2008 IEEE International Conference on
Robotics and Automation
Pasadena, CA, USA, May 19-23, 2008

Modeling and Recognition of Actions through Motor Primitives

David Martínez and Danica Kragic
Computational Vision and Active Perception Lab
Centre for Autonomous Systems
KTH-Royal Institute of Technology, Stockholm, Sweden

Abstract— **We investigate modeling and recognition of object manipulation actions for the purpose of imitation based learning in robotics. To model the process, we are using a combination of discriminative (support vector machines, conditional random fields) and generative approaches (hidden Markov models). We examine the hypothesis that complex actions can be represented as a sequence of motion or action primitives. The experimental evaluation, performed with five object manipulation actions and 10 people, investigates the modeling approach of the primitive action structure and compares the performance of the considered generative and discriminative models.**

I. Introduction

In artificial systems, imitation has been frequently used for automated programming and control of robots, for finding more natural ways for interacting with robots and task learning in general. Imitation has also been viewed as the capability to acquire new skills by observation based on some existing behavioral repertoire, [1]. In [2], it has been shown that an action perceived by a human can be represented as a sequence of clearly segmented action units. These action units can then serve as the basis for building up the behavioral repertoire. Thus, the action recognition process may be considered as an interpretation of the continuous human behaviors which, in its turn, consists of a sequence of action primitives such as *reaching*, *picking up*, *putting down*. The notion of actions and action primitives is thus of significant importance for building structures that directly link sensory and motor systems of artificial systems since they define the necessary mapping for the implementation of the perception-action mechanism.

In this work, we study the problem of modeling and recognition of actions and action primitives using Support Vector Machines (SVM), [3], Hidden Markov Models (HMM), [4] and Conditional Random Fields (CRF), [5]. To start with, SVM is used to model and recognize individual action primitives. Actions, built from a set of action primitives, are then modeled using HMMs and CRFs and their performance is evaluated and compared. We also evaluate the plausibility of using CRFs for recognition of composite actions. The measurements are based on four magnetic sensors where each of the sensors provide a complete pose estimate. The contributions are:

- We investigate modeling strategies for object manipulation actions that are very similar to each other (*grasping*, *pushing*, *moving*). Most of the current work on arm/hand action recognition concentrates on actions that are easy to discriminate (*waving*, *pointing*).
- We implement, evaluate and compare both generative and discriminative approaches while most of the reported work concentrates on one of the approaches.
- We consider the problem of recognition both on the primitive and on the composite action level.
- Our measurements are based only on four magnetic sensors while most other systems use complex motion capture systems.

II. Related work

There is a large body of work on the problem of human action recognition from images or from 3D positions on the human body, [6]. Examples from computer vision community include [7], where actions are represented as a sequence of key postures. Segmentation is then performed implicitly by searching the observation sequence for key postures that then are used for recognition. The key postures are represented as topological edge maps extracted from video frames.

An alternative approach is to avoid the segmentation problem altogether by employing a discriminative action recognition approach. For example, [8] use conditional random fields (CRF) for recognition of full human body actions. This method for modeling sequential data is similar to HMM but has the advantage that no explicit model of the sequence of observations has to be learned, thereby rendering explicit data segmentation unnecessary as well. The downside of CRF, as with any discriminative approach, is however the inability to generalize to previously unseen action examples when the detailed imitation of the pose is needed.

In terms of the adopted theoretical framework, support vector machine (SVM) has been applied to several different application areas such as visual and speech data modeling and recognition. One of the early works on SVMs, [9], presented a drawback of the method when working with sequential data, namely, that SVM lacks a way of handling the time dependencies in the data. In order to use time dependent sequences as SVM input, variable length time sequences can be either normalized to same length before applying the SVM. Another approach is to embed dynamic time warping (DTW) directly into the SVM kernel function [10]. Third, probably the most common way to handle the "time problem" is to combine a SVM with Hidden Markov Models (HMM) [9], [11], [12]. SVM is still used to classify single points or brief time windows, but the output of the

978-1-4244-1647-9/08/$25.00 ©2008 IEEE.

SVM is then used an input to a HMM which then finds the most probable path or sequence in consideration of time.

From the point of view of imitation learning or "learning by showing", the primitives are an attractive option since they can alleviate mapping motion from humans to robots which differ in their embodiment. In addition, having a common vocabulary of primitives can aid in task understanding and planning as the task can be then described as a sequence of events.

III. Modeling Actions

In the work presented here, we consider five different object manipulation actions: 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. To include variation in the actions, each action is performed in 12 different conditions, namely on two different heights, two different locations on the table, and having the demonstrator stand in three different locations (0, 30, 60 degrees), see Fig. 1. All actions are demonstrated by 10 people.

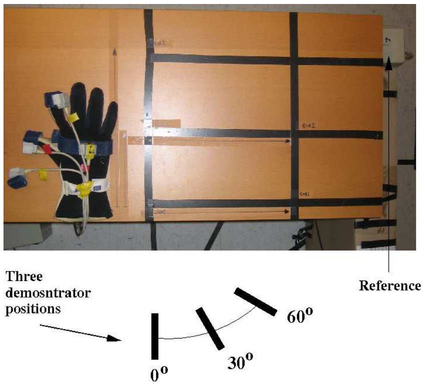

Fig. 1. Experimental setup.

The movement is measured using a Nest-of-Birds magnetic tracker. The test subject is endowed with four sensors each registering their full 3-dimensional pose with respect to a reference, see Fig. 1. The sensors are located on: a) chest, b) back of hand, c) thumb, and d) index finger. The chest sensor is used to provide a reference to the demonstrator position while the back of the hand can be used as a reference for the thumb and index finger. The measured sequences have been annotated by hand such that the current action primitive is known for training.

A. Support vector machines

Support vector classification aims at separating data classes, mapped into a high dimensional feature space, by hyperplanes with a maximal margin to the classes. A hyperplane represents the decision boundary of the classifier with feature vectors on one side belonging to one class and vectors on the other side to another one. To represent complex decision boundaries, the mapping (kernel) from the original feature space to the high dimensional space is nonlinear. In this work, a standard SVM with Gaussian kernel is used.

$$\underbrace{\underbrace{11111111}_{ap}\underbrace{22222}_{gr}\underbrace{7777777}_{rm}}_{grasping\ \ an\ \ object}\underbrace{\underbrace{111111}_{ap}\underbrace{2222}_{gr}\underbrace{555555}_{rt}\underbrace{7777}_{rm}}_{rotating\ \ an\ \ object}$$

Fig. 2. Example output of the SVM classifier.

In our work, we apply SVM classification to multiple classes each representing an action primitive class. For this purpose, we adopt one-against-one approach. In other words, by denoting the number of classes by k, $k(k-1)/2$ classifiers are trained using all pairs of classes. To classify a sample from an unknown class, it is classified by all classifiers, and each result is a vote for the class. Majority voting is then used to decide the class of the sample. The one-against-one approach has been found very successful with SVMs but it suffers from increased number of individual classifiers when the number of classes is very high.

The output of the SVM is then a sequence of classified primitive actions at each time instant. In Fig. 2 we show an example of two concatenated actions: grasping and rotating an object. A grasping action is composed of three primitives (approach, grasp and remove), while the rotation is composed of four primitives (approach, grasp, rotate and remove hand). In terms of lengths, a real grasp ,am be composed of 30 approach, 20 grasp and 50 "remove hand with object" primitives. Our SVM implementation uses 7 different classes, which correspond to the 7 primitive actions: approach, grasp, rotate, push forward, push side, remove and remove with object. It is worth mentioning the difference between a primitive and an action: a primitive is our basic unit, like a letter in a word, while an action is a composite of primitives, like a word.

B. Markov chain and hidden Markov models

To model the temporal dependencies of actions, the first approach we adopt are time-homogeneous Markov chain models where the state transition probabilities are invariant over time. As a short reminder, denoting the state i by ω_i, the time evolution of states can then be described using the state transition probabilities $P(\omega_j(t+1)|\omega_i(t)) = a_{ij}$. The states themselves are hidden, not directly observable. Instead, in each state, an observation $\mathbf{x}(t)$ is made. The observation depends only on the current state according to a selected probabilistic model, that is, $P(\mathbf{x}(t)|\omega_i(t)) = P(\mathbf{x}|\omega_i)$. If the set of observations X is discrete and finite, $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_M\}$, the observation probabilities can be written more shortly as $P(\mathbf{x}_j|\omega_i) = b_{ij}$. Finally, the probability of starting in state ω_i can be defined as $\pi_i = P(\omega_i(1))$. Thus, the parameters can be collected to matrices $\mathbf{A}$ and $\mathbf{B}$ and a vector $\boldsymbol{\pi}$.

The objective of the work presented here is to model actions based on motor primitives. Motor primitives correspond to individual states of the HMM. A typical approach for using HMMs in recognition is to build a single HMM for each class to be recognized and then determine the class of an unknown sample by using the maximum likelihood method. In our work, we take another approach and represent the whole set of actions with a single HMM, such that different paths through the HMM correspond to different actions. This is because many actions contain similar parts and since the HMMs are not considering dependencies between the observations as such, we believe that this model may account for some of the problems caused by this. A simple example is shown in Fig. 3. Here, both rotating and pushing an object both require first the hand to approach the object.

As mentioned earlier, a hypothesis followed in this work is also that more complex actions can be modeled using a set of motor primitives. Thus, instead of making a choice between several HMMs, the most probable path through the HMM is sought. The path is found by the Viterbi algorithm [4], a dynamic programming based algorithm for determining the maximum likelihood path through a HMM given a sequence of observations $(\mathbf{x}(1), \mathbf{x}(2), \ldots)$. It finds the state sequence $(\omega(1), \ldots)$ for which $P\left(\mathbf{x}(1), \ldots, \mathbf{x}(T) | \omega(1), \ldots, \omega(T)\right)$ is maximal. We employ the computationally tractable solution based on defining it recursively.

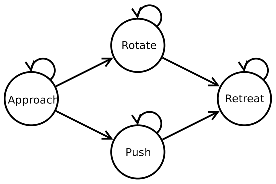

Fig. 3. Modeling two actions (rotate, push) using primitives.

For initial learning of the HMM parameters, an alternative approach to the traditional Baum-Welch learning is adopted. We assume that the training data is labeled, that is, for each time step, the current motor primitive is known. Then, the transition probability matrix $\mathbf{A}$ can be directly estimated from the training data, as if in the case of a Markov chain model instead of a HMM. We use the maximum likelihood estimate, in other words, the transition probabilities are calculated directly from the training data. The output of the SVM is used as the observations of the HMM. The observation probabilities need also be estimated as it is not expected that the classifier will be able to classify all samples correctly. Maximum likelihood estimation using the known correct classes is also used to estimate the observation probabilities. Therefore, the observation matrix $\mathbf{B}$ corresponds to the confusion matrix of the classifier.

1) Action modeling using HMMs: The hypothesis in the modeling is that each of the manipulation primitives is generic and that their number is limited. However, the best applicable set of primitives is not known and one of the goals of our previous work was to study how the manipulation actions can be considered in terms of primitives. In [13], we have investigated two different models of action representation, see Fig. 4. Approach 1 considered each of the manipulation actions as a primitive. In addition to the manipulation actions, two assisting actions, *approach* and *remove* are inherent in all action sequences (see Fig. 4). The assisting actions alleviate the segmentation of the manipulation part of the action. Approach 2 considered therefore that the manipulation part of the action can be composed of multiple primitives. The model on the right in Fig. 4 can be chosen based on the knowledge that the rotation and moving the object require grasping. Our working hypothesis in [13] was that Approach 2 would be more effective in recognizing actions compared to the first approach. In addition it would allow learning of new actions based on the known primitives. Considering both approaches, an action was represented by a separate path through the left-to-right Markov model. We have shown how the process of embedding new actions given primitives can be formalized.

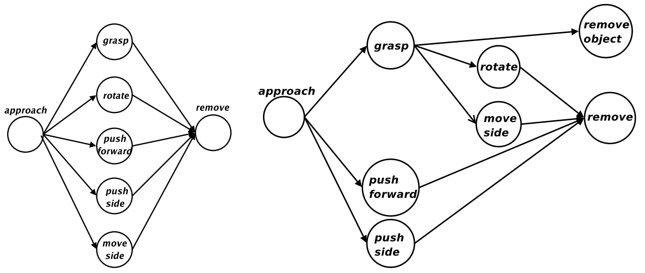

Fig. 4. (left) Approach 1: Actions as primitives; (right) Approach 2: Composite actions.

C. Conditional Random Fields

A conditional random field, [5] is a stochastic process for segmenting and labeling data; it is a discriminative framework that describes the probability of having a set of labels S given the set of observations O, $\mathrm{P}(S|O)$. It is modeled as an undirected graph $\mathrm{G} = (\mathrm{V}, \mathrm{E})$ where each vertex is associated with a label S_i. Only vertices connected by an edge are conditionally dependent. Although the graph can be as complex as desired, in this work we focus on linear CRFs, where any vertex v_i is connected to the previous and the following ones (v_{i-1} and v_{i+1}). Linear CRFs are adequate when data is a sequence $O = o_1, \ldots, o_n$, and then the resulting set of labels will be also a sequence $S = s_1, \ldots, s_n$. Each s_i is an element of a finite alphabet γ.

At the use stage, we need to obtain the labeled sequence with the highest probability given the set of observations O. That probability is defined as

$$\mathrm{P}_\lambda(\mathrm{S|O}) = \frac{\exp\left(\sum_{i \in 1..N} \lambda_i \cdot \mathrm{f}(s, O, i)\right)}{\mathrm{Z}_\lambda(O)} \tag{1}$$

where $\mathrm{Z}_\lambda(O)$ is a normalization factor of the form

$$\mathrm{Z}_\lambda(O) = \sum_{s \in S} \exp\left(\sum_{i \in 1..N} \lambda_i \cdot \mathrm{f}(y, O, i)\right) \tag{2}$$

Here, $\Lambda = \{\lambda_i, i \in 1..N\}$ are the parameters that define the CRF model (those that will we estimated at training), while

$\mathrm{f}(s, O, i)$ are feature functions. Those features are used to describe the attributes of the data. The sequence of labels with the highest probability is then found by $\arg\max_s \mathrm{P}_\lambda(\mathrm{S}|\mathrm{O})$. Regarding training, we have a set $\mathrm{T} = \{(\mathrm{O}^{(i)}, \mathrm{S}^{(i)})\}_{i=1}^{L}$. Each $\mathrm{S}^{(i)}$ is the correct labeling for the set of observations $\mathrm{O}^{(i)}$, and L is the number of training pairs. The goal of the training algorithm is to estimate all the parameters Λ that define the CRF model; there is one parameter per feature. One way of determining these parameters is by maximizing the log-likelihood function $\mathrm{L}(\lambda) = \sum_{i=1}^{L} \log \mathrm{P}_\lambda(\mathrm{S}^{(i)}|\mathrm{O}^{(i)})$

1) Data representation and action modeling using CRFs: First, the training data has to be labeled for the estimation of the CRF parameters. The easiest way is to have an alphabet of labels $\gamma = \{\texttt{g\_}, \texttt{r\_}, \texttt{pf}, \texttt{ps}, \texttt{m\_}\}$; where each member of γ represents one of the 5 actions presented in Section III. All the observations that belong to the same action receive the same label. This labeling approach, denoted as Format 1, was used to compare the performance of the CRF and the HMM approach. We also perform recognition of continuous actions where it is assumed that the observation sequence is built of multiple actions denoted as Format 2. Here, the label set is denoted as $\gamma = \{1, 2, 3, 4, 5, 6, 7\}$ representing the number of primitives. Table I shows an example of both formats for an input sequence consisting of two actions, a grasping and a rotation.

Once the CRF returns the most probable sequence, we have to post-process it in order to obtain a sequence of actions; with the first format this is trivial, since keeping the label that received the majority of votes performs well in most of the cases. For the second format, the approach is more complex, as we still have a sequence of primitive labels. Here, we make the use of regular expressions, as our results can be seen as text strings. We match several regular expressions and make appropriate substitutions; first we make substitutions of the kind `1546` → `r_` (full isolated actions); second, `154` → `r_` (actions without retreat); third, actions without approach, and finally, actions without either approach or retreat primitives, like `54` → `r_`. The order is important, because if we first substitute `154` in the sequence `1546136`, at the end we will have an isolated `6`, which will be considered as a mistake while in fact it is not. Table I shows the final solution after this post-processing step.

Input sequence	1 2 5 5 5 7 7 1 1 5 5 7 4 4 6
CRF output, format 1 CRF output, format 2	g_g_g_g_g_g_g_r_r_r_r_r_r_r_r_ 1 1 5 5 5 7 7 1 1 5 5 4 4 4 6
Final result	g_ r_

TABLE I

FINAL RESULT FROM AN OUTPUT LABELED SEQUENCE

IV. SYSTEM DESIGN AND IMPLEMENTATION

We start by pre-processing the input data for noise removal and continue with the SVM classification; the output is a sequence of primitives that we use to feed a HMM and a CRF. Two cases considered in the experimental evaluation are i) comparison of the HMM and a linear CRF for action recognition, and ii) the use of CRF for simultaneous segmentation and classification of continuous action sequences.

A. Pre-processing

The following sensor measurements are used:

- position of the hand relative to the chest: x, y and z
- position of the index relative to the hand: x, y and z
- position of the thumb relative to the hand: x, y and z
- velocity of the hand: v_x, v_y and v_z.

The hand and finger locations were transformed into the chest reference frame. The position of both the thumb and index was calculated with respect to the back of the hand. The velocity was estimated by time differences between two consecutive time instants. Each dimension was linearly scaled to interval $[0, 1]$.

B. Action recognition

The training data collected are representing 'isolated' actions; we did not collect continuous sequences of actions. The process of labeling complex continuous sequences of actions for the purpose of using them as the input for the CRF training would have to be done by hand, which is a time-consuming task. Instead, we generate continuous action sequences by concatenating primitive actions. While an action always begins with an approaching an object, and ends with a retreating of the hand in a movement involving two consecutive actions on the same object, we do not move away the hand after the first action and approach again to the object, as the hand is already in the vicinity of the object. This adds more complexity to the classification system, as now when we mix actions and action primitives in a sequence, some of them are very short (there are continuous grasping actions with 5 observations) while others (mainly rotations) are very long (some of them have more than 180 observations).

C. Classification of continuous action sequences

In order to evaluate the performance of the CRF based approach, we compare it with the ground truth data using Needleman-Wunsch algorithm [14] to align the sequences. This algorithm is based on the dynamic programming approach and it is commonly used to find the most probable global alignment between two sequences of data. We want to measure differences in terms of the number of inserted, deleted and misclassified actions. After running the algorithm, it returns both sequences aligned in the optimal way and a score, which gives us the similarity or dissimilarity between both sequences, depending on a set of predefined parameters. In Figure 5, we show an example of the best alignment of two sample sequences; the three possible types of errors are labeled as D (Deletion), I (Insertion) and M (Misclassification).

Two parameters must be set, one called a gap, associated with the weight given to an insertion or a deletion, and S, a similarity matrix. S_{ij} (and S_{ji}) indicates how similar symbols

```
g_r_g_pfr_pspfg_  r_r_g_r_g_ps
  r_g_pfr_  pfg_pfr_r_g_r_pfps
↑                 ↑         ↑
D                 I         M
```

Fig. 5. Best alignment of two sequences.

t_i and t_j are. Each t_i belong to a finite alphabet T that contains all the possible symbols that can be inside the sequences to be compared; in our case T = {g_,r_,pf,ps,m_}. The values for both parameters in this work are $gap = 1$ and

$$S_{\mathbf{ij}} = \begin{cases} 1 & \text{if } i \neq j \\ 0 & \text{if } i = j \end{cases}$$

With these parameters, the algorithm will return the sum of insertions, deletions and misclassified actions found after the alignment. Different metrics could be used to obtain a numerical measure of how good the solution is; we have chosen the F1-score:

$$F1 = \frac{2PR}{P+R}$$

where P denotes precision and R denotes recall.

V. Experimental evaluation

We gathered the training data with 10 people, performing all actions in 12 different conditions. For each action, there are thus 120 different samples. Based on these, we generated 240 mixed sequences consisting of actions and primitive actions. Prior to executing the actions, the demonstrators were given only an oral explanation of the task; for this reason, both the inter- and cross-personal variance in the data is high. For SVM learning the training sequences were segmented and labeled manually.

A. HMMs and CRFs for isolated recognition

We start with a comparison of recognition rates for action modeling using HMMs and CRFs. Although we use input sequences with only one action, that action can be either with or without the approach/remove part. In our previous work, [13] we have evaluated different structures of HMMs for action modeling. Based on that, we kept the composite action model that performed best. We note that this model is applicable for recognition of both composite and primitive actions since parsing through the model and state change depends on the probability of observations. We performed leave-one-out cross validation for all ten cases and averaged the results. Tables II– IV show the results in form of confusion matrices. We added a sixth column, 'unrec', used to count all the actions that could not be classified. For instance, in some cases, the CRF returns a sequence with two different labels having the same probability; in that case, we assume that we cannot disambiguate.

The strength of the CRFs in being able to take several observations into account, significantly improves the recognition of grasping actions (g_). On the other hand, CRFs experience more difficulty in recognizing move-to-side actions(m_) which is often confused with push-to-side. The explanation is that the move-to-side action is explicitly embedded in the HMM model and requires grasping primitive action to occur before the actual side movement. When parsing through a HMM model, the right route will be taken and the probability of recognition move-to-side rather than push-to-side will be higher. On the other hand, when training the CRF, inter-and cross-personal variations in performing these two actions affect the representation more significantly. We have also performed the evaluation of the CRF based on the format 2 of the data. The results are shown in Table IV. A slightly improved performance compared to format 1 can be seen.

	g_	r_	pf	ps	m_	unrec
g_	63.8	7.5	2.5	1.7	19.1	5.4
r_	0	98.7	0	0	1.3	0
pf	2.1	0	94.6	0.8	0.8	1.7
ps	0	1.7	4.1	56.3	37.5	0.4
m_	0	3.4	0	5.8	90.8	0

TABLE II

HMM results for individual action recognition.

	g_	r_	pf	ps	m_	unrec
g_	95.9	0.5	0.5	2.2	0.9	0
r_	0	98.3	0.4	0	1.3	0
pf	1.7	1.7	92.9	3.7	0	0
ps	2.1	0.8	5	59.2	32.9	0
m_	1.7	1.3	0	19.5	77.1	0.4

TABLE III

CRF results for individual action recognition and format 1.

B. CRFs for continuous recognition

The second evaluation consisted in testing CRFs with continuous sequences of actions, once more using the mixed training sequences of composite and primitive action samples. For this purpose, we consider only the format 2 of the data, as explained in Section III-C.1. CRFs are able to learn a grammar if the training process is modeled suitably and we can benefit from it in our case. For instance, they can learn that having a grasping after a rotation is very common, but that having two consecutive graspings is not probable. In our experiments, we have constructed a few simple task models. Each task is composed of 10 sentences, and each sentence contains a random amount of actions, always between 3 and 10 where the actions are randomly chosen. For each task, training and testing datasets have been made. We can see each sentence corresponding to a specific tasks such as, for example, serving somebody a coffee or setting up a dinner table. Training data contains 10 samples of each task; for each sample we have only used actions from the same person, as this is what we anticipate will happen in realistic applications. Test data consist of 300 sequences, each one

	g_	r_	pf	ps	m_	unrec
g_	92.2	1.8	0.9	3.2	0.9	0.9
r_	0	95	0.4	0	2.5	2.5
pf	2.1	0	91.9	3.4	0.9	1.7
ps	2.5	0	3.3	62.1	31.3	0.8
m_	1.3	0.8	0	17.5	79.6	0.8

TABLE IV

CRF RESULTS FOR INDIVIDUAL ACTION RECOGNITION AND FORMAT 2.

following one of the tasks. For testing, actions inside the task are chosen amongst all the people in the original dataset. In short, we assume training with one person but test with several. The overall averaged results are shown in Table V.

Since in the previous section we have observed that the confusion between move to a side and push side is an important problem, we attempted to analyze the system with and without that specific action. Another feature analyzed is what happens when two consecutive actions are equal; the hypothesis is that if both actions are action primitives, they will be mixed and then, the number of deletions when estimating the final classification results will increase.

Test	With m_	Repeat actions	F1 score
1	No	No	92.6
2	Yes	No	84.8
3	No	Yes	92.8
4	Yes	Yes	85.2

TABLE V

CRF RESULTS FOR CONTINUOUS SEQUENCES

From the results in Table V, it can be seen that the difference between having and omitting move-to-side action is significant. The results also show that when we consider repeated continuous actions results are worse which means that the segmentation and recognition can be performed simultaneously. The fact that they are even a bit better in this case is that the training and test data are different compared to the previous experiment.

Although comparing results for individual and continuous recognition is not completely fair, as data representation differs, we see that continuous recognition is still rather good. Moreover, these results can be considered as a lower bound of the results we would obtain in a real scenario. This is due to the fact that many consecutive continuous actions are mixed, and in the solution they appear as only one action. For instance, a primitive grasping action followed by a primitive rotation action on the same object can be considered as a single composite rotation action. This suggests that, for future work, a hierarchical model may be considered.

VI. CONCLUSIONS

We have presented a system for modeling and recognition of primitive and composite actions using generative and discriminative machine learning approaches. We have started by using support vector machines for primitive action classification and integrated this with models that can take care of the temporal aspects of actions, namely hidden Markov models and conditional random fields. We have built upon our previous work where only hidden Markov models where considered for isolated action recognition - apart from using the conditional random fields we are also investigating their use in a continuous action recognition scenario. We follow the assumption that we can build a system consisting of different sensory primitives from which more complex actions can be built. These sensory primitives should be natural and easily used in programming motor primitives for robots.

The experimental evaluation performed with seven primitives and five composite actions is based on the training data obtained with ten people in twelve different conditions. The recognition rates on isolated actions show similar performance between hidden Markov models and conditional random fields, with latter having the capability of classifying very short activities due to the ability of modeling the dependence between the observations. In the case of the continuous action recognition, conditional random fields show high recognition rates.

REFERENCES

[1] M. Mataric, “Sensory-motor primitives as a basis for imitation: Linking perception to action and biology to robotics,” 2000.

[2] D. Newton et al, “The objective basis of behavior unit,” *Journal of Personality and Social Pshychology*, vol. 35, no. 12, pp. 847–862, 1977.

[3] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines*. Cambridge, UK: Cambridge University Press, 2000.

[4] 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.

[5] J. Lafferty, A. McCallum, and F. Pereira, “Conditional random fields: Probabilistic models for segmenting and labeling sequence data,” in *Proc. 18th International Conf. on Machine Learning*, pp. 282–289, Morgan Kaufmann, San Francisco, CA, 2001.

[6] T. B. Moeslund, A. Hilton, and V. Krüger, “A survey of advances in vision-based human motion capture and analysis,” *Comput. Vis. Image Underst.*, vol. 104, no. 2, pp. 90–126, 2006.

[7] J. Sullivan and S. Carlsson, “Recognizing and tracking human action,” in *European Conference on Computer Vision*, 2002.

[8] C. Sminchisescu, A. Kanaujia, and D. Metaxas, “Conditional models for contextual human motion recognition,” *Computer Vision and Image Understanding*, vol. 104, no. 2-3, 2005.

[9] S. E. Golowich and D. X. Sun, “A support vector/hidden Markov model approach to phoneme recognition,” in *ASA Proceedings of the Statistical Computing Section*, pp. 125–130, 1998.

[10] H. Shimodaira, K. Noma, M. Nakai, and S. Sagayama, “Dynamic time-alignment kernel in support vector machine,” in *Advances in Neural Information Processing Systems*, pp. 921–928, 2001.

[11] M. Bartlett, G. Littlewort, B. Braathen, T. Sejnowski, and J. Movellan, “A prototype for automatic recognition of spontaneous facial actions,” in *Advances in Neural Information Processing Systems, NIPS 2003*, pp. 1271–1278, 2002.

[12] D. Surendran and G.-A. Levow, “Dialog act tagging with support vector machines and hidden Markov models,” in *Interspeech 2006 — ICSLP*, (Pittsburgh, PA, USA), 2006.

[13] V. Kyrki, I. Serrano, D. Kragic, and J.-O. Eklundh, “Action recognition and understanding using motor primitives,” in *In RO-MAN’07: The 16th IEEE International Symposium on Robot and Human Interactive Communication, Jeju Island, Korea*, 2007.

[14] S. Needleman and C. Wunsch, “A general method applicable to the search for similarity in the amino acid sequences of two proteins,” *Journal of Molecular Biology*, vol. 48, pp. 443–453, 1970.