , a pyramid with L levels of histograms with different spatial resolutions are created; on each level l , the gradient orientation image is divided into $2^{L-l} \times 2^{L-l}$ equal partitions. A histogram with B bins is computed from each partition.

The hand view at time t is represented by the HOG \mathbf{x}_t which is the concatenation of all histograms at all levels in the pyramid. The length of \mathbf{x}_t is thus $B \sum_{l=1}^L 2^{2(L-l)}$. Empirically, we obtained the best performance with a reasonable running time using $B = 8$ and $L = 3$. A discussion about how different parameters of the HOG affect human detection can be found in [16].

IV. NON-PARAMETRIC POSE RECONSTRUCTION

In this section, we regard the problem of estimating a single pose \mathbf{p} from a single HOG \mathbf{x} omitting the time index.

The goal of the hand pose reconstruction process is to find the mapping $\hat{\mathbf{p}} = \mathcal{M}(\mathbf{x})$, where $\hat{\mathbf{p}}$ is the estimated 31D hand pose in terms of global orientation (lower arm yaw, pitch, roll) and joint angles (3 wrist joint angles, 5 joint angles per finger), and \mathbf{x} is the observed 168D HOG representation of the hand view, described in Section III.

The mapping function \mathcal{M} can be expected to be highly non-linear in the HOG space, with large discontinuities. Following [6], \mathcal{M} is therefore represented non-parametrically, i.e., as a database of example tuples $\{\langle \mathbf{x}_i, \mathbf{p}_i \rangle\}, i \in [1, N]$. Due to the high dimensionality of both the HOG space (168D) and the state space (hereafter denoted JOINT space, 31D), the database needs to be of a considerable size to cover all hand poses to be expected; in our current implementation, $N = 90000$. This has two implications for our mapping method, as outlined in the subsections below.

A. Generation of Database Examples

Generating a database of 10^5 examples from real images is intractable. Instead, we used the graphics software Poser 7 to generate synthetic views $\mathbf{H}_i^{\text{synth}}$ (see Figure 4) of different poses. The database was generated offline and it took around 5 days to render all the poses on a standard desktop computer. We are here motivated by the LbD application where we envision human to perform different types of grasps on objects in the environment. Therefore, the database examples are chosen as frames from short sequences of:

- 1) different grasp types, from
- 2) different view points, with
- 3) different grasped objects, and with
- 4) different illuminations.

The grasp types are selected according to the taxonomy developed in the GRASP project¹, which integrates the Cutkosky [17], Kamakura [18], and Kang [19] taxonomies. The whole database is also available at the same place. For each grasp type, a number of poses from whole grasp sequences (rest, approach and grasp) are included. Each pose is rendered with four different illuminations and from 386 different points of view uniformly distributed on a sphere. Standard objects are included to simulate typical occlusions.

From each example view $\mathbf{H}_i^{\text{synth}}$, the tuple $\langle \mathbf{x}_i, \mathbf{p}_i \rangle$ is extracted, where \mathbf{x}_i is generated from $\mathbf{H}_i^{\text{synth}}$ as described in Section III, and \mathbf{p}_i is the pose used to generate the view $\mathbf{H}_i^{\text{synth}}$ in Poser 7.

B. Approximate Nearest Neighbor Extraction

Given an observed HOG \mathbf{x} , the goal is to find an estimated pose $\hat{\mathbf{p}} = \mathcal{M}(\mathbf{x})$. With the non-parametric mapping approach, the mapping task $\hat{\mathbf{p}} = \mathcal{M}(\mathbf{x})$ is one of searching the database for examples $\langle \mathbf{x}_i, \mathbf{p}_i \rangle$ such that $\mathbf{x}_i \approx \mathbf{x}$. More formally, X_k , the set of k nearest neighbors to \mathbf{x} in terms of Euclidean distance in HOG space, $d_i = \|\mathbf{x} - \mathbf{x}_i\|$ are retrieved.

As an exact k NN search would put serious limitations on the size of the database, an approximate k NN search method,



Fig. 4. Synthetic sequence not contained in the database. Note that the object in the hand just contributes with occlusion of the hand in HOG extraction, as it is then colored uniformly with background color.

Locality Sensitive Hashing (LSH) [20] is employed. LSH is a method for efficient ϵ -nearest neighbor (ϵ NN) search, i.e. the problem of finding a neighbor $\mathbf{x}_{\epsilon\text{NN}}$ for a query \mathbf{x} such that

$$\|\mathbf{x} - \mathbf{x}_{\epsilon\text{NN}}\| \leq (1 + \epsilon)\|\mathbf{x} - \mathbf{x}_{\text{NN}}\| \quad (1)$$

where \mathbf{x}_{NN} is the true nearest neighbor of \mathbf{x} .

The number of hyperplanes and number of tables used in the LSH search are learned from the database, as explained in [20]. In our current implementation, $K = 30$ and $T = 5000$.

The computational complexity of ϵ NN retrieval with LSH [20] is $\mathcal{O}(DN^{\frac{1}{1+\epsilon}})$ which gives sublinear performance for any $\epsilon > 0$.

C. The Mapping \mathcal{M} is Ambiguous

The database retrieval described above constitutes an approximation to the true mapping $\hat{\mathbf{p}} = \mathcal{M}(\mathbf{x})$, robust to singularities and discontinuities in the mapping function \mathcal{M} .

However, it can be shown empirically that \mathcal{M} is inherently ambiguous (one-to-many); substantially different poses \mathbf{p} can give rise to the similar HOGs \mathbf{x} [14]. An example of this is shown in Figure 2.

Thus, the true pose \mathbf{p} can not be fully estimated from a single HOG \mathbf{x} (using any regression or mapping method); additional information is needed. In the next section, we describe how temporal continuity assumptions can be employed to disambiguate the mapping from HOG to hand pose.

V. TIME CONTINUITY ENFORCEMENT IN JOINT SPACE

We now describe how temporal smoothness in hand motion can be exploited to disambiguate the mapping \mathcal{M} .

Consider a sequence of hand poses $[\mathbf{p}_t], t = 1, \dots, n$, that have given rise to a sequence of views, represented as HOGs $[\mathbf{x}_t], t = 1, \dots, n$. Since the mapping \mathcal{M} is ambiguous, the k nearest neighbors to \mathbf{x}_t in the database, i.e. the members of the set X_k , are all similar to \mathbf{x}_t but not necessarily corresponding to hand poses similar to \mathbf{p}_t . An important implication of this is that a sequence of hand poses $[\mathbf{p}_t], t = 1, \dots, n$ does not necessarily give rise to a sequence of HOGs $[\mathbf{x}_t], t = 1, \dots, n$ continuous in the HOG space. This is illustrated in the upper part of Figure 3, where we see that the red crossed arrow forcing continuity in HOG space points to the wrong pose.

This property of the data makes the problem of continuous hand pose recognition intrinsically different to other continuous NN problems found in the literature. For example, in [21] the “visible” feature displays time continuity, thus allowing the k NN answers from previous time steps to guide a new k NN query.

¹www.grasp-project.eu.

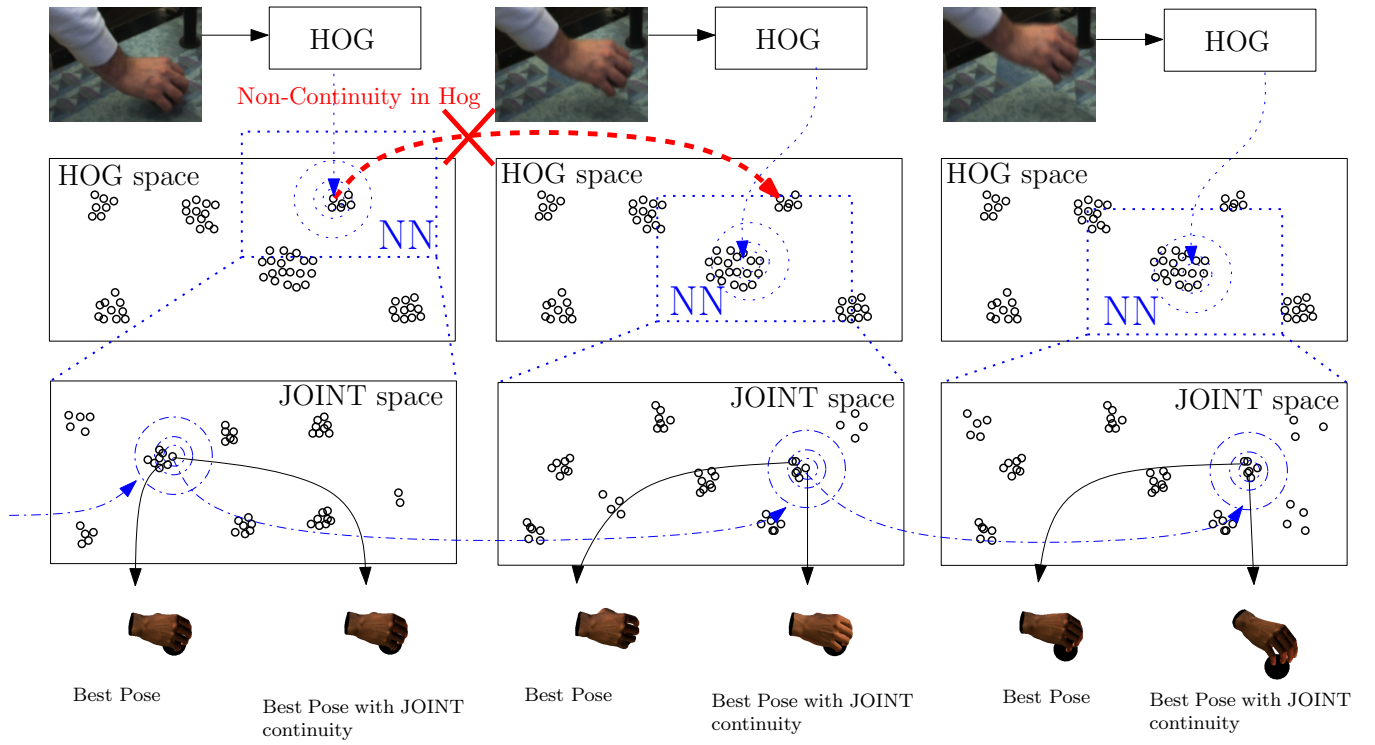


Fig. 3. Due to the underlying physics, a sequence of poses is continuous in the JOINT space, but not in HOG space.

However, due to the physics of the human body, the speed of the hand articulation change is limited. Thus, the sequence of hand poses $[\mathbf{p}_t], t = 1, \dots, n$, i.e. the *hidden variables*, display a certain continuity in the JOINT space. This is illustrated in Figure 3.

The hand pose recognition for a certain frame t is therefore divided into two stages; I) retrieval of a set of k nearest neighbors X_k using single frame non-parametric mapping, as described in Section IV; II) weighting of the members of X_k according to their time continuity in the JOINT space.

Let P_k be the set of poses corresponding to the k NN set X_k found in stage I). Moreover, let $\hat{\mathbf{p}}_{t-1}$ be the estimated pose in the previous time step. In stage II), the members $\mathbf{p}_j, j \in [1, k]$ of P_k are weighted as

$$\omega_j = e^{-\frac{\|\mathbf{p}_j - \hat{\mathbf{p}}_{t-1}\|}{2\sigma^2}}. \quad (2)$$

where σ^2 is the variance of the distance from each entry pose \mathbf{p}_j to the previous estimated pose $\hat{\mathbf{p}}_{t-1}$.

The pose estimate at time t is computed as the weighted mean of P_k :

$$\hat{\mathbf{p}}_t = \left(\sum_{j=1}^k \omega_j \mathbf{p}_j \right) / \left(\sum_{j=1}^k \omega_j \right). \quad (3)$$

It should be noted that this is very similar in spirit to temporal filtering. The main difference is that a filtering approach can be regarded as *top-down*, making predictions about future poses according to some motion model, predicting how the observations of those prior poses should appear, and comparing the expected observations with the actual observations. Our approach can instead be regarded as *bottom-up*, making estimates directly from the observations, and then evaluating them in terms of the motion model.

In order to weight the poses $\mathbf{p}_j, \hat{\mathbf{p}}_{t-1}$ could be substituted by more complex predictions such as Kalman Filters or Particle Filters. However, the dynamics of the joints are not easy to model, so we preferred to keep the assumption about the dynamics as simple as possible as a first step. We leave the inclusion of a particle filter predictor for future work.

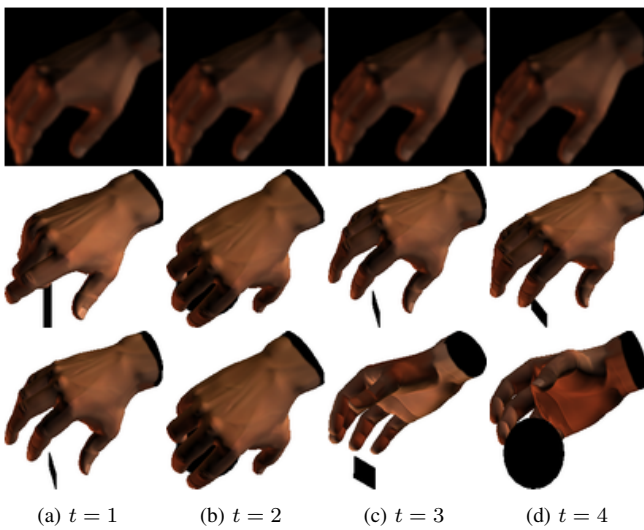


Fig. 5. Recognition of hand pose with perfect segmentation. Row 1: query pose \mathbf{p}_t ; Row 2: estimated pose $\hat{\mathbf{p}}_t$; Row 3: estimated pose $\hat{\mathbf{p}}_t^{\text{uniform}}$.

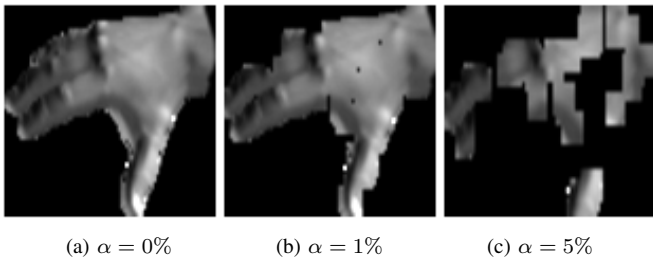


Fig. 6. Synthesizing imperfect segmentation for synthetic images with 3 noise levels: fraction α pixels removed, followed by an opening-closing operation on the image.

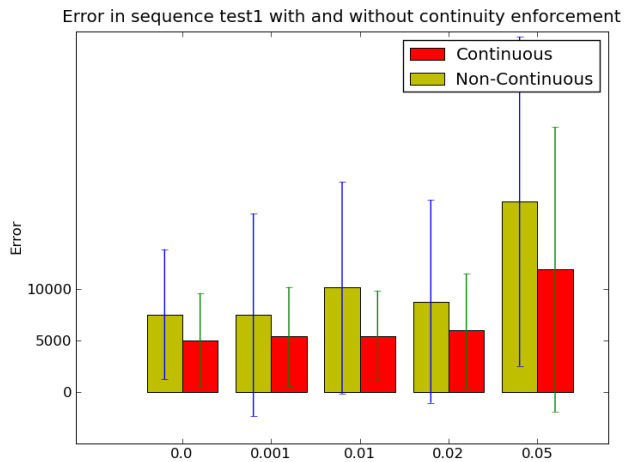


Fig. 7. Mean square error of 3D pose vector for continuous and non-continuous recognition

VI. EXPERIMENTS

The experiments are designed to measure the effect of taking time continuity into account in the hand pose estimation as described in Equations (2), (3) as opposed to unweighted averaging

$$\hat{\mathbf{p}}_t^{\text{uniform}} = \left(\sum_{j=1}^k \mathbf{p}_j \right) / k. \quad (4)$$

Firstly, a quantitative analysis is made, using a synthetic sequence not included in the database. Secondly, the performance of the method is qualitatively evaluated on real images with hand poses not included in the database.

A. Quantitative Analysis

It is difficult to obtain ground truth poses \mathbf{p}_t for a real image sequence; this would mean introducing markers, which would seriously affect the appearance of the hand. Therefore, a synthetic sequence is created, shown in Figure 4. The sequence depicts a typical approach-grasp action. Neither the rest position, the pose after the approach nor the final grasp pose are included in the database.

The quality of the estimated pose vector $\hat{\mathbf{p}}_t$ is measured in terms of Euclidean distance from the ground truth pose vector \mathbf{p}_t in JOINT space: $E_t = \|\hat{\mathbf{p}}_t - \mathbf{p}_t\|$.



Fig. 8. General comparison. Row 1: query pose \mathbf{p}_t , not included in the database; Row 2: estimated pose $\hat{\mathbf{p}}_t$; Row 3: estimated pose $\hat{\mathbf{p}}_t^{\text{uniform}}$.

Figure 5 shows reconstructed poses $\hat{\mathbf{p}}_t$ compared to the baseline of $\hat{\mathbf{p}}_t^{\text{uniform}}$. The time continuity constraint is clearly effective: The estimates $\hat{\mathbf{p}}_t^{\text{uniform}}$ are much more incoherent over time than $\hat{\mathbf{p}}_t$. Figure 7, leftmost bar, shows that the mean error of sequence $[\hat{\mathbf{p}}_t], t = 1, \dots, n$ is 50% lower than that of $[\hat{\mathbf{p}}_t^{\text{uniform}}], t = 1, \dots, n$.

The comparison becomes more valid if we simulate realistic image noise conditions for this synthetic sequence. Noise is thus introduced in the segmentation of the image, in order to simulate imperfect segmentation in real sequences. This is done by removing a certain fraction of the pixels in the segmentation mask, followed by opening-closing morphological operations. Figure 6 shows how this operation affects the segmentation mask.

Figure 7 shows how the error (vertical axis) develops as the image segmentation noise level increases (horizontal axis). It is apparent that the estimation with pose continuity is much more robust to segmentation errors up to $\alpha = 2\%$. $\alpha = 5\%$ there is an abrupt increase in error for both methods, indicating that the segmentation (Figure 6c) then is too poor to yield descriptive HOGs.

B. Qualitative Analysis

The algorithm was also evaluated with a real image sequence without known ground truth. The sequence contains grasps that do not correspond exactly to poses included in the database. Moreover, some grasps are performed with high velocity, yielding frames with substantial motion blur.

It should be noted that the experiments were performed with different people, only changing parameters of color skin segmentation. The system is quite robust to different hand shapes. The sequences were recorded with the ARMAR humanoid head (see Figure 1). There is a decrease on performance when the hand occupies less than approximately 40x40 pixels.

Sample frames from the sequence are shown in Figure 8. The whole video with the results from the recognition system can be found at <http://www.csc.kth.se/~jrgn/handTracking264.mov>.

The main point of using continuity is to overcome ambiguity arising during a few frames, by taking into account past



Fig. 9. Segmentation error comparison. Column 1: query pose \mathbf{p}_t ; Column 2: segmentation mask; Column 3: estimated pose $\hat{\mathbf{p}}_t$; Column 4: estimated pose $\hat{\mathbf{p}}_t^{\text{uniform}}$.

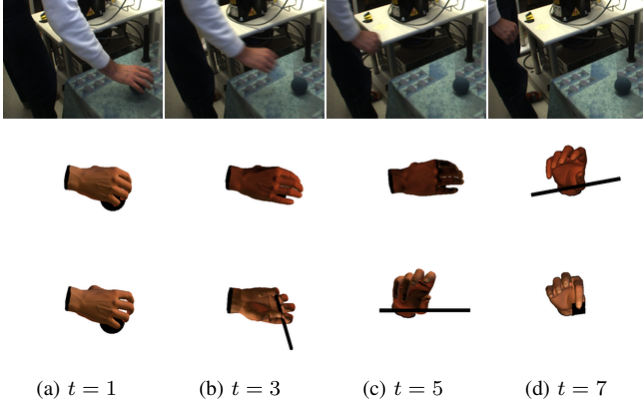


Fig. 10. Blurriness persistence. Row 1: query pose \mathbf{p}_t ; Row 2: estimated pose $\hat{\mathbf{p}}_t$; Row 3: estimated pose $\hat{\mathbf{p}}_t^{\text{uniform}}$.

estimations. As expected, Figure 8 shows that the estimates $\hat{\mathbf{p}}_t^{\text{uniform}}$ are less robust to temporal ambiguities in the mapping \mathcal{M} . Enforcing continuity over time also improves the robustness towards motion blur and bad segmentation, as shown in Figures 10, 9. However, if the different problems (blurriness, poor segmentation) persist over more than 5–10 frames, the continuity enforcement does not contribute to the same extent.

Finally, we got some early results on a humanoid LbD scenario for grasping purposes².

VII. CONCLUSIONS

A non-parametric method for 3D hand pose estimation over time from a monocular video sequence was presented. Experiments showed that the system estimates the hand pose in real time robustly against segmentation errors. It was also shown that enforcing continuity in the hand pose space improves the quality of the hand pose estimation. Initial results showed that the system can be used in a LbD scenario for humanoid imitation.

Future work along these lines includes improving the motion model; currently, a static model is implicitly assumed. We can for example include angular velocities in the pose state space, thus encapsulating velocity information in the database examples. Furthermore, we will update the database to represent poses of differently shaped hands under different illumination conditions. We also plan to investigate methods for mapping the human hand pose to a lower dimensional space suitable for the robot hand that is going to actuate the grasp after LbD.

²<http://www.csc.kth.se/~jrgn/GraspRecognitionDivx.avi>

VIII. ACKNOWLEDGMENTS

This project has been supported by the EU IST-FP6-IP PACO-PLUS, EU IST-FP7-IP GRASP and Swedish Foundation for Strategic Research through project CORS.

REFERENCES

- [1] S. Ekvall and D. Kragić, “Grasp recognition for programming by demonstration tasks,” in *IEEE International Conference on Robotics and Automation*, 2005, pp. 748–753.
- [2] L. Y. Chang, N. S. Pollard, T. M. Mitchell, and E. P. Xing, “Feature selection for grasp recognition from optical markers,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007.
- [3] A. Erol, G. N. Bebis, M. Nicolescu, R. D. Boyle, and X. Twombly, “A review on vision-based full DOF hand motion estimation,” in *Vision for Human-Computer Interaction*, 2005, pp. III: 75–75.
- [4] M. de la Gorce, N. Paragios, and D. J. Fleet, “Model-based hand tracking with texture, shading and self-occlusions,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–8.
- [5] V. Athitsos and S. Sclaroff, “Estimating 3D hand pose from a cluttered image,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2003, pp. 432–439.
- [6] H. Kjellström, J. Romero, and D. Kragić, “Visual recognition of grasps for human-to-robot mapping,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008.
- [7] M. J. Mataric, “Sensory-motor primitives as a basis for imitation: Linking perception to action and biology to robotics,” in *Imitation in Animals and Artifacts*, K. Dautenhahn and C. Nehaniv, Eds., 2000.
- [8] V. Pavlovic, R. Sharma, and T. S. Huang, “Visual interpretation of hand gestures for human-computer interaction: A review,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 677–695, 1997.
- [9] T. Starner and A. Pentland, “Visual recognition of american sign language using hidden markov models,” in *IEEE International Conference on Automatic Face and Gesture Recognition*, 1995.
- [10] A. A. Argyros and M. I. A. Lourakis, “Real time tracking of multiple skin-colored objects with a possibly moving camera,” in *European Conference on Computer Vision*, vol. 3, 2004, pp. 368–379.
- [11] E. Sudderth, M. I. Mandel, W. T. Freeman, and A. S. Willsky, “Visual hand tracking using non-parametric belief propagation,” in *IEEE Workshop on Generative Model Based Vision*, 2004.
- [12] M. T. Ciocarlie, S. T. Clanton, M. C. Spalding, and P. K. Allen, “Biomimetic grasp planning for cortical control of a robotic hand,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008, pp. 2271–2276.
- [13] A. Tsoli and O. C. Jenkins, “Neighborhood denoising for learning high-dimensional grasping manifolds,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008, pp. 3680–3685.
- [14] W. T. Freeman and M. Roth, “Oriental histograms for hand gesture recognition,” in *IEEE International Conference on Automatic Face and Gesture Recognition*, 1995.
- [15] G. Shakhnarovich, P. Viola, and T. Darrell, “Fast pose estimation with parameter sensitive hashing,” in *IEEE International Conference on Computer Vision*, vol. 2, 2003, pp. 750–757.
- [16] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2005, pp. I: 886–893.
- [17] M. Cutkosky, “On grasp choice, grasp models and the design of hands for manufacturing tasks,” *IEEE Transactions on Robotics and Automation*, vol. 5, no. 3, pp. 269–279, 1989.
- [18] N. Kamakura, M. Matsuo, H. Ishi, F. Mitsuboshi, and Y. Miura, “Patterns of static prehension in normal hands,” *Am J Occup Ther*, vol. 7, no. 34, pp. 437–45, 1980.
- [19] S. B. Kang and K. Ikeuchi, “Grasp recognition using the contact web,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1992, pp. 194–201.
- [20] A. Andoni and P. Indyk, “Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions,” *Communications of the ACM*, vol. 51, 2008.
- [21] R. Benetis, C. S. Jensen, G. Karciuskas, and S. Saltenis, “Nearest and reverse nearest neighbor queries for moving objects,” *The VLDB Journal*, vol. 15, no. 3, pp. 229–250, 2006.

Modeling and Evaluation of Human-to-Robot Mapping of Grasps

Javier Romero

Hedvig Kjellström

Danica Kragic

Abstract—We study the problem of human to robot grasp mapping as a basic building block of a learning by imitation system. The human hand posture, including both the grasp type and hand orientation, is first classified based on a single image and mapped to a specific robot hand. A metric for the evaluation based on the notion of virtual fingers is proposed. The first part of the experimental evaluation, performed in simulation, shows how the differences in the embodiment between human and robotic hand affect the grasp strategy. The second part, performed with a robotic system, demonstrates the feasibility of the proposed methodology in realistic applications.

I. INTRODUCTION

Programming service robots for new tasks puts significant requirements on the programming interface and the user. Programming by Demonstration (PbD) systems offer a great opportunity to unexperienced users for integrating complex tasks in a robotic system. However, representing, detecting and understanding human activities has been proven difficult and has been investigated closely during the past several years, [1], [2], [3], [4].

In the past, we have studied imitation of object manipulation tasks, using magnetic trackers, [4]. Although magnetic trackers and datagloves deliver exact values of hand joints, it is desirable that the user demonstrates tasks to the robot in a natural way; the use of gloves or other types of sensors may prevent a natural grasp. This motivates the use of systems based on visual input. In this paper, we concentrate on the use of our vision based grasp classification system presented in [5] for evaluation and execution of the grasp mapping on a robot. The contributions of the work presented here are:

- Based on a single image and the classification methodology [5], we propose a mapping strategy between a human and two robot hands of different kinematical properties. The mapping is performed according to the grasp taxonomy proposed in [6], shown in Fig. 1.
- The distinction between the grasp categories is made based on the preshape of the hand and also in terms of different strategies for approaching the objects.
- We propose a metric for evaluation of the mapping strategy and use it in the experimental evaluation.

We start with the state of the art description in Section II, followed by a short overview of grasp classification in Section III. Section IV describes the grasp mapping strategy and Section V presents the evaluation of the system. The paper is concluded in Section VI.

Authors are with the Centre for Autonomous Systems, CVAP, KTH, Stockholm, Sweden. {jrgn, hedvig, dani}@ckth.se. The work was supported by the EU projects GRASP, IST-FP7-IP-215821 and PACO-PLUS, FP6-2004-IST-4-27657 and the Swedish Foundation for Strategic Research.

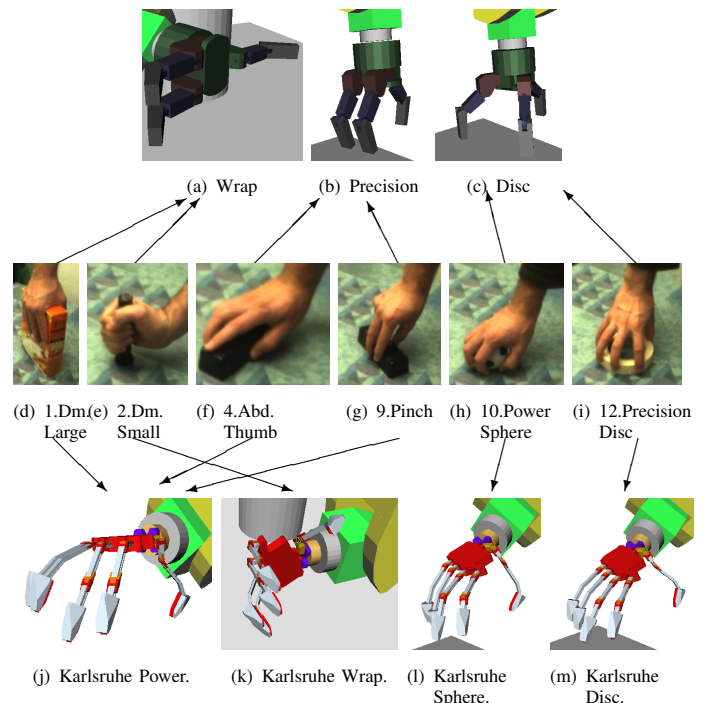


Fig. 1. The six grasps according to Cutkosky’s grasp taxonomy [6] considered in the classification, and the three grasps for a Barrett hand and Karlsruhe hand, with human-robot class mappings ((d,e)→(a),(f,g)→(b), (h,i)→(c)), (d,f,g)→(j), (e)→(k), (h)→(l), (i)→(m) shown. a) Barrett Wrap, b) Barrett Precision, c) Barrett Precision Disc d) Large Diameter grasp, 1. e) Small Diameter grasp, 2. f) Abducted Thumb grasp, 4. g) Pinch grasp, 9. h) Power Sphere grasp, 10. i) Precision Disc grasp, 12. j) Karlsruhe Power, k) Karlsruhe Wrap, l) Karlsruhe Sphere, m) Karlsruhe Disc

II. RELATED WORK

In the field of robotics, most of the object grasping systems are based on the a-priori object knowledge where either analytical or off-line methodology is used for grasp execution. In the work presented in this paper, we aim for learning grasps directly from human and mapping them on different robot hands. This relates to the work of [7] that classify objects based on their affordances (categories like “sidewall-graspable”), so the classification itself determines how to grasp the object. Also, as it has been showed by [8], the appropriate usage of grasps is not necessarily related just to the object’s shape. For example, the way to grasp a hammer is not the most natural or most stable for this object, but it is the best for the purpose a hammer is used.

According to [9], a grasp action involves two main functions well separated: the approach component (involving the arm muscles) and the grasp component (involving the hand muscles). Although it has been showed that these systems

are closely correlated, people focused mainly on one of the two subsystems. There are systems performing imitation of the arm [10] or, more generally, the upper-body [11]. The arm/upper-body imitation does not experience the self occlusion to the same extent as the hand does.

Our previous work [4] considered an HMM framework for recognition of grasping sequences using magnetic trackers and evaluated both the fingertip and the posture mapping. There are approaches that imitate the whole hand posture [12] or perform a simple mapping to a gripper based on two fingertips [13]. Mapping grasps to more complex hands is usually much more complicated. In [14] the concept of "virtual finger" is introduced: one or more real fingers acting in unison. Kang and Ikeuchi [15] use this concept as an intermediate step in their mapping procedure. Our approach integrates vision based mapping and the notion of virtual fingers for mapping human grasps to two robot hands: one of them resembles human kinematics (Karlsruhe hand) and one of them does not (Barrett hand). Thus, the robot here does not explore a range of approach vectors, but instead directly imitates the human approach vector, encoded in the hand position and orientation relative to the object.

III. VISION BASED GRASP CLASSIFICATION

The details of our methodology for visual recognition of grasps can be found in our previous work, [5]. We only summarize the most important aspects. The input to the grasp classification method is a single monocular image in which we first segment the hand. The classification method is non-parametric; grasp classification and hand orientation regression is formulated as a problem of finding the hand poses most similar to image example \mathbf{H} in a large database. Each database sample $\mathbf{H}_{i,j}^{\text{synth}}$, where i denotes grasp type and j denotes sample number, has associated with it a class label $y_{i,j} = i$ and a hand-vs-camera orientation $o_{i,j} = [\phi_j, \theta_j, \psi_j]$, i.e. the Euler angles from the camera coordinate system to a hand-centered coordinate system. To find the grasp class \hat{y} and orientation \hat{o} of an unknown grasp view \mathbf{H} acquired by the robot, a distance-weighted k -nearest neighbor (k NN) classification/regression procedure is used. First the set of k nearest neighbors to \mathbf{H} in terms of Euclidean distance between gradient orientation histograms obtained from the grasp images are retrieved from the database. From the found approximate nearest neighbors, the estimated class of \mathbf{H} is found as a distance-weighted selection of the most common class label among the k nearest neighbors, and the estimated orientation as a distance-weighted mean of the orientations of those samples among the k nearest neighbors for which $y_{i,j} = \hat{y}$.

IV. EXAMPLE-BASED MAPPING OF GRASP TO ROBOT

The estimated grasp class as well as hand and object orientation and position are used to instantiate a robot grasp strategy. The human-to-robot grasp mapping scheme is defined depending on the type of robot hand used. The Barrett hand is a three fingered, 4DOF robotic hand with an embodiment substantially different to a human hand.

Karlsruhe hand is a five fingered, 8DOF robotic hand with an embodiment similar to a human hand. The preshapes used for the hands are shown in Fig. 1. There are three preshapes for the Barrett hand:

- Barrett Wrap: used for grasps with a preshape with large aperture, like Large and Small Diameter grasps;
- Barrett Precision Grasp: for small aperture preshapes like the Pinch grasp and the Abducted Thumb (executed as a pinch grasp due to the hand kinematic constraints);
- Barrett Precision Disc: for circular objects.

There are four preshapes for the Karlsruhe hand:

- Karlsruhe Power preshape is applied for grasps with four parallel fingers and thumb opposed to them, like Large Diameter, Pinch and Abducted Thumb.
- Karlsruhe Wrap is applied for the Small Diameter where the thumb is not opposed to the rest of the fingers.
- There are two preshapes for round objects, Karlsruhe Sphere (for Power Sphere) and Karlsruhe Disc (for Precision Disc); the differences are in the pose of the thumb (more opposed in the Disc) and in how straight are the rest of the fingers (more bent in Power Sphere).

The grasp mapping is performed as shown in Algorithm 1. The hand orientation estimate $o_{h \rightarrow c}$, along with the hand position estimate $p_{h \rightarrow c}$ and the estimated position and orientation $o_{o \rightarrow c}, p_{o \rightarrow c}$ of the grasped object all relative to the camera are used to derive the estimated position and orientation of the human hand relative to the object $o_{h \rightarrow o}, p_{h \rightarrow o}$. The hand orientation relative to the table plane o_h is extracted from $o_{h \rightarrow c}$ and the orientation of the camera o_c , obtained through the robotic head kinematics. The estimation of object position and orientation is assumed perfect; this part of the system is not implemented, instead the ground truth is given in the simulations.

The system first decides which preshape to use based on the recognized grasp. Then, the approach vector is chosen. Two different ways of approaching the object are used, based on the orientation of the human hand; if the palm orientation o_h is similar to the one in Fig. 1(e) the object is approached from the side, otherwise it is approached from the top. Based on the estimated type of grasp, the system differentiates between volar and non-volar grasp, [15], i.e., whether there should be a contact between the palm and object or not. The original volar grasps are the Large Diameter, Small Diameter, Abducted Thumb and Power Sphere grasps, see Fig. 1. However, the limitations of the hands embodiments make impossible to use the palm in the Abducted Thumb and Power Sphere grasps. In a human Abducted Thumb grasp the palm adapts its shape to the object, and the abduction/adduction degrees of freedom of the fingers are used; the robotic hands studied here lack those degrees of freedom, so the Abducted Thumb is mapped to a Pinch Grasp. In the case of the Power Sphere, the robotic hands cannot apply a volar grasp due to the larger length of robotic fingers.

The volar grasping is performed in the following order:

- 1) The robot adopts the hand orientation and preshape

```

Data: Human Grasp  $G_h, p_{h \rightarrow o}, o_{h \rightarrow o}, o_h$ 
/* Robot Hand  $H \in \{Barrett, Karlsruhe\}$  */
/* Robotic Grasp  $G_r$  */
/* Approach Vector  $a$  */
/* Distance palm-fingertip  $\delta$  */
if  $H = Barrett$  then
| if  $G_h \in \{LargeDiameter, SmallDiameter\}$  then
| |  $G_r = BarrettWrap$ ;
| else if  $G_h \in \{Pinch, Abducted\}$  then
| |  $G_r = BarrettPrecision$ ;
| else if  $G_h \in \{PowerSpherical, PrecisionDisc\}$ 
| then
| |  $G_r = BarrettPrecisionDisc$ ;
else if  $H = Karlsruhe$  then
| if  $G_h \in \{LargeDiameter, Pinch, Abducted\}$  then
| |  $G_r = KarlsruhePower$ ;
| else if  $G_h = SmallDiameter$  then
| |  $G_r = KarlsruheWrap$ ;
| else if  $G_h = PowerSphere$  then
| |  $G_r = KarlsruhePowerSphere$ ;
| else if  $G_h = PrecisionDisc$  then
| |  $G_r = KarlsruhePrecisionDisc$ ;
if  $o_h = o_{side}$  then ; /* see Fig. 1(e) */
|  $a = a_s$ ; /* approach from side */
else
|  $a = a_v$ ; /* approach from top */
/* execute */
Set hand to preshape  $G_r$ ;
Set hand to orientation  $o_{h \rightarrow o}$ ;
if  $G_h \in \{LargeDiameter, SmallDiameter\}$  then ;
/* if volar */
| Approach following  $a$  towards  $p_{h \rightarrow o}$  until contact;
| while contact  $p_c$  out of palm do
| | Retreat;
| |  $p_{h \rightarrow o} = \alpha p_c + (1 - \alpha) p_{h \rightarrow o}$ ;
| | Approach following  $a$  towards  $p_{h \rightarrow o}$ ;
| |  $\alpha = \alpha^2$ ;
else ; /* if non-volar */
| Approach following  $a$  towards  $p_{h \rightarrow o} - \delta$ ;
Grasp;

```

Algorithm 1: Pseudo-code for grasp mapping.

corresponding to the estimated human grasp.

- 2) The robot hand approaches the object centroid until it detects contact on the palm sensor. After that, it closes the hand.

When the robot detects that the first contact did not occur in the palm, the trajectory is replanned. The new goal position for the hand is a weighted average between the detected contact p_c and the original goal position $p_{h \rightarrow o}$, as

explained in Algorithm 1. The non-volar grasps, which have no contact between the palm and the object, are originally the Pinch and Precision Disc grasps (see Fig. 1). Since there is no contact between the tactile sensor in the palm and the object, in our system the grasp is performed without any feedback. For this reason, the non-volar grasps depend heavily on the precision of the object position and orientation estimation. The difference between non-volar and volar strategies is the absence of the loop where the contact location is checked (see Algorithm 1).

V. EXPERIMENTAL RESULTS

We first evaluate our approach in the GraspIt! simulator and then demonstrate it in a real robotic scenario.

A. Simulated grasping with GraspIt

As stated, evaluating the performance of a grasp imitation system is not trivial. It cannot be based on grasp stability, and comparison between joint angles in robotic hands and human hand is not possible because of the differences in the embodiment. We have decided to compare the grasps using the concept of virtual fingers ([14]), computed based on the equations stated in [16]. As cited in Section II, a virtual finger is a group of real fingers (including the palm) that act in unison. So, in theory, the average position and orientation of the virtual finger contacts in the imitated grasp should be similar to the ones in the original grasps. However, as it will be discussed later, that is not always the case.

The virtual finger configuration tries to minimize two factors: the number of virtual fingers N , in order to achieve a compact representation, and the heterogeneity of the real fingers R_i conforming each virtual finger V_k , described as a cohesive index for virtual finger k , C_{V_k} . The cohesive index of each virtual finger is computed based on the degree of force coupling (cosine of the angle between the forces) between each two forces f_i, f_j applied with any of the fingers within a virtual finger:

$$D_c(i, j) = \frac{f_i \cdot f_j}{|f_i| \cdot |f_j|}, \quad m_{ij} = \frac{1 + D_c(i, j)}{2}$$

$$C_{V_k} = \prod_{\substack{i \in R_i, j \in R_j \\ R_i, j \in V_k}} m_{ij}^\xi, \quad \xi = \binom{F(V_k)}{2}^{-1}$$

where $F(V_k)$ is the number of fingers within V_k . For example, if all forces within a virtual finger k are parallel, $C_{V_k} = 1$. If any two forces belonging to the virtual finger are perpendicular, $C_{V_k} = 0$. So, in order to find the best configuration of virtual fingers, we maximize the cohesive indexes C_{V_k} trying to keep the number of virtual fingers small:

$$\begin{aligned} &\text{Maximize} && C_{eff} = \left(\frac{1}{N!} \prod_{i=1}^N C_{V_i} \right)^{\frac{1}{N}} \\ &\text{Subject to} && N \in 1, 2, 3, 4, 5, 6, \quad \bigcup_{i=1}^N V_i = R \\ &&& V_i \cap V_j = \emptyset, i \neq j, 1 \leq i, j \leq N \end{aligned}$$

So for every possible combination of real fingers R_i assignment to virtual fingers V_k , the coefficient C_{eff} is

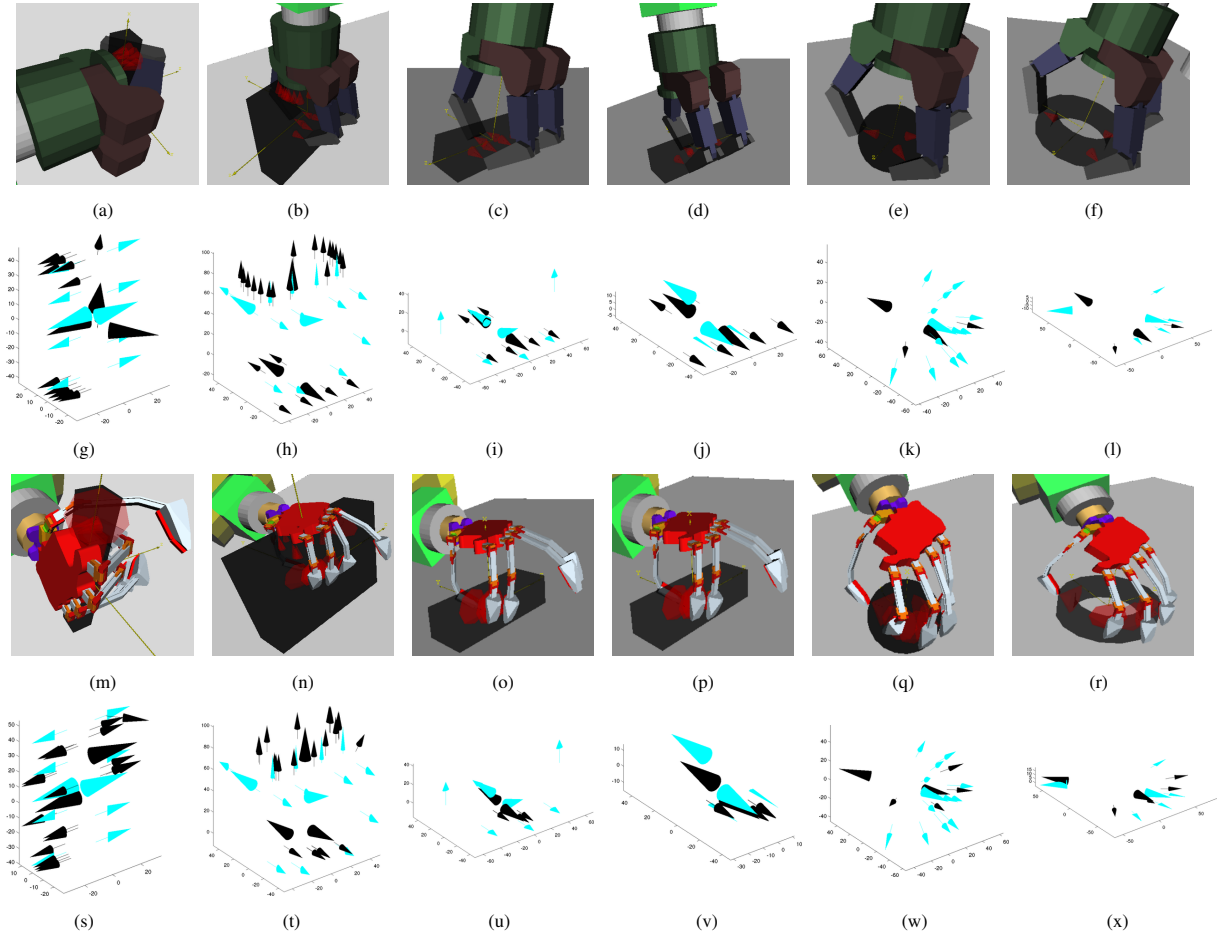


Fig. 2. First and third row shows the grasp execution in absence of errors for Barrett and Karlsruhe Hand. Second and forth row show a comparison between the contacts for Barrett(black)-Human(blue) hands and Karlsruhe(black)-Human(blue) hands. The big arrows show the average pose of the virtual fingers, \mathcal{P}_{V_k}

computed. The assignment with the highest C_{eff} is selected as the virtual finger representation of the grasp. The position and orientation of contacts is automatically extracted from the robotic simulator for the robot grasps, and it was tagged manually from images for the human grasp.

So far, all the real fingers R_i have been assigned to a virtual finger V_k . In order to compare the configuration of different hands we will define the contact pose (6d, including position and orientation) \mathcal{P}_{V_k} of a virtual V_k as the average of the contact poses \mathcal{P}_i within this virtual finger:

$$\mathcal{P}_{V_k} = \frac{1}{T(V_k)} \sum_{\substack{i \in R_i \\ R_i \in V_k}} \mathcal{P}_i$$

$$T(V_k) \equiv \text{number of contacts within } V_k$$

In the first experiment, perfect object pose estimation and perfect hand pose recognition is assumed. Fig. 2 represents the grasps and the contact comparison between the robotic hands (black) and the human hand (blue). The big arrows show \mathcal{P}_{V_k} , and the small arrows show all \mathcal{P}_i . It can be seen in this figure that the pose of the virtual fingers and even the number of them does not coincide always. For example, the Barrett hand has three virtual fingers for the

Small Diameter grasp, while Human and Karlsruhe hands have two (Fig. 2a,g,m,s). The reason for this mismatch is that Barrett fingers are longer than human and Karlsruhe fingers, so the object is touched by the last phalanx in the edges instead of the face. Another significant difference in the number of virtual fingers appears in the Power Sphere grasp (Fig. 2c,k,q,w). The human grasp has just one virtual finger, while the robotic hands have two. For the human, placing the thumb opposed to the rest of the fingers is uncomfortable. The big contact surface and therefore big friction between the hand and the ball allows him to place the fingers in a relatively unstable way. However, the contact surface between the robotic hands and the object is much smaller, so the thumb should be placed in opposition to the rest of the fingers. In contrast, in the Precision Disc grasp (Fig. 2f,r,l,x) the human needs to place the thumb in opposition to the rest of the fingertips due to the lower friction between the hand and the object. It is also interesting to reason about the results for the Large Diameter grasp (Fig. 2b,h,n,t). Apparently there is a big difference between the average virtual finger position, but the actual contacts look similar. The reason is that the human fingers have contacts in both the proximal and distal phalanges, while

the robot achieves a contact just in the distal "phalanges". Finally, it should be pointed that despite the impossibility of imitating properly the Abducted Thumb grasp, the position of the virtual fingers is quite accurate, with a deviation in orientation in one of them due to the two contacts from the human palm and index fingertip in the top of the object, inexistent in the robotic grasps.

We present a more concise view of the experiments in Fig. 3: it represents the average error in virtual fingers position and orientation (position and orientation component of \mathcal{P}_{V_k}) between the robotic hands (where Barrett error is represented in black and Karlsruhe in white) and the human hand. However, it should be noted that this measure is a lower bound of the error: in cases where the number of virtual fingers is different, this represents the average distance between the best matching virtual fingers. Sometimes this mismatch is known and natural (like the different number of virtual fingers in the Power Sphere grasp), but sometimes this means that one finger failed to touch the object. This happens mainly in the experiments with position error with the Karlsruhe hand, and will be mentioned later. For the case of perfect data (Fig. 3a for orientation, Fig. 3e for position) we can infer a number of conclusions: Karlsruhe hand performance in terms of orientation is better than the performance of Barrett hand; the performance in terms of position of the virtual fingers is similar; the biggest errors appear in the Large Diameter grasp, for the reasons stated before.

The next experiment tests the robustness with respect to the object position errors. We introduced an error of $\rho = 5cm$ in 6 different directions:

$$\begin{aligned} \hat{p}_o &= p_o + \rho[\cos \beta, \sin \beta, 0] \\ \beta &= \{0, \frac{\pi}{3}, 2\frac{\pi}{3}, 3\frac{\pi}{3}, 4\frac{\pi}{3}, 5\frac{\pi}{3}\} \end{aligned}$$

The difficulty of the problem should be noted: the size of the objects in their biggest axis is around 10cm, so the error is significant compared to their size. Another factor is the lack of any visual feedback. This experiment show us principally the importance of the feedback (tactile feedback in our case) in the presence of errors. The only grasps where tactile feedback was used are the Large and Small Diameter grasps. The reason for that is because those grasps are the only ones where we expect a first contact with the palm: this means that a first contact detected in any other finger suppose an error in the object pose detection that should be corrected. It can be seen that, in the presence of object pose errors, the error increases much more in the grasps without corrective movements (grasps 3,4,5 and 6) than in the ones with corrective movements (grasps 1 and 2). Another thing that can be inferred in Fig. 3b,f is that the Karlsruhe hand is more sensitive to the errors than the Barrett hand; the error increases more for Karlsruhe hand in presence of errors than for the Barrett hand. Actually the error for the Karlsruhe hand in non-corrected grasps is higher than the one showed, because the thumb usually fail to touch the

object, and therefore the thumb virtual finger not compared. There are principally two reason for this worse robustness to errors: first, the shorter length of the fingers; second, the palm configuration in the Karlsruhe hand. The shorter length of Karlsruhe fingers affect the non-volar grasps, as we can see in Fig. 4a,b. The small distance between the base of the thumb and the base of the rest of the fingers affects the volar grasps, that usually collide with the finger bases before touching the palm. However, this is mostly solved by the corrective movements.

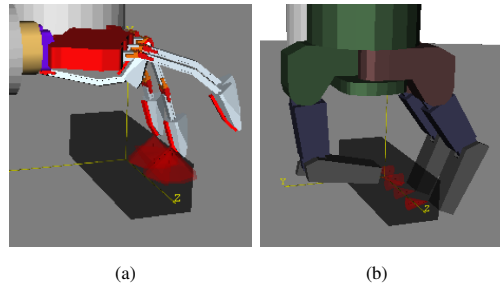


Fig. 4. Example of performance of a grasp without corrective movements

B. Real grasping with a KUKA arm

An image of the human grasp is captured and passed to the grasp recognition module [5], which returns the type of grasp and the position and orientation of the hand. Using the object pose, the grasp policy is selected and executed. The scenario, illumination and subject is different to the experiments in [5], but we get similar results in the classification. Large diameter, small diameter and abducted thumb are correctly classified most of the time, while pinch grasp, power sphere and precision disc grasp are sometimes confused with the power grasp. In terms of orientation, the typical error is around 15 degrees, which is acceptable in the execution of the grasp. The object position is given manually, with an error of ± 3 cm. The position error did not inflict on the grasp execution, except when performing Precision Disc grasp with a ball, which rolled when the hand was not centered over the ball. Fig. 5 shows the robot being shown four different grasps (Large Diameter, Abducted Thumb, Pinch and Precision Disc, respectively), mapping them and performing the corresponding grasp (Barrett Wrap, Barrett Precision, Barrett Precision and Barrett Precision Disc, respectively).

VI. CONCLUSIONS

We have presented a human-to-robot grasp mapping system based on a single image. We have proposed an evaluation metrics for assessing the quality of mapping. The approach was demonstrated both in simulation and in a real robot setup. The system presented here can be improved in several ways. The database of hand poses should include more objects of different sizes, and more advanced non-parametric regression methods could be employed for estimating grasp type and hand pose. Moreover, the addition of visual servoing would improve considerably the performance in grasps without tactile feedback (volar grasps) or grasps when the

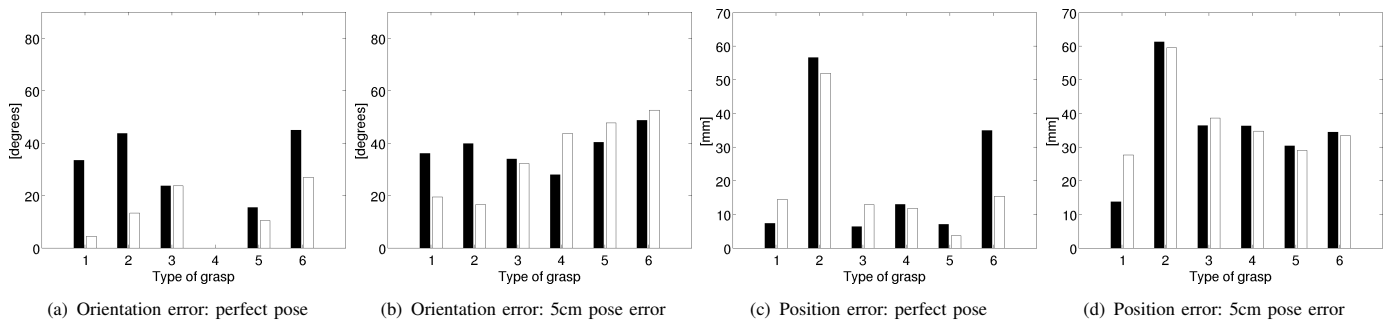


Fig. 3. Error in position (mm) and orientation (degrees) for each of the six grasps tested (Small Diameter, Large Diameter, Abducted Thumb, Pinch, Power Sphere, Precision Disc). Each column represents experiments with no error and error of 5cm.

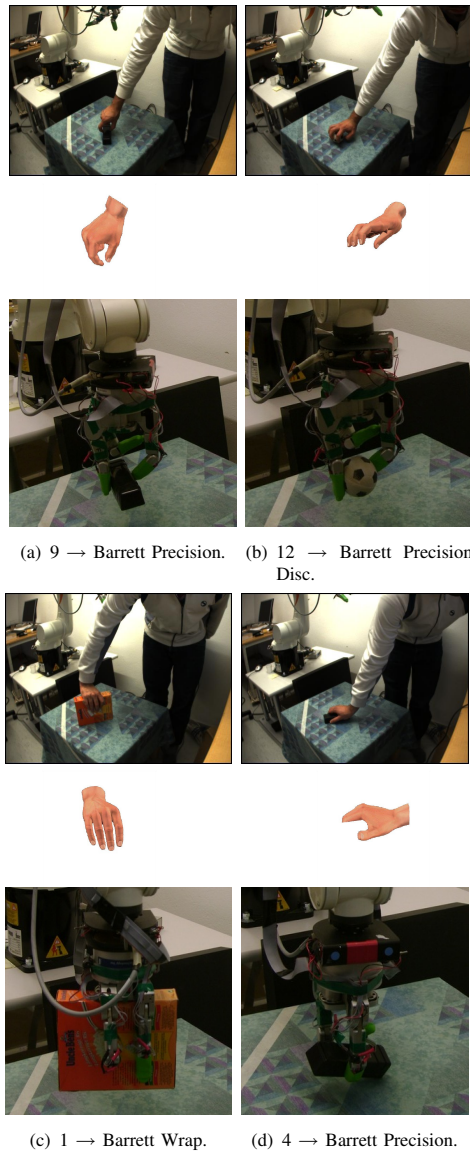


Fig. 5. Execution of grasps in a real robot environment: original images, nearest neighbors in the database and robot execution.

tactile feedback fail due to the limitations of the hand sensors. Finally, we will investigate the performance using a

humanoid robot hand.

REFERENCES

- [1] Y. Kuniyoshi, M. Inaba, and H. Inoue, "Learning by watching," *IEEE Transactions on Robotics and Automation*, vol. 10, no. 6, pp. 799–822, 1994.
- [2] S. Schaal, "Is imitation learning the route to humanoid robots?" *Trends in Cognitive Sciences*, vol. 3, no. 6, pp. 233–242, 1999.
- [3] K. Ogawara, S. Iba, H. Kimura, and K. Ikeuchi, "Recognition of human task by attention point analysis," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2000, pp. 2121–2126.
- [4] S. Ekvall and D. Kragić, "Grasp recognition for programming by demonstration tasks," in *IEEE International Conference on Robotics and Automation*, 2005, pp. 748–753.
- [5] H. Kjellström, J. Romero, and D. Kragić, "Visual recognition of grasps for human-to-robot mapping," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008.
- [6] M. Cutkosky, "On grasp choice, grasp models and the design of hands for manufacturing tasks," *IEEE Transactions on Robotics and Automation*, vol. 5, no. 3, pp. 269–279, 1989.
- [7] M. Stark, P. Lies, M. Zillich, J. Wyatt, and B. Schiele, "Functional object class detection based on learned affordance cues," in *Computer Vision Systems*, 2008.
- [8] A. M. Borghi, "Object concepts and action," in *The Grounding of Cognition: The role of perception and action in memory, language, and thinking*, D. Pecher and R. Zwaan, Eds. Cambridge University Press, 2005, part 2, pp. 8–34.
- [9] M. Jeannerod, "Intersegmental coordination during reaching at natural visual objects," *Attention & Performance*, vol. IX, 1981.
- [10] A. Billard, Y. Epars, S. Schaal, and G. Cheng, "Discovering imitation strategies through categorization of multi-dimensional data," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3, 2003, pp. 2398–2403.
- [11] P. Azad, A. Ude, T. Asfour, and R. Dillmann, "Stereo-based markerless human motion capture for humanoid robot systems," in *IEEE International Conference on Robotics and Automation*, 2007, pp. 3951–3956.
- [12] J. Rehg and T. Kanade, "Visual tracking of high dof articulated structures: An application to human hand tracking," in *European Conference on Computer Vision*, vol. 2, 1994, pp. 35–46.
- [13] J. Triesch, J. Wiegardt, E. Mael, and C. Malsburg, "Towards imitation learning of grasping movements by an autonomous robot," in *International Gesture Workshop*, 1999, pp. 73–84.
- [14] M. A. Arbib, T. Iberall, and D. M. Lyons, "Coordinated control programs for movements of the hand," in *Hand function and the neocortex. Experimental Brain Research Supplemental 10*, A. W. Goodwin and I. Darian-Smith, Eds., 1985.
- [15] S. B. Kang and K. Ikeuchi, "Toward automatic robot instruction from perception: Mapping human grasps to manipulator grasps," *IEEE Transactions on Robotics and Automation*, vol. 13, no. 1, pp. 81–95, 1997.
- [16] S. B. Kang and K. Ikeuchi, "A framework for recognizing grasps," in *CMU Robotics Institute*, 1991. [Online]. Available: <ftp://reports.adm.cs.cmu.edu/usr/anon/robotics/CMU-RI-TR-91-24.ps.Z>

Primitive Based Action Representation and Recognition

Sanmohan † Volker Krüger † Danica Kragic ‡ and Hedvig Kjellström ‡

† *Copenhagen Institute of Technology*

Computer Vision and Machine Intelligence Lab,

Latrupvang 15, 2750 Ballerup, Denmark

‡ *Royal Institute of Technology, Computational Vision and Active Perception lab,*

Centre for Autonomous Systems, S-100 44 Stockholm, Sweden

{san,vok}@cvmi.aau.dk, danik@nada.kth.se, hedvig@kth.se

Abstract

With the recent finding of mirror neurons there has been a growing interest in expressing actions as a combination meaningful subparts called primitives. Primitives could be thought of as an alphabet for the human actions. In this paper we investigate modeling and recognition of arm manipulation actions at different levels of complexity using primitives. Primitives are detected automatically in a sequential manner. Here, we assume no prior knowledge on primitives but look for correlating segments across various sequences. All actions are then modeled within a single HMMs whose structure is learned incrementally as new data is observed. We also generate an action grammar based on these primitives and thus link signals to symbols.

keywords: Primitive Detection, Imitation learning, high level event and activity understanding.

1 Introduction

An efficient method to transfer skills to a robot is very important to design a humanoid robot. Imitation learning is a promising approach to teach a robot new motor skills [1, 2, 3, 4, 5, 6, 7, 8, 9]. A standard approach in teaching tasks such as robot arm movements to a robot, is to store complete arm movement that are to be executed by the robot. For complex tasks an entire library of specific arm movement may be required. Humanoid robots, that are required to interact with humans need to have the ability to a) recognize human movements in order to react to them and/or learn to perform tasks from human demonstration and b) cope possibly with complex human environments by way of adapting their movements[1, 5, 6, 7, 8, 9].

One way for humanoid robots to recognize a human movement is to find in the library the movement model that explains the observed movement best. However, for large libraries this becomes very unrobust in terms of recognition rate and very ineffective in terms of computation time. Concerning the ability to adapt the movements, the robot is limited to choose a predefined movement from its library.

An alternative to storing a predefined and complete movement library is to break down the library movements into their common pieces and to represent the movements in a grammatical manner, with the common movement *primitives* as the grammar symbols. In linguistics [10] and computer vision [11] this has been proven to an effective approach for recognition, both in terms of recognition performance and computational efficiency. Similar results were recently shown in robotics [12] where Vicente *et al.* compared the use of a movement library with a more grammatical-like representation.

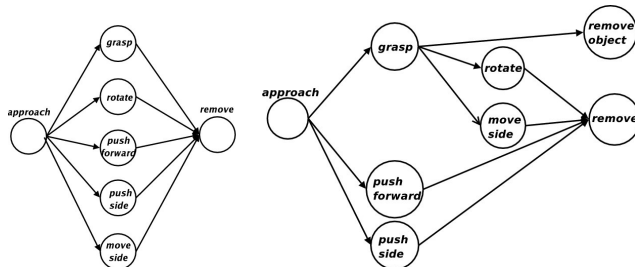


Figure 1: HMM models used in [12]. (left): Structure of HMM model I with actions as primitives; (right): Structure of HMM II with composite actions.

A grammatical representation of actions has the advantage of allowing to re-combine the grammatical symbols through planning strategies [13] and thus allowing the robot to adapt to new situations and even develop new movements based on new combinations of the available set of movement primitives, an issue that was also discussed in [12].

A common problem in using a grammatical representation is the definition of a proper set of primitives. All above mentioned works [10, 11, 12] rely on hand-selecting the primitives. In this paper, we present a systematic approach for finding the primitives for the robotics scenario presented in [12] automatically.

In Sec. 2 we give a summary of the work by Vicente *et al* [12] which could be considered as the starting point of our work. In [12], motion trajectories are segmented in a supervised manner and the resulting primitives are modeled with an hidden Markov model (HMM). The observed superiority of recognition rates [10, 11, 12] and the ability to model unknown actions with primitives motivates the search for an unsupervised method for trajectory segmentation and modeling. This is in detail explained in Sec. 3. We base our approach on HMMs, and we use a model merging technique to build a model and segment trajectories into primitives. In this way, we arrive at a single HMM which is similar to the one in the supervised learning scenario in [12]. We further learn a stochastic context free grammar for the primitives we have found.

2 Summary of Supervised Approach

Our work is an enhancement to the work done in [12]. Hence we start with a brief overview of [12] so that we could compare the works easily.

The results of [12] are a study of modeling and understanding of manipulation actions performed by humans on a table top scenario. Five actions are considered: a) pick up an object from a table, b) rotate an object on a table, c) push an object forward, d) push an object to the side, and e) move an object to the side by picking it up.

Each action is performed in 12 different conditions: Objects placed on two different heights and two different locations on the table, and the demonstrator stand in three different locations (0, 30, 60 degrees). All the actions are demonstrated by 10 different people.

Four sensors are attached to each person and their positions in 3D coordinates are measured. The sensors are located on: a) chest, b) back of hand, c) thumb, and d) index finger. The measurements from chest sensor is used to provide a reference to the demonstrator position while the sensor at the back of the hand is used as a reference for the thumb and index finger. The raw measurements are then preprocessed and the following 12 measurements are used for experiments:

- position of the hand relative to the chest
- position of the index finger and the thumb relative to the hand
- velocity of the hand

Primitives are manually extracted from the data and two different HMM structures are considered for modeling the actions which are shown in Fig. 1 and their results are compared. In the first model the primitives are *grasp(g)*, *rotate(r)*, *push forward(ps)*, *push side(ps)*, *move side(m)*, *approach* and *remove*. In the second model, the grasping part of *rotate* and *move side* primitives were considered as separate primitives. SVMs are used to recognize the primitives and the outcome of SVMs are then fed into the HMM used for modeling the actions. Using the outcome of SVMs as observations, the HMM parameters are learned through standard Baum-Welch algorithm.

The results of using Model I is presented in Tab.1 on the left. The entries on the diagonal shows the correct recognition rates. In this model individual primitives did not yield good results due to their high overlap.

In HMM model II , the common parts are kept as a separate primitive as shown in Fig. 1 on the right. This way of representation will give the primitives a semantic meaning as well. One new state, *remove with object*, was introduced to show that the end state is different from other cases . Each person is holding the object at the end only for the *grasp* action.

Tab. 1 on the right presents the recognition results by the HMM where the numbers on the diagonal give the correct recognition rate. The results of recognizing *grasp* and *move* have increased significantly.

2.1 Discussion

We could see the work of [12] as an extensive study on the modeling of the manipulation actions, which have the characteristic of being very similar to each other. The most important findings of their

HMM	pf	ps	r	g	m
pf	87.50	4.17	0.00	4.17	4.17
ps	8.33	48.33	2.50	3.33	37.5
r	0.83	2.5	95	1.67	0.00
g	5.83	10	9.17	52.5	22.5
m	1.67	24.17	4.17	2.5	67.5

HMM	pf	ps	r	g	m
pf	85	7.5	5	0.83	1.67
ps	9.17	47.5	4.17	2.5	36.67
r	0	0	92.5	0	7.5
g	4.17	7.5	10.83	72.5	5
m	1.67	10	6.67	0	81.67

Table 1: Recognition results from [12] for primitives using different HMMs .(Left)Confusion matrix for the recognition rates using HMM model I. (Right) Confusion matrix for the recognition rates using HMM Model II. Rows represent predicted class and columns represent actual class. Correct results are given on the diagonal.

experiments could be stated as: a) sequences of simple semantic primitives can be used in describing actions, and b) actions learned as sequences of primitives from other demonstrators can be combined with knowledge of personal primitives to recognize new actions.

3 Automatic segmentation of primitives

From the discussions above we can see that an efficient model could be made if we have the primitives at hand. Thus, it is desired to have a mechanism to detect the primitives automatically from action sequences. We note from the previous section that we had two types of primitives: primitives that were unique to an action and primitives that were common to more than one action. Thus our hypothesis is that if we segment action sequences into parts that are common across more than one action and parts that are unique to each of the actions, we will arrive at a set of action primitives that can be used for recognition as discussed in the previous section.

We define two sets of primitives. One set contains parts that are unique to one *type* of action and another set that contains parts that are common to more than one *type* of action. Two sequences are of the same *type* if they do not differ significantly, e.g., two different walking paths. Hence we attempt to segment sequences into parts that are common across sequences types and parts that are not shared. Then, each sequence will be a combination of these segments. We also want to generate rules that govern the interaction among the primitives. Keeping this in mind we state our objectives as:

1. Let $\mathcal{L} = \{X_1, X_2, \dots, X_m\}$ be a set of data sequences where each X_i is of the form $x_1^i x_2^i \dots, x_{T_i}^i$ and $x_j^i \in \mathbb{R}^n$. Let these observations be generated from a finite set of sources (or states) $\mathcal{S} = \{s_1, s_2, \dots, s_r\}$. Let $S_i = s_1^i s_2^i \dots, s_{T_i}^i$ be the state sequence associated with X_i . Find a partition \mathcal{S}' of the set of states \mathcal{S} where $\mathcal{S}' = \mathcal{A} \cup \mathcal{B}$ such that $\mathcal{A} = \{a_1, a_2, \dots, a_k\}$ and $\mathcal{B} = \{b_1, b_2, \dots, b_l\}$ are sets of state subsequences of X_i 's and each of the a_i 's appear in more than one state sequence and each of the b_j 's appear in exactly one of the state sequence. The set \mathcal{A} corresponds to common

actions and the set \mathcal{B} correspond to unique parts.

2. Generate a grammar with elements of \mathcal{S}' as symbols which will generate primitive sequences that match with the data sequences.

3.1 Outline of our Approach

We approach the problem by building a model that will generate the first data sequence that we encounter and check if the upcoming data sequences could have been generated from the constructed model. If not, modify the model to accommodate the newly observed data sequence. We continue this until we are able to create a single model that is capable of generating all the data sequences. In our case, we make a single hidden Markov model that will generate all the data sequences (explained in Sec. 3.2-3.2.3). Then by examining the sequence of states the observation sequences are going through, the common states are identified. States(or sequence of states) common to sequences are separated out to form \mathcal{A} and the remaining contiguous states make the set \mathcal{B} which were defined in Sec. 3. Creation of the sets \mathcal{A} and \mathcal{B} is explained in Sec. 3.3.

3.2 Modeling the Observation Sequences.

3.2.1 Modeling the First Sequence

Let X_1 be the first sequence with data points $x_1^1 x_2^1 \cdots x_{T_1}^1$. Since we have just one data sequence to start with, we generate a few more spurious sequences of the same type by adding Gaussian noise to it. Then we choose (μ_i^1, σ_i^1) , $i = 1, 2, \dots, k_1$ so that parts of the data sequence are from $\mathcal{N}(\mu_i^1, \Sigma_i^1)$ in that order. The value of k_1 is such that $\mathcal{N}(\mu_i^1, \Sigma_i^1)$, $i = 1, 2, \dots, k_1$ will cover the whole data. This value is not chosen before hand and varies with the variation and length of the data.

The next step is to make an HMM $\lambda_1 = (A_1, B_1, \pi_1)$ with k_1 states where k_1 is the number of Gaussians needed to cover X_1 . We let A_1 to be a left-right transition matrix and $B_{1_j}(x) = \mathcal{N}(x, \mu_j^1, \Sigma_j^1)$. All the states at this stage get a label 1 to indicate that they are part of sequence type 1. We require this information to link final primitives with different types of sequences and also for generating a grammar for primitives.

3.2.2 Modeling the rest of the data

Let $n - 1$ be the number of types of data sequences we have seen so far. Let X_c be the next data sequence to be processed. Calculate $P(X_c|\lambda_M)$ where λ_M is the current model at hand. If we get a high value for $P(X_c|\lambda_M)$ it indicates that λ_M models sequences of type X_c well, and so we proceed to the next data sequence. A low value for $P(X_c|\lambda_M)$ indicates that the current model is not good enough to model the data sequences of type X_c and hence we make a new HMM λ_c for X_c as described in Sec. 3.2.1. The newly constructed HMM λ_c will be used to modify λ_M so that the updated λ_M will be able to generate data sequences of type X_c . The modification procedure of λ_M using λ_c is described in Sec.

3.2.3. We increase the number of types of data sequences by one at this stage. All the states in X_c will be labeled n .

We might get a high value for $P(X_k|\lambda_M)$ for a new data sequence which has no unique part of its own but is part of several different types of data sequences we have seen so far. We resolve this by making use of the state labeling we have performed during the modeling. Whenever we get a high value for $P(L_k|\lambda_M)$ we look at the Viterbi path of the data sequence and examine the labels of the state sequence. If it is a new type then there will be two states whose labels have empty intersection. In that case we increase the number of types of data sequences by one and append the new type number to each of the states it is passing through.

3.2.3 Merging of similar states

This section explains the most important part of our method: modifying the existing model to generate a newly observed type of data. We do this by adding new states or by modifying existing states.

Suppose we want to merge λ_c into λ_M where λ_M is the current model so that $P(X_k|\lambda_M)$ is high if $P(X_k|\lambda_c)$. We do this by adding states to λ_M from λ_c or by merging states of λ_M with states of λ_c . Let $S_c = \{1, 2, \dots, c\}$ and $S_M = \{1, 2, \dots, M\}$ be the set of states of λ_c and λ_M respectively. Then the state set of the modified λ_M will be $S_M \cup D_1$ where $D_1 \subseteq S_c$. Each of the states i in λ_c affects λ_M in one of the following ways:

1. If $d(i, j) < \theta$, for some $j \in \{1, 2, \dots, M\}$, then i and j will be merged into a single state. Here d is a distance measure and θ is a threshold value. The output probability distribution associated with state j in λ_M is modified to be a combination of the existing distribution and $b_{k_i}(x)$. Thus $b_{Mj}(x)$ is a mixture of Gaussians. We append n to the label of the state j in λ_M . All transitions to state i in λ_c are redirected to state j in λ_M and all transitions from state i in λ_c will now be from state j in λ_M .
2. If $d(i, j) > \theta$, $\forall j$, a new state is added to λ_M . i.e. $i \in D_1$. Let i be the r^{th} state to be added from λ_c . Then, i will become the $(M+r)^{th}$ state of λ_M . The output probability distribution associated with this new state in λ_M will be the same as it was in λ_c . Hence $b_{M+r}(x) = \mathcal{N}(x, \mu_i, \Sigma_i)$. Initial and transition probabilities of λ_M are adjusted to accommodate this new state. The newly added state will keep its label n .

We use Kullback-Leibler Divergence to calculate the distance between states. The K-L divergence from $\mathcal{N}(x, \mu_0, \Sigma_0)$ to $\mathcal{N}(x, \mu_1, \Sigma_1)$ has a closed form solution given by :

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

Here n is the dimension of the space spanned by the random variable x .

Now we elaborate more on the addition and merging of states into the combined model. Our aim is to make the new model compatible with the newly observed type of data sequences. Since the states are

probability distributions, if we see that two probability distributions corresponding to different states are very close we do not need to keep them apart. Keeping these two states together will help us to model the observations generated from two distributions by a single one. We use (1) to compute the similarity between two states. We can observe that (1) will not handle mixture of Gaussians. We still use this equation to evaluate component wise distances in mixtures and check if any of the components are close to the distribution we are testing. We justify this criteria since our aim is to find out if a new state is to be embedded into another state or not.

3.3 Finding Primitives

Primitive searching starts when we have processed all the available data sequences. Now using Viterbi algorithm on the final merged model λ_M , the hidden states associated with each of the sequences are generated. Let P_1, P_2, \dots, P_r be different Viterbi paths at this stage. Since we want the common states that are contiguous across state sequences, it is similar to finding the longest common substring(LCS) problem. We take all paths with non-empty intersection and find the largest common substring a_k for them. Then, a_k is added to \mathcal{A} and is replaced with an empty string in all the occurrences of a_k in P_i , $i = 1, 2, \dots, r$. We continue to look for largest common substrings until we get an empty string as the common substring for any two paths. Thus, we end up with new paths P'_1, P'_2, \dots, P'_r where each P'_i consists of one or more segments with empty string as the separator¹. These remaining segments in each P'_i are unique to P_i . Each of them are also primitives and form the members of the set \mathcal{B} . Our objective was to find these two sets \mathcal{A} and \mathcal{B} as was stated in Sec. 3.

We also note that the computational complexity of calculating the longest common subsequence between two sequences of length T_i and T_j is $O(T_i + T_j)$. Hence primitive finding is solvable in linear time.

3.4 Generating the grammar for primitives

Let $\mathcal{S}' = \{c_1, c_2, \dots, c_p\}$ be the set of primitives available to us. We wish to generate rules of the form $P(c_i \rightarrow c_j)$ which will give the likelihood of occurrence of the primitive c_j followed by primitive c_i . We do this by constructing a directed graph G which encodes the relations between the primitives. Using G we will derive a formal grammar for the elements in \mathcal{S}' .

Let n be the number of types of data that we have processed. Then, each of the states in our final HMM λ_M will have labels from a subset of $\{1, 2, \dots, n\}$, see Fig. 2. By way of definition each of the states that belong to a primitive c_i will have the same label set l^{c_i} . Let $\mathcal{L} = \{l_1, l_2, \dots, l_p\}$ $p \geq n$ be the set of different type of labels received by the primitives. Let $G = (V, E)$ be a directed graph where $V = \mathcal{S}'$ and $e_{ij} = (c_i, c_j) \in E$ if there is a path $P_k = \dots c_i c_j \dots$ for some k . We have given the directed graph constructed for our test data described in Sec. 3.5.2 in Fig. 2.

¹The segmentation is caused by the gaps produced by the removal of elements of \mathcal{A} .

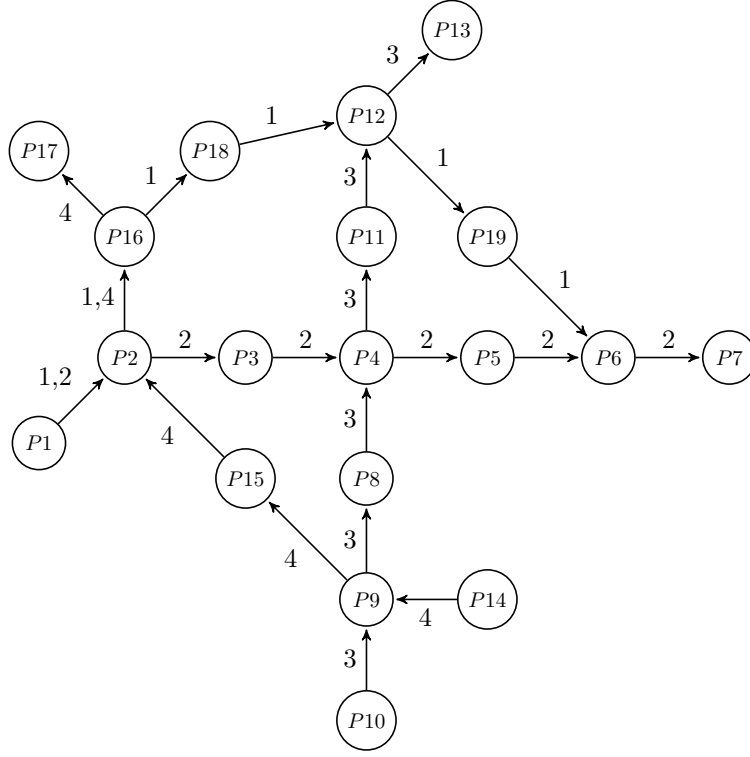


Figure 2: Directed graph for finding the grammar. This is the primitive graph for the data described in Sec. 3.5.2

We proceed to derive a Stochastic Context Free Grammar(SCFG) from the directed graph G we have constructed. Let $N = S'$ be the set of terminals. To each vertex c_i with an outgoing edge with label $l^{e_{ij}}$, associate a corresponding non-terminal $A_{c_i}^{l^{e_{ij}}}$. Let $\mathcal{N} = S \cup \{A_{c_i}^{l^{e_{ij}}}\}$ be the set of all non-terminals where S is the start symbol. For each primitive c_i that occurs at the start of a sequence and connecting to c_j define the rule

$$S \longrightarrow c_i A_{c_j}^{l^{e_{ij}}} . \quad (2)$$

To each of the internal nodes c_j with an incoming edge e_{ij} connecting from c_i and an outgoing edge e_{jk} connecting to c_k define the rule

$$A_{c_i}^{l^{e_{ij}} \cap l^{e_{jk}}} \longrightarrow c_j A_{c_k}^{l^{e_{jk}}} . \quad (3)$$

For each leaf node c_j with an incoming edge e_{ij} connecting from c_i and no outgoing edge define the rule

$$A_{c_j}^{l^{e_{ij}}} \longrightarrow \epsilon . \quad (4)$$

The symbol ϵ denotes an empty string. We assign equal probabilities to each of the expansions of a nonterminal symbol except for the expansion to an empty string which occurs with probability 1. Thus

$$P(A_{c_i}^{l^{e_{ij}}} \longrightarrow c_j A_{c_j}^{l^{e_{jk}}}) = \frac{1}{|c_i^{(o)}|} \text{ if } |c_i^{(o)}| > 0 . \quad (5)$$

$$P(A_{c_i}^{l^{e_{ij}}} \longrightarrow \epsilon) = 1 \text{ if } |c_i^{(o)}| = 0 . \quad (6)$$

where $|c_i^{(o)}|$ represents the number of outgoing edges from c_i and $l_{mn} = l^{c_m} \cap l^{c_n}$. Let \mathcal{R} be the collection of all rules given in (2), (3), and (4). For each $r \in \mathcal{R}$ associate a probability $P(r)$ using (5) and (6). Then $(\mathcal{N}, \mathcal{S}', S, \mathcal{R}, P(\cdot))$ is the stochastic grammar that models our primitives.

One might wonder why the HMM λ_M is not enough to describe the grammatical structure of the observations and why the SCFG is necessary. The HMM λ_M would have been sufficient for a single observation type. However for several observation types as in final λ_M , regular grammars, as modeled by HMMs are usually too limited to model the different observation types so that different observation types can be confused.

3.5 Experiments

We have run four experiments: In the first experiment we have used a synthetic data set with two types of sequences. The second experiment is motivated by the surveillance scenario of Stauffer and Grimson [14] and shows a complex set of paths as found outside our building. The third experiment is motivated by the work of Vincente and Kragic [12] on the recognition of human arm movements. In the fourth experiment we learn the movement primitives for a chess game.

3.5.1 Testing on Simulated Data

We illustrate the result of testing our method on a set of two sequences generated with mouse clicks. We have selected 2 simple sequences to illustrate the whole process. The two sequences share the initial portion as shown in Fig. 3(a). There is an intuitive segmentation with 3 parts for these two sequences: one segment containing the shared part and two separate segments for the unique parts. Our method extracts exactly these segments. The whole process is illustrated in Fig. 3. Sequence 1 with additional sequences generated by noise addition is shown in Fig. 3(b). Fig. 3(c) shows the result of covering these sequences with Gaussians. Covering of sequence 2 along with the first one is shown in Fig. 3(d). The sequences require 8 and 7 states respectively for covering. Hence the resulting individual HMMs will have 8 and 7 states respectively, see Fig. 3(e). Then the distances between the states of HMM1 and HMM2 are computed(Fig. 3(f)). Rows represent states of sequence 2 and columns represent states of sequence 1. One can notice the low values for the first four elements in the diagonal. Thus we have 4 pairs to merge: S_{21} with S_{11} , S_{22} with S_{12} , S_{23} with S_{13} and S_{24} with S_{14} . Merging is performed sequentially as shown in Fig. 3(g)- Fig. 3(j). When the model merging took place, the overlapping states were merged into one. The final HMM structure is shown in Fig. 3(j). The state sequences for the observed sequences are shown in Fig. 3(k)(Multiple occurrences are removed). Primitive segmentation will give us three primitives $p1, p2$ and $p3$, Fig. 3(l). Using the primitives, we can write the two sequences as primitive sequences: $(p1, p2)$ and $(p1, p3)$. Primitive tree in Fig. 3(m) shows the structure of primitives observed in the data. Using the primitive tree, a Stochastic Context Free Grammar is extracted by using the method illustrated in Sec. 3.4 and is shown in Fig. 3(n). The numbers in the brackets represent the probability of choosing the corresponding derivation. Finally the segmentation

of original data using primitives is shown in Fig. 3(o).

3.5.2 2D-Trajectory Data

The second experiment was done on a surveillance-type data inspired by [14]. The paths represent typical walking paths outside of our building. In this data there are four different types of trajectories with heavy overlap, see Fig. 4(left). We can also observe that the data is quite noisy. Fig. 4(right) shows the result of covering with Gaussians. The result of primitive segmentation is shown in Fig. 5. Different primitives are colored differently and we have named the primitives with different letters. The detected common primitives are the junctions where different trajectories intersect. As one can see, our approach results in primitives that coincide roughly with our intuition. Furthermore, our approach is very robust even with such noisy observations and lot of overlaps.

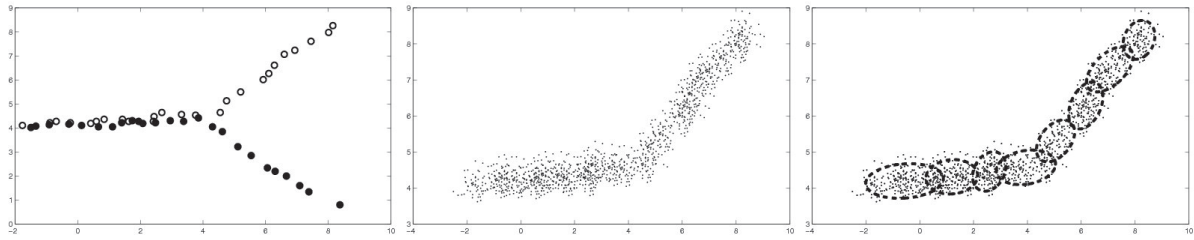
It should also be noted at this point that this kind of merging will not make the intersection arbitrarily large. Merging is done only when there is a good overlap. Also for each new type of sequences, there cannot be more than one Gaussian that gets merged into the same state.

3.5.3 Hand gesture data

We have tested our approach on the dataset described in Sec. 2 without annotation. Thus we use only the trajectory information for the sensors attached to the hand. The input to our system is the raw data from the sensors. We do not use the transformation of data described in Sec. 2. The original data is available on line [15]. We expect to extract a set of primitives so that each of these sequences can be expressed as a combination of these primitives. Since each of these sequences started and ended at the same position, we expect the primitives that represent the starting and end positions of actions will be the same across all the actions.

By applying the techniques described in Sec. 3 to the hand gesture data, we ended up with 9 primitives. The temporal order of primitives for actions for different actions are shown in Fig. 8. One can compare this with Fig. 1 and see that they are very closely related. For an easy comparison we plot the result of converting a grasp action sequence into a sequence of extracted primitives along with ground truth data in Fig. 6. The ground truth was obtained by looking at each sequences and manually segmenting them. This particular sequence had 119 points in it. In the ground truth, *Reach* extends from $t=1$ to $t=42$, *Grasp* extends from $t=43$ to $t=52$ and *Retrive* extends from $t=53$ to $t=119$. In our segmentation P_1 and P_2 combined extends from $t=1$ to $t=44$, P_3 extends from $t=45$ to $t=61$ and P_4 extends from $t=62$ to $t=119$. Thus we can infer from the figures Fig. 8 and Fig. 6 that P_3 and P_2 together constitute *approach* primitive, P_5 refers to *grasp* primitive and P_6 corresponds to *remove* primitive. Similar comparison could be made with other actions using the comparison diagram given in Fig. 7.

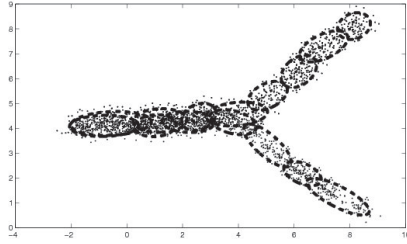
Using these primitives, an SCFG was built as described in Sec. 3.4. This grammar is used as an input to the Natural Language Toolkit (NLTK, <http://nltk.sourceforge.net>) which is used to parse the sequence of primitives. This grammar is used to test the validity of the primitive sequence for an



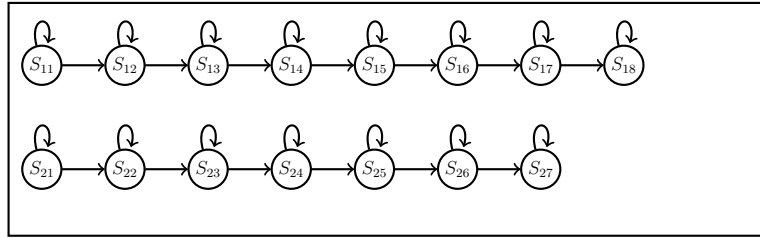
(a) Original Data

(b) Data with additional sequences

(c) Data Covered with Gaussians



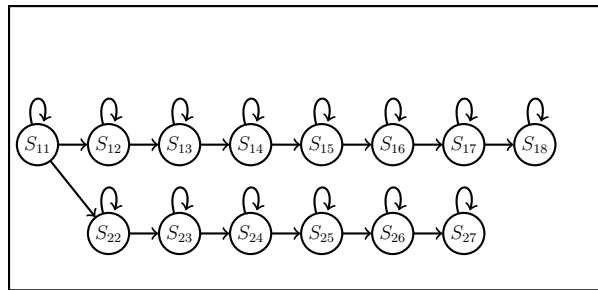
(d) Covering of sequence 2



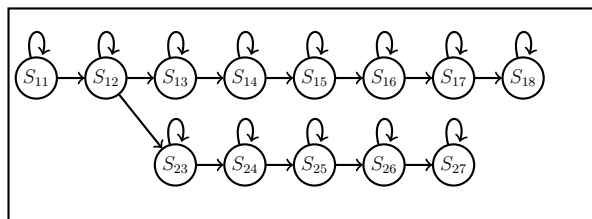
(e) HMMs for the data

0.17	6.2	34.0	29.0	85.0	155.0	155.0	322.0
5.5	0.47	9.5	20.0	56.0	111.0	133.0	288.0
11.0	2.0	1.0	5.9	28.0	72.0	100.0	222.0
34.0	15.0	6.1	1.2	13.0	46.0	88.0	188.0
88.0	47.0	32.0	11.0	24.0	60.0	122.0	233.0
133.0	93.0	91.0	52.0	77.0	111.0	200.0	300.0
122.0	111.0	133.0	77.0	122.0	155.0	255.0	344.0

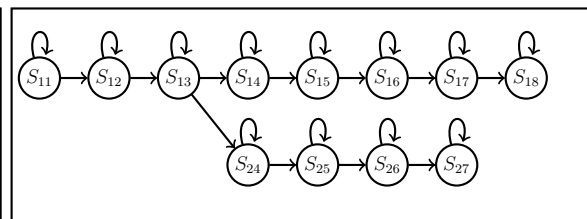
(f) Distance between states. Rows and columns represent states of HMM2 and HMM1 respectively



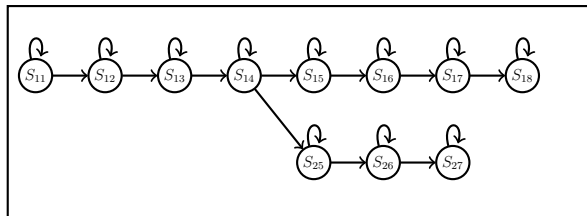
(g) Merging step 1. S_{21} is merged with S_{11} .



(h) Merging step 2. S_{22} is merged with S_{12} .



(i) Merging step 3. S_{23} is merged with S_{13}



(j) Merging step 4. S_{24} is merged with S_{14}

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 1 & 2 & 3 & 4 & 9 & 10 & 11 \end{pmatrix}$$

(k) State sequences of sequences

Figure 3: Illustration of the complete process with simulated data. Data was generated with mouse clicks.

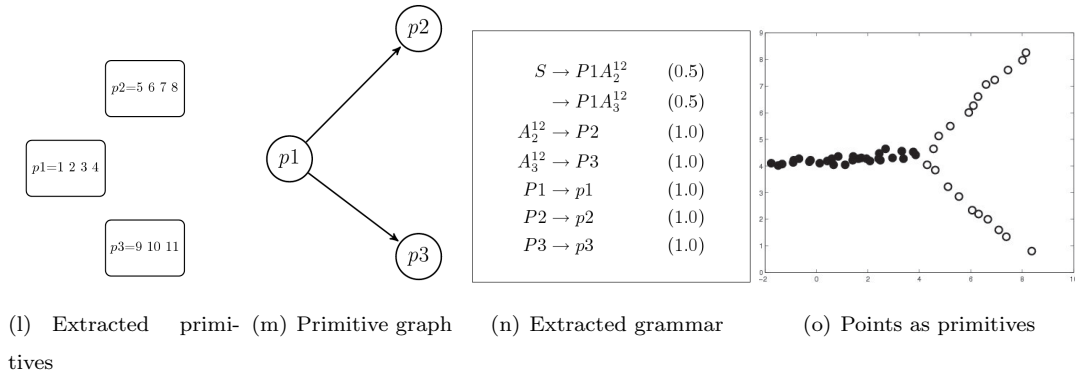


Figure 3: Illustration of the complete process with simulated data. Data was generated with mouse clicks.

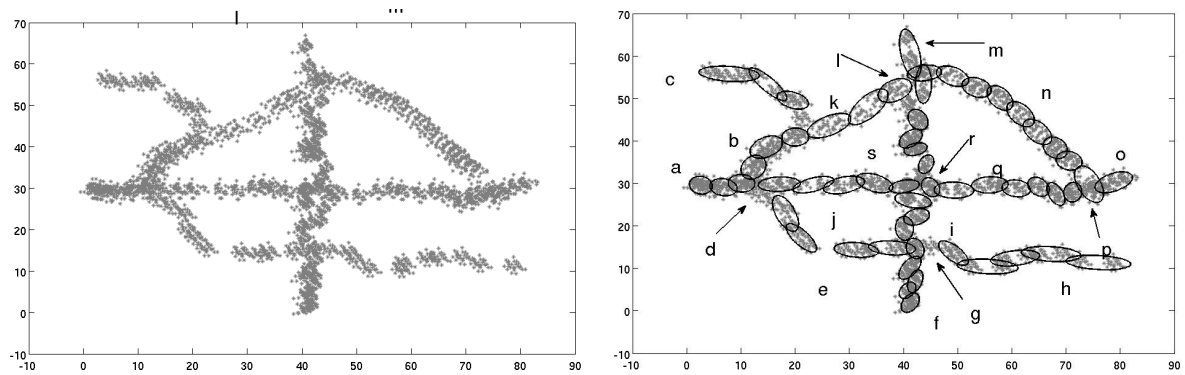


Figure 4: (Left) Trajectories from tracking data. Each type is colored differently. Values along the axes represent pixels. Only a part of the whole data is shown. (Right) The results of the covering procedure with the Gaussian mixtures. The numbers shown are the state numbers in the final model. Merged states are not shown. Hence some data points might appear to be unassigned.

unknown sequence. It can also be used to predict the future observation of a partially observed sequence.

Results of primitive segmentation for *push sideways*, *push forward*, *move*, and *grasp* actions are shown in the tables 2 and 3. The numbers given in the tables represent the primitive numbers shown in Fig. 8. The sequences that are identified correctly have a white background and the sequences that are not classified correctly have light gray background. We can see that all the correctly identified sequences start and end with the same primitive as expected. In Tab:3 on the right, Person 1 and Person 4 are marked with a dark color to indicate that they differ in end and start primitive respectively from the correct primitive sequence. This might be due to the variation in the starting and end position in the sequence. We could still see that the primitive sequence is correct for them.

Person	Push Aside				
Person 1	3	2	9	4	1
Person 2	3	5	8	4	1
Person 3	3	5	8	4	1
Person 4	3	5	8	4	1
Person 5	3	5	8	4	1
Person 6	3	5	8	4	1
Person 7	3	5	8	4	1
Person 8	3	5	8	4	1
Person 9	3	2	9	4	1
Person 10	3	2	9	4	1

Person	Push Forward				
Person 1	3	5	7		1
Person 2	3	5	7		1
Person 3	3	5	7		1
Person 4	3	5	7		1
Person 5	3	5	7		1
Person 6	3	5	8	4	1
Person 7	3	5	7		1
Person 8	3	5	7		1
Person 9	3	5	8	4	1
Person 10	3	5	8	4	1

Table 2: Primitive segmentation and recognition results for Push aside and Push Forward action. Sequences that are identified incorrectly are marked in light gray.

Person	Move				
Person 1	3	2	9	4	1
Person 2	3	5	8	4	1
Person 3	3	2	9	4	1
Person 4	3	2	9	4	1
Person 5	3	2	9	4	1
Person 6	3	5	8	4	1
Person 7	3	2	9	4	1
Person 8	3	2	9	4	1
Person 9	3	2	9	4	1
Person 10	3	2	9	4	1

Person	Grasp				
Person 1	3	2	6		
Person 2	3	2	6		1
Person 3	3	5	7	6	1
Person 4		2	6		1
Person 5	3	2	6		1
Person 6	3	2	6		1
Person 7	3	2	9	4	1
Person 8	3	2	6		1
Person 9	3	2	6	7	1
Person 10	3	2	6		1

Table 3: Primitive segmentation and recognition results for Move Object and Grasp actions. Sequences that are identified incorrectly are marked in light gray.

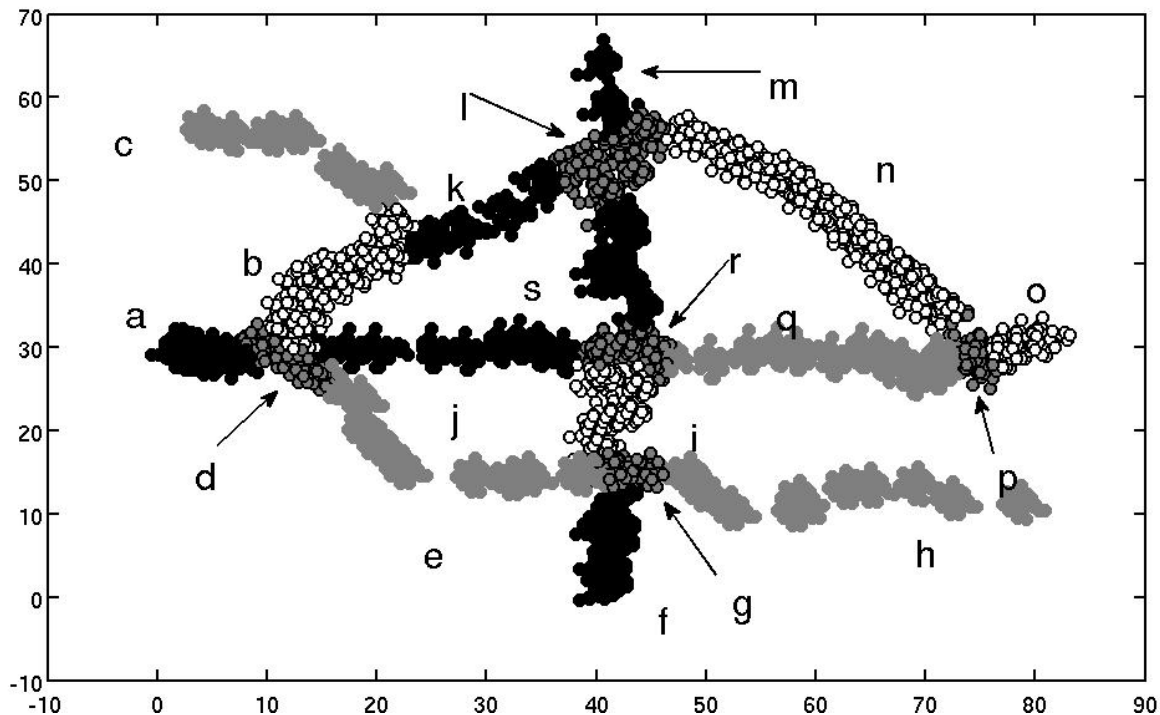


Figure 5: This figure shows the detected primitives. Each primitive is denoted by a letter.

3.6 Chess movements data

To further illustrate the application of our algorithm, we have tested our algorithm in a chess movements learning scenario. The aim of this experiment is to learn the different type of movements from the trajectory data. An object was placed on a chess board and was subjected to movements similar to that of chess pieces. We have recorded horizontal and vertical movements for the rook and queen at different lengths and the L-shape movements by the knight and diagonal movements for bishop. Some sample tracks are shown in Fig. 9. Only 8 out of the 12 knight moves were considered. In the collected dataset, we do not know how many types of movements are allowed, and what are the types of movements in there. Each of the recorded sequences were fed to the model and the primitives were extracted and the resulting structure is shown in Fig. 10. In this figure P1 represent moving one square to the right and P1-P2 represent moving two square to the right etc. Each path with a green primitive followed by a blue primitive represents an L-shaped movement for the knight. Thus P1-P25-P11 is moving one square to the right and moving 2 square to the top. Note that all of our primitives represent moving one square each. Paths along a brown primitive followed by a blue primitive represent diagonal moves for bishop.

3.7 Effect of Parameters

In this section we give a brief discussion on the effects of various parameters that we have used. One parameter of interest is the number of Gaussians that we have used to cover the sequences. This number

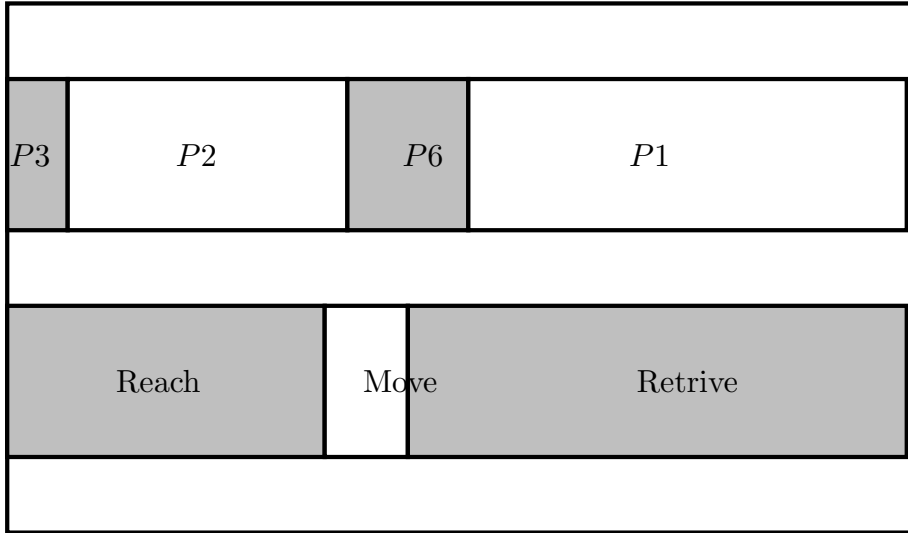


Figure 6: Comparing automatic segmentation with manually segmented primitives for one grasp sequence. The horizontal axis represents the length of the sequence. The plot compares a grasp sequence with length 119. Using the above diagram with Fig. 8, we can infer that P_3 and P_2 together constitute *approach* primitive, P_6 refers to *grasp* primitive and P_1 corresponds to *remove* primitive.

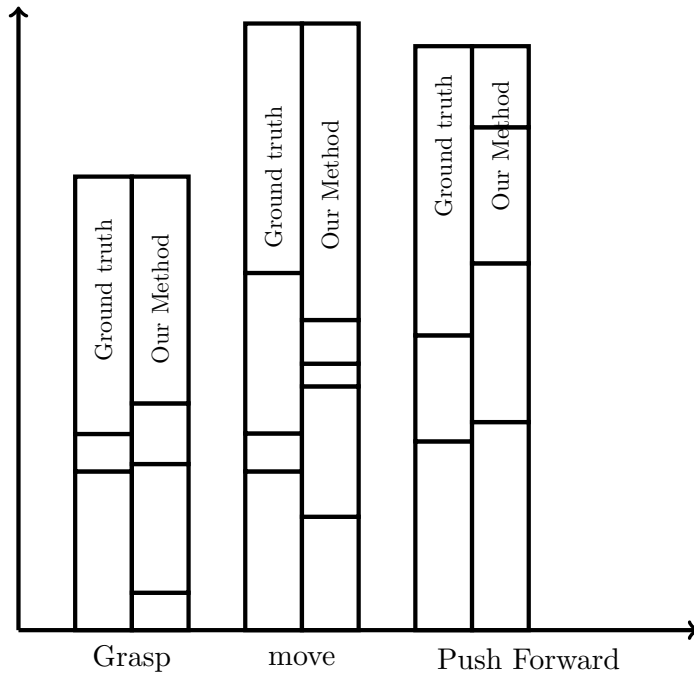


Figure 7: Comparing primitive segmentation with ground truth data. Average comparison results are shown. Height represents sequence length. Each of the segments in the bars represent a primitive.

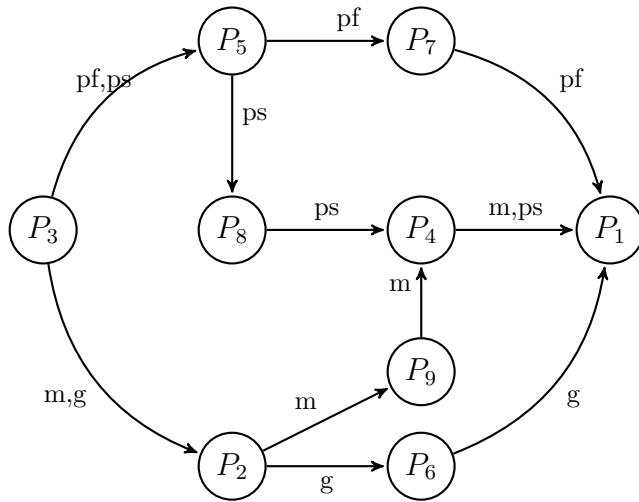


Figure 8: The temporal order for primitives of hand gesture data. Node number corresponds to different primitives. All actions start with P_3 and end with P_1 .

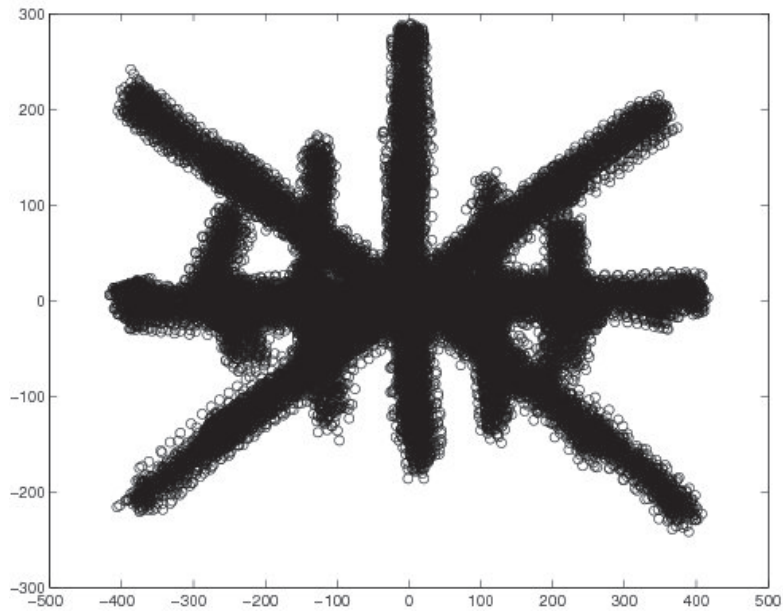


Figure 9: Some sample tracks for the chess data. Movements for rook, bishop and knight are shown.

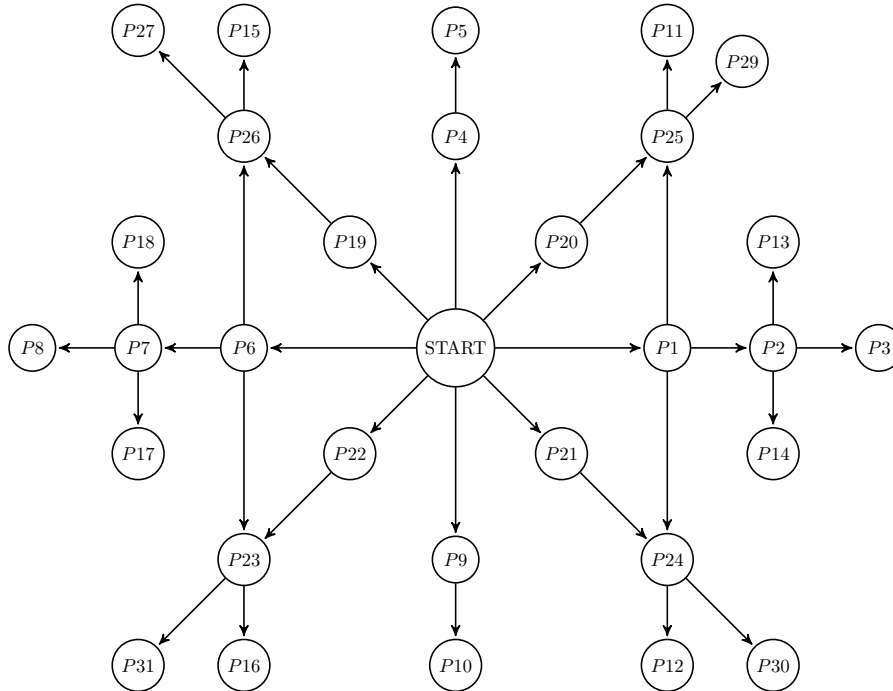


Figure 10: Extracted primitives for the chess data. Straight line paths represent moves for the rook and the queen. L-shaped paths represent moves for the knight. Paths along the diagonals represent diagonal moves for bishop.

and size of the Gaussians are inversely related. The effect of size of the Gaussians in the final result can be illustrated with the simulated data that we have used in Sec. 3.5.2. Allowing a small variance along the main diagonal of the estimated covariance matrix results in a covering as shown in Fig. 3(c). Allowing more variance will result in less number of Gaussians as shown in Fig. 11. In Fig. 11(a)-Fig. 11(c) the result of a particular covering is shown. Here we have exactly one Gaussian to cover the shared region and 2 more Gaussians to share the rest of the data. The final result matches the one shown in Fig. 3(o). Further decrement in the number of Gaussians will fail to discover the structure in the data as shown in Fig. 11(d)-Fig. 11(f). In this case one sequence is covered with a single Gaussian. This sequence cannot be modeled with a single Gaussian. Thus the size of Gaussians should be such that they do not violate the assumption of normality for the underlying data. The Gaussians should be big enough so that in any repetition of the same sequence, it should pass through the same state sequence.

To analyze the size of Gaussians further we can look at the chess scenario. If each of the Gaussians cover 2 squares, we will not be able to differentiate between moving one square and two squares. Therefore the value should be chosen such that it will not exceed the smallest primitive we expect to find.

Another parameter of interest is θ which is the threshold for deciding if two states should be merged or not. We chose this value to be half of average distance of adjacent pair states in a sequence. This

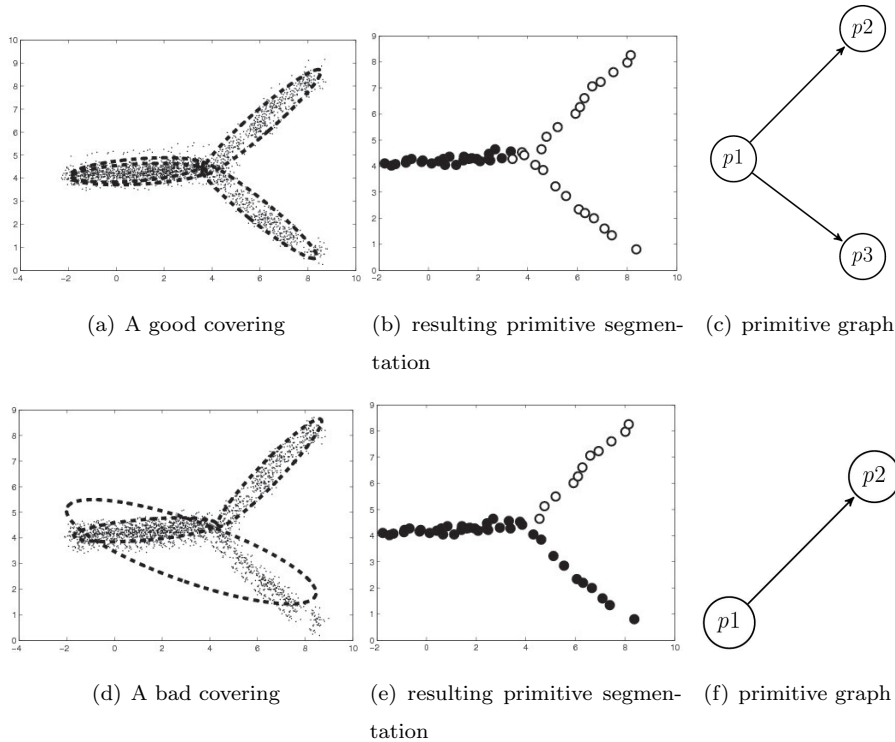


Figure 11: Illustration of the effect of parameters

ensures that states with good overlap are combined and represented by a single state.

4 Conclusions

We have presented and tested an approach for automatically computing a set of primitives and the corresponding stochastic context free grammar from a set of training observations. Our stochastic regular grammar is closely related to the usual HMMs. One important difference between common HMMs and a stochastic grammar with primitives is that with usual HMMs, each trajectory (action, arm movement, etc.) has its own, distinct HMM. This means that the set of HMMs for the given trajectories are not able to reveal any commonalities between them. In case of our arm movements, this means that one is not able to deduce that some actions share the grasp movement part. Using the primitives and the grammar, this is different. Here, common primitives are shared across the different actions which results into a somewhat symbolic representation of the actions. Indeed, using the primitives, we are able to do the recognition in the space of the primitives or symbols, rather than in the signal space directly, as it would be the case when using distinct HMMs. Using this symbolic representation would even allow to use AI techniques for, e.g., planning or plan recognition. Another important aspect of our approach is that we can modify our model to include a new action without requiring the storage of previous actions for it.

Our work is segmenting an action into smaller meaningful segments and hence different from [16]

where the authors aim at segmenting actions like walk and run from each other. Many authors point at the huge task of learning parameters and the size of training data for an HMM when the number of states are increasing. But in our method, transition, initial and observation probabilities for all states are assigned during our merging phase and hence the use of the EM algorithm [17] is not required. Thus our method is scalable to the number of states. Our approach of using states have a close connection to [18] but our method is superior in preserving the temporal order and hence in recognition.

In [19] primitives are found by thresholding angular velocities. In this work 4 dimensional data of joint trajectories were segmented and the resulting segments for each of the joints were interpolated with 100 elements. Elements of each joint were concatenated to form 400 dimensional vectors and PCA was applied to reduce the dimension to 11. k-means clustering was then performed in the latent space to find the control points. Reproduction was performed by projecting points back to the input space. The use of PCA was unnecessary complication in this case since the input space was only 4 dimensional. Another disadvantage with this method is that strong assumptions must be made about the segmentation of the data, and the duration of the primitives. We have provided a higher level abstraction of primitives using grammar which is not possible with the approach in [19]. In [20] human motions are represented as a binary tree. Actions are recognized by finding the optimal node transitions in the tree. The binary tree construction approach in [20] is not suitable for sequential learning. Each node in the tree is modeled with a single Gaussian. Such a modeling is not suitable for the initial levels of the tree since the nodes in those levels will contain many frames from observation sequences. Takano and Nakamura [21] have also approached the problem of finding motion primitives using HMMs. They have modeled each actions via a discrete Hidden Markov model. In their approach primitives are assumed to be known where as our approach learns primitives from the data. In [22] subgoals are detected from trajectories by detecting regions that the agent visits frequently on successful trajectories but not on unsuccessful trajectories. This paper addresses a reinforcement learning scenario and appears to be quite different. They use a diverse density approach which requires positive and negative instances.

It is interesting to note that stochastic grammars are closely related to Belief networks [23] where the hierarchical structure coincides with the production rules of the grammar. We will further investigate this relation ship in future work.

In future work, we will also evaluate the performance of normal and abnormal path detection using our primitives and grammars.

REFERENCES

- [1] S. Schaal, "Is imitation learning the route to humanoid robots?" *Trends in Cognitive Sciences*, vol. 3, no. 6, pp. 233–242, 1999.
- [2] A. Billard and R. Siegwart, "Robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 47, no. 2-3, pp. 65–67, 2004, robot Learning from Demonstration.

- [3] R. Dillmann, “Teaching and learning of robot tasks via observation of human performance,” *Robotics and Autonomous Systems*, vol. 47, no. 2-3, pp. 109–116, 2004, robot Learning from Demonstration.
- [4] S. Schaal, A. Ijspeert, and A. Billard, “Computational Approaches to Motor Learning by Imitation,” *Philosophical transactions: biological sciences*, vol. 358, no. 1431, pp. 537–547, 2003, philosophical transactions: biological sciences (The Royal Society).
- [5] K. Ogawara, S. Iba, T. Tanuki, H. Kimura, and K. Ikeuchi, “Recognition of human task by attention point analysis,” *Intelligent Robots and Systems, 2000. (IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference on*, vol. 3, pp. 2121–2126, 2000.
- [6] Y. Kuniyoshi, M. Inaba, and H. Inoue, “Learning by watching: extracting reusable task knowledge from visual observation of human performance,” *Robotics and Automation, IEEE Transactions on*, vol. 10, no. 6, pp. 799–822, Dec 1994.
- [7] S. Ekvall, “Robot task learning from human demonstration,” Ph.D. dissertation, KTH, Stockholm, Sweden, 2007.
- [8] A. Billard, S. Calinon, and R. Dillmann, “Robot programming by demonstration,” in *Springer Handbook of Robotics*, 2008, pp. 1371–1394.
- [9] S. Ekvall and D. Kragic, “Grasp recognition for programming by demonstration tasks,” in *IEEE Int. Conf. on Robotics and Automation, ICRA '05*, 2005, pp. 748–753.
- [10] L. R. Rabiner, “A tutorial on hidden Markov models and selected applications in speech recognition,” *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.
- [11] Y. Ivanov and A. Bobick, “Recognition of visual activities and interactions by stochastic parsing,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 852–872, 2000.
- [12] I. S. Vicente, V. Kyrki, and D. Kragic, “Action recognition and understanding through motor primitives,” *Advanced Robotics*, vol. 21, pp. 1687–1707, 2007.
- [13] V. Krueger, D. Kragic, A. Ude, and C. Geib, “Meaning of action,” *Int. Journal on Advanced Robotics, Special issue on Imitative Robotics, T. Inamura and G. Metta (eds.)*, 2007.
- [14] C. Stauffer and W. Grimson, “Learning Patterns of Activity Using Real-Time Tracking,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 747–757, 2000.
- [15] http://www.nada.kth.se/~danik/gesture_database/.
- [16] J. Barbič, A. Safonova, J.-Y. Pan, C. Faloutsos, J. K. Hodgins, and N. S. Pollard, “Segmenting motion capture data into distinct behaviors,” in *GI '04: Proceedings of Graphics Interface*. Canadian Human-Computer Communications Society, 2004, pp. 185–194.

- [17] A. Dempster, , M. Laird, and D. Rubin, “Maximum likelihood from incomplete data via the em algorithm,” *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, no. 1, pp. 1–38, 1977.
- [18] A. Bobick and A. Wilson, “A state-based approach to the representation and recognition of gesture,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 19, no. 12, pp. 1325–1337, Dec 1997.
- [19] A. Fod, M. J. Matarić, and O. C. Jenkins, “Automated derivation of primitives for movement classification,” *Autonomous Robots*, vol. 12, no. 1, pp. 39–54, 2002. [Online]. Available: <http://www.springerlink.com/content/n14t2272246jj54p>
- [20] K. Yamane, Y. Yamaguchi, and Y. Nakamura, “Human motion database with a binary tree and node transition graphs,” *Robotics: Science and Systems V*, 2009.
- [21] W. Takano and Y. Nakamura, “Integrating whole body motion primitives and natural language for humanoid robots,” in *Humanoid Robots, 2008. Humanoids 2008. 8th IEEE-RAS International Conference on*, Dec. 2008, pp. 708–713.
- [22] A. McGovern and A. G. Barto, “Automatic discovery of subgoals in reinforcement learning using diverse density,” in *ICML '01: Proceedings of the Eighteenth International Conference on Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2001, pp. 361–368.
- [23] M. I. Jordan, Ed., *Learning in graphical models*. MIT Press, 1998.