

# Evaluation of Layered HMM for Motion Intention Recognition

Daniel Aarno and Danica Kragic

Centre for Autonomous Systems and Computational Vision and Active Perception Laboratory  
Royal Institute of Technology (KTH) - SE-100 44 Stockholm, SWEDEN

**Abstract**—We evaluate Layered Hidden Markov Models (LHMM) for motion intention recognition based on action-primitives or gestemes. The proposed methodology uses three different HMM models at the gesteme level: one-dimensional HMM, multi-dimensional HMM and multi-dimensional HMM with Fourier transform. These three models are evaluated with respect to the number of gestemes, the influence of the number of training samples, the effect of noise and the effect of the number of observation symbols.

## I. INTRODUCTION

Learning and recognizing human skills is an important research problem in teleoperation, programming-by-demonstration (PbD), Human-Machine Collaborative systems (HMCS) and human-computer interaction [1], [2], [3], [4], [5], [6], [7], [8], [9], [10]. The problem studied in this paper is recognition of an operator’s intention in a teleoperated system. The assumption is that if the intention can be recognized online in real-time, it is possible to improve the task execution by allowing the system to adapt to the operator’s need by applying the correct control mode in the transfer step [3], [6], [7].

The methodology is based on a layered hidden Markov model (LHMM) where there is a HMM modeling the overall task and a HMM modeling each action primitive, hereafter referred to as gesteme. In particular, given a set of gestemes generated in the learning step, the online recognition step is responsible for choosing the most likely mental state/intention given the measurements. We aim at extending motion intention recognition using HMMs to tasks in 2D/3D with a large set of primitives. Although a straightforward approach may be to model every high-level task with a single complex HMM, we are interested in learning hierarchical representations of tasks for which the low-level set of skill primitives is common. We thoroughly evaluate the proposed methodology using three different types of HMM models at the gesteme level. These three types of models are evaluated with respect to the number of gestemes, the influence of the number of training samples, the effect of noise and the effect of the number of observation symbols.

## II. MOTIVATION AND RELATED WORK

Hidden Markov models have been used frequently in trajectory tracking and virtual fixture applications, [3], [11], [12]. In our previous work, [6], we used a combination of K-means clustering, SVMs and HMMs to automatically extract virtual fixtures during task execution. This was then used to segment the task into a number of subtasks, corresponding to a particular fixture and provide online assistance by applying

the correct fixture during subsequent task executions. The output of the HMM was used to adjust the compliance of the virtual fixture so that the fixture was harder when the system was more certain about the current state. This allowed the system to handle task-deviations (i.e. none of the subtasks were executed) by lowering the stiffness of the fixture. Work presented in [1], shows how HMMs can be used at the gesteme level as opposed to the task level. The basic interaction primitives are modeled by an HMM and the task is represented as a network of simpler HMMs. In our current work we combine gesteme classification with task-level modeling by the suggested LHMM approach in order to handle more complicated types of tasks. This is an extension of the work presented in [6] where the SVM classifiers are replaced by the more expressive HMM classifiers.

Our work relates to the work on Hierarchical Hidden Markov Models (HHMMs). HHMMs have been used to model stochastic structures at different levels in speech and text recognition, modeling of group actions in meetings and extracting contexts, [13], [14], [15]. The reason for using the LHMM instead of the HHMM structure is that it corresponds well with the intended scenario. At the lowest level there are several models active in parallel classifying sensor data into action primitives. The classification then progresses through the LHMM until finally the task is modelled at the top level. In other words, in the LHMM there are several HMMs running at parallel at any given level of the hierarchy, where each HMM corresponds to a different “concept”. Even though a fully connected HMM could always be used if enough training data is available, it is useful to constrain the model by not allowing arbitrary state transitions. In the same way, it can be beneficial to embed the HMM into a more complex structure. In principle, this may not facilitate the solution of more complex problems compared to the basic HMM but can solve some problems more efficiently when it comes to the amount of required training data.

## III. LAYERED HIDDEN MARKOV MODELS

In our approach, we adopt layered hidden Markov model (LHMM), [14] that consists of  $N$  levels of HMMs where the HMMs on level  $N + 1$  corresponds to observation symbols or probability generators at level  $N$ . Every level  $i$  of the LHMM consists of  $K_i$  HMMs running in parallel, Fig. 1. At any given level  $L$  in the LHMM a sequence of  $T_L$  observation symbols  $\mathbf{o}_L = \{o_1, o_2, \dots, o_{T_L}\}$  can be used to classify the input into one of  $K_L$  classes, where each class corresponds to each of the  $K_L$  HMMs at level  $L$ . This classification can then

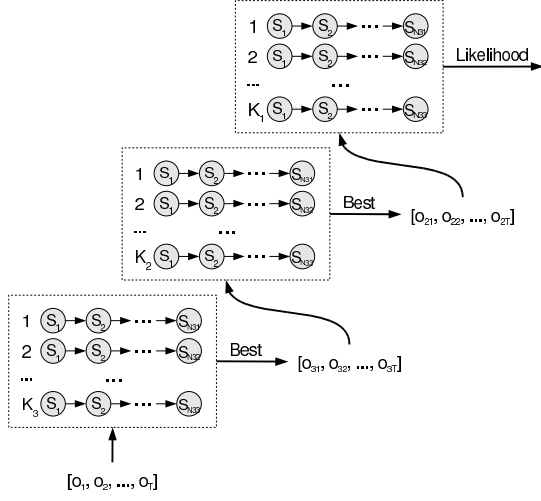


Fig. 1. A layered hidden Markov model.

be used to generate a new observation for the level  $L - 1$  HMMs. At the lowest layer, i.e. level  $N$ , primitive observation symbols  $\mathbf{o}_p = \{o_1, o_2, \dots, o_{T_p}\}$  would be generated directly from observations of the modeled process. For example, in a trajectory tracking task, the primitive observation symbols would originate from the quantized sensor values. Thus at each layer in the LHMM, the observations originate from the classification of the previous layer, except for the lowest layer where the observation symbols originate from measurements of the observed process.

It is not necessary to run all levels at the same level of granularity. For example, it is possible to use windowing at any level in the structure so that the classification takes the average of several classifications into consideration before passing the results up the layers of the LHMM. Instead of simply using the winning HMM at level  $L + 1$  as an input symbol for the HMM at level  $L$ , it is possible to use it as a probability generator by passing the complete probability distribution up the layers of the LHMM. Hence, instead of having a “winner takes all” strategy where the most probable HMM is selected as an observation symbol, the likelihood  $L(i)$  of observing the  $i$ -th HMM can be used in the recursion formula of the level  $L$  HMM to account for the uncertainty in the classification of the HMMs at level  $L + 1$ . Thus, if the classification of the HMMs at level  $n + 1$  is uncertain, it is possible to pay more attention to the a-priori information encoded in the HMM at level  $L$ .

A LHMM could in practice be transformed into a single layered HMM where all the different models are concatenated together. Some of the advantages that may be expected from using the LHMM over a large single layer HMM is that the LHMM is less likely to suffer from over-fitting since the individual sub-components are trained independently on smaller amounts of data. A consequence of this is that a significantly smaller amount of training data is required for the LHMM to achieve a performance comparable of the HMM. Another advantage is that the layers at the bottom of the LHMM, which are more sensitive to changes in the

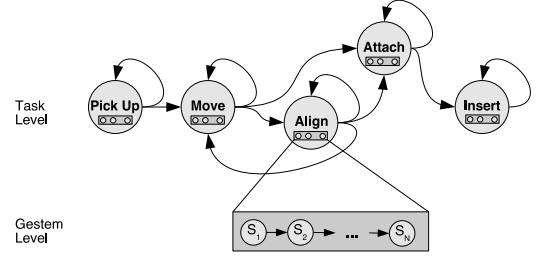


Fig. 2. A two level layered hidden Markov model, modeling gestemes at level 2 and a task at level 1.

environment such as the type of sensors, sampling rate etc, can be retrained separately without altering the higher layers of the LHMM.

Here, a LHMM with two levels are considered. At level 1 a single HMM is used to model the task, where each state in the HMM corresponds to a sub-task. At level 2 there is a HMM for each of the  $K_2$  possible gestemes that may occur during execution of the task. The observation sequence for the level 2 HMMs is generated from the quantized motion direction of the trajectory recorded during task operation. The index of the HMM with highest likelihood among the  $K_2$  HMMs at level 2 is then taken to be the the observation symbol for the level 1 HMM. The level 1 HMM is then used to compute the probability of a certain state as a function of time given the observation sequence produced by the HMMs at level 2. Since each state in the level 1 HMM corresponds to a mental stage of the teleoperation task this information can be used to understand the operator’s intention. The proposed structure is outlined in Fig. 2. Here, the winning HMM at level 2, i.e. the one with the highest likelihood, is chosen and an observation symbol corresponding to this gesteme is generated for the level 1 HMM. The alternative would be to use the complete probability distribution and have the HMMs at level 2 act as a probability estimator for the level 1 HMM. However, according to [14] using the complete distribution does not give any apparent advantage over the simpler winner takes all model.

#### A. The gesteme HMM

The goal of the gesteme HMMs is to distinguish between different motion primitives. For example, there can be gesteme HMMs to recognize motion along lines with different direction or circles with different radii and orientation in space. In our case, the gestemes can be any arbitrary motion in 2D or 3D. The observations for the gesteme HMMs are extracted from motion data. The trajectory is recorded, normalized and differentiated in order to compute the motion directions which are then mapped to corresponding observation symbols as described later in this section.

For the gesteme HMMs we evaluate the following types of models: **One-dimensional HMM (OD)**: Here, the observation symbols are taken from a set  $\mathcal{O} = \{O_1, O_2, \dots, O_K\}$  of  $K$  discrete symbols. The  $\mathbf{B}$  matrix is used to store the probability of observing the  $j$ th symbol in state  $i$ ,  $\mathbf{B}_{i,j} = P(O_j | \text{state } i)$ .

The symbols are generated by k-means clustering of all the training directions. The number of cluster centers is 25 in all experiments, if not stated otherwise. This number was chosen by an offline examination of the data. Using too few clusters makes it hard to distinguish between different motion directions while using too many makes the generalization difficult.

**Multi dimensional HMM (MD):** The MD HMM assumes independence between the different dimensions of the input data. Thus, there is a  $\mathbf{B}$  matrix for each dimension of the input data. This means that for a  $D$  dimensional HMM the observation symbols are also  $D$  dimensional where each dimension  $d$  contains values from a finite enumerated set. Each dimension is split into 10 equally sized bins and the input directions are projected into these bins generating the observation symbols.

**Multi dimensional HMM with FT (MD-FT):** MD-FT is similar to the MD except that instead of mapping the raw motion directions to symbols, each dimension of the raw input directions are pre-processed by applying the Fourier transform to small overlapping windows, similar to that reported in [7]. In this work a Hamming window, [16] of size 6 was used with 50% overlap.

### B. The Task HMM

The task HMM, or the level 1 HMM in the LHMM structure, encodes the task sequencing. Both levels of the LHMM work on the same time granularity and for each observation generated from the motion data the likelihood of the gesteme (level 2) HMMs are computed. The gesteme HMMs are enumerated and the index of the most likely gesteme HMM is used as an observation for the task level (level 1) HMM. Each of the states in the task level HMM corresponds to a sub-task in the operator's mental model and the most likely state can be computed in order to, for example, aid the operator with the execution of that sub-task. It should be noted that there need not be a one-to-one mapping between a state in the task level HMM and a gesteme HMM. Rather a specific gesteme can correspond to different states depending on the previous state (the Markov assumption). Furthermore there may be several gestemes that can appear in a single state.

As mentioned previously, the states of the task level HMM are supposed to correspond to the mental states of the operator. As a consequence, it is not possible to use the Baum-Welch algorithm to train the task level HMM, because it will optimize the HMM parameters  $\lambda$  in order to maximize  $P(Q|\lambda)$  for the state sequence  $Q$ . The approach taken in this work is to have the operator manually segment the trajectory into sub-tasks corresponding to the mental model of the operator. The gesteme HMMs are trained using the Baum-Welch algorithm. Using the gesteme HMMs to classify the training data a new observation sequence  $\mathbf{o}'$  is obtained. From the observation sequence  $\mathbf{o}'$  the  $\mathbf{B}$  can be computed by counting the occurrences of each symbol in every state and then normalizing the rows of  $\mathbf{B}$ . The task level HMM can now be trained by a modified

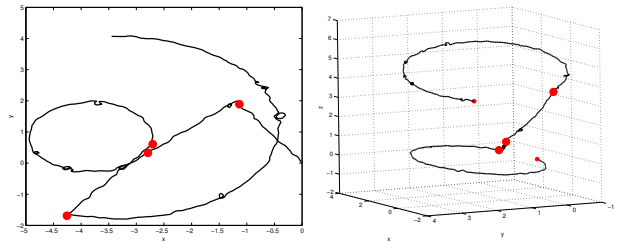


Fig. 3. Example trajectories in 2D (left) and 3D (right). The red dots marks the change from one primitive to the next.

version of the Baum-Welch algorithm where the  $\mathbf{B}$  matrix is kept constant.

## IV. EXPERIMENTAL EVALUATION

To better analyze and reproduce the results, we carry out experiments on synthetic data. The goal of the evaluation is to evaluate the three models: one-dimensional, multi-dimensional and multi-dimensional HMM with Fourier transform, with respect to the number of gestemes, the influence of the number of training samples, the effect of noise and the effect of the number of observation symbols. For the evaluation on real sensor data we refer to our previous work presented in [17].

A reference task consists of a sequence of motion primitives randomly generated from two groups of motion primitives. The first group contains straight lines of varying directions and lengths and the second group is made up of circle segments with varying starting and ending angles as well as orientations and radii. Fig. 3 shows example trajectories. These trajectory types may seem simple, but they were chosen because we believe that there exists several relevant tasks in areas such as medical surgery or automotive assembly that can be decomposed into a sequence of linear and circular motions.

The simulated trajectories are created in the following way. Given a reference trajectory  $T_r$  a target point  $\mathbf{p}$  is selected on  $T_r$  so that the distance to  $\mathbf{p}$  from the current position  $\mathbf{q}$  is larger than some threshold  $\xi$ . A direction of motion  $\mathbf{d}$  is then computed as the average between the direction towards  $\mathbf{p}$  from  $\mathbf{q}$  and the current direction of motion. A random error  $\mathbf{e}_d$  is then added to  $\mathbf{d}$  where each element of  $\mathbf{e}_d$  is generated independently according to  $\mathbf{e}_d(i) = \kappa \cdot \Gamma$ , where  $\Gamma$  is generated from a normal distribution ( $\mu = 1, \sigma = 1$ ) and  $\kappa$  determines the noise level. Finally the current position  $\mathbf{q}$  is updated by taking a step of size  $\delta \cdot (1 + 2\kappa \cdot \Gamma)$  in the direction of  $\mathbf{d}$  where  $\delta$  determines the step-size, which was set to 0.05 in all experiments. The value of  $\kappa$  was set to 0.15 for all experiments if not otherwise stated. Here, three classes of reference trajectories are used. They are referred to as *line*, *circle* and *mixed* trajectories in 2D respectively 3D. The line trajectories are made up of a sequence of linear segments, the circle trajectories are comprised of circle segments and the mixed trajectory type consists of a mixture of linear and circular segments.

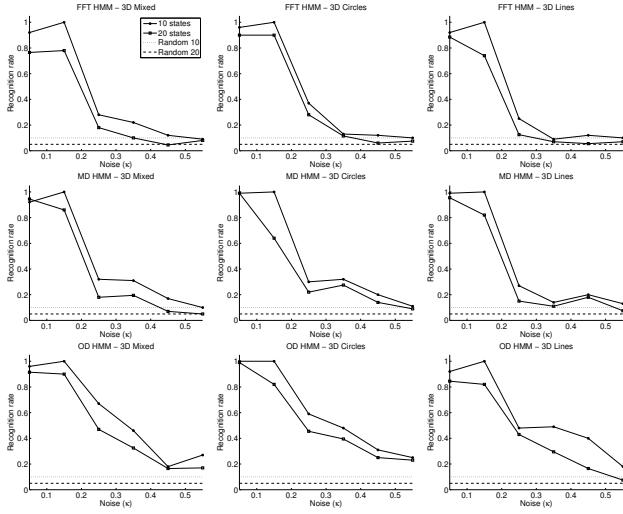


Fig. 4. Classification performance as a function of noise.

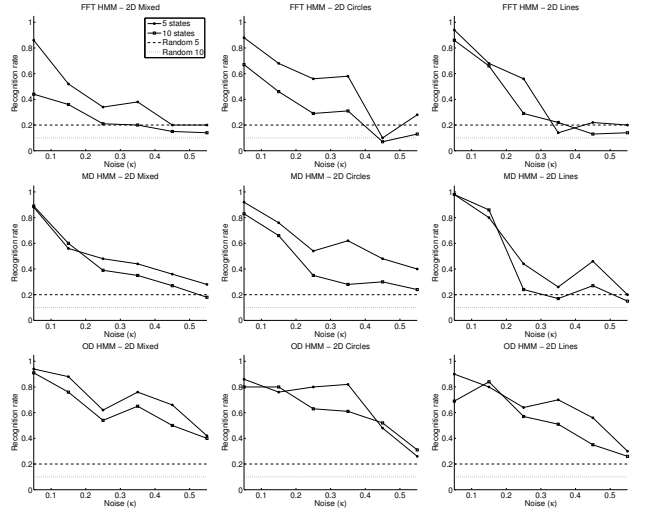


Fig. 5. Classification performance as a function of noise.

### A. The Gesteme Classifier

The HMM is able to handle a large amount of noise as long as the noise is consistent during training and classification. To evaluate what amount of noise the gesteme classifiers can handle, we tested the classification performance with several synthetic runs generated by varying the value of  $\kappa$  from 0.05 to 0.55. If the gestemes are not generated at random but chosen from some set of gestemes that are constructed to be easy to distinguish between (such as the letters of the alphabet) the performance could be expected to be better than that reported here. For the proposed methods to work in the intended setting it is required to obtain good results with only a limited amount of training samples. Therefore only five training samples were used for the experiments in this section, if not otherwise stated. Furthermore, the results presented in this section are the average of 10 independent trials, if not explicitly stated.

Fig. 4 and 5 shows the classification performance as a function of the noise,  $\kappa$ . We can conclude that a reasonable value for the noise parameter  $\kappa$  is less than 0.2 – 0.25. For the remainder of the experimental results on synthetic data the value of  $\kappa$  is therefore set to 0.15 unless explicitly specified. Note that a value of  $\kappa \in [0.3, 0.5]$  is almost as bad as guessing. By examining the individual runs, it can be seen that the noise sensitivity is highly affected by the similarity of the gestemes. If the gestemes are similar, the performance decreases almost linearly with increased noise. If the gestemes contain few common symbols, the classification performance remains relatively unaffected until the noise starts to dominate (i.e. is large compared to the nominal motion).

One interesting result is that the OD HMM appears to have better performance with respect to noise sensitivity. We believe that the reason for this is the low dimensionality and that the k-means clustering of the pre-processing step helps with generalization since the cluster centers are affected by the actual training data instead of using pre-defined bins.

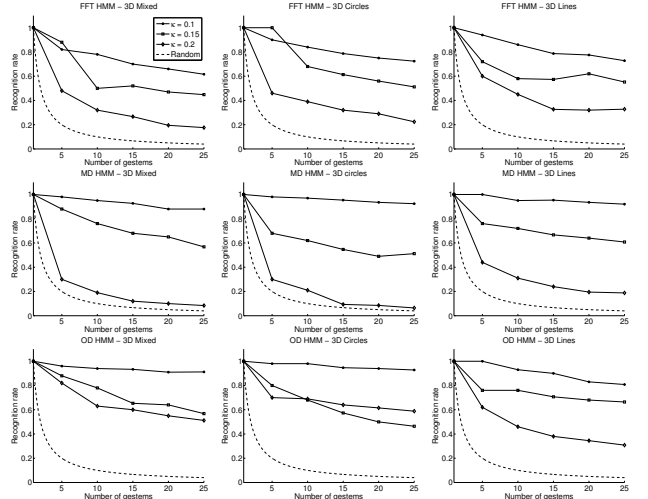


Fig. 6. Classification performance as a function of the number of gestemes.

The second experiment evaluates the effect of the number of gestemes on classification. For every gesteme there is a corresponding HMM which is trained to recognize it. As it can be seen in Fig. 6 and 7, the performance drops almost linearly from 100% to about 60% for 25 gestemes for the medium noise case where  $\kappa = 1.5$ . It is again interesting to note that the OD HMM appears to have better performance w.r.t noise. The classification performance for the 3D data is a bit better but that can be explained with the fact that the individual gestemes are less likely to be similar.

It is known that HMMs can be trained with only a small amount of data. Fig. 8 and 9 show that the recognition rate is quite high even for only two training runs. This is a good feature of the HMM gesteme classifier since in many settings extensive training is not possible. When the type of noise changes and outliers are introduced the necessary number of training sequences will increase in order to be able to capture the larger variations that occurs. However,

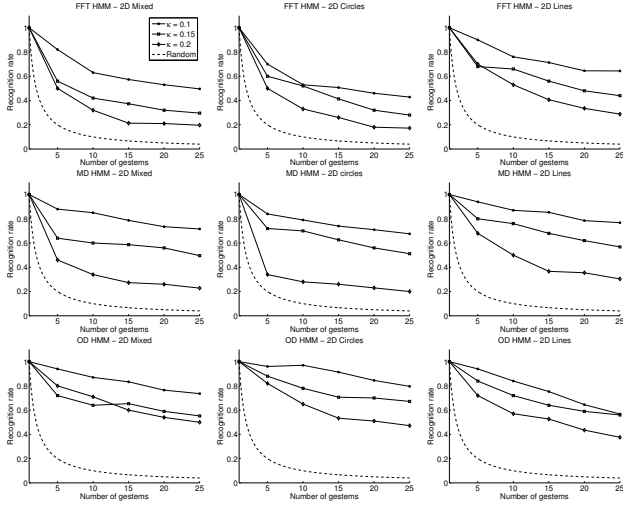


Fig. 7. Classification performance as a function of the number of gestemes.

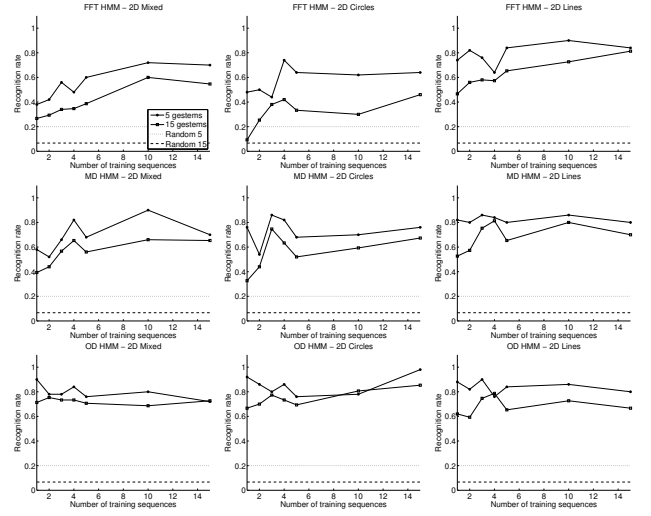


Fig. 9. Classification as a function of the number of training sequences.

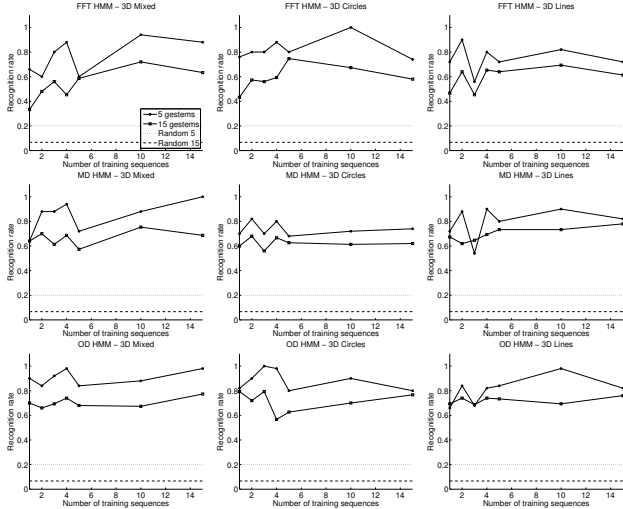


Fig. 8. Classification as a function of the number of training sequences.

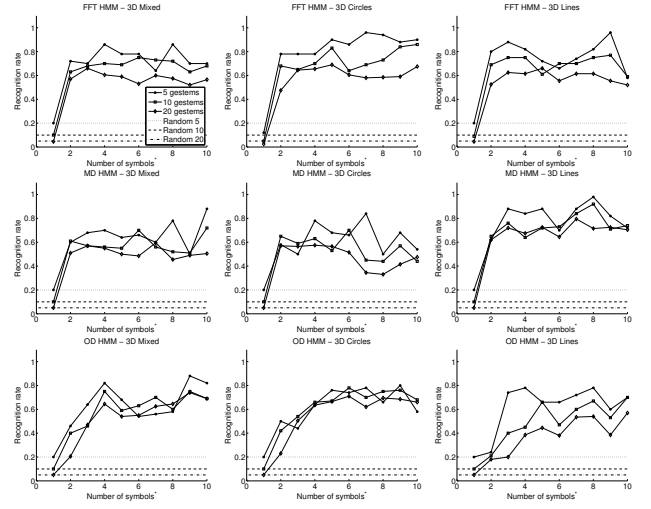


Fig. 10. Classification as a function of the number of symbols. Note that the number of symbols have different meaning for OD and MD HMMs.

preliminary results indicate that in practice the necessary number of training sequences is actually quite low as long as the training sequences are representative for what will occur during execution.

The number of observation symbols is not crucial but have to be set reasonably. If too few symbols are used the HMM can not distinguish between different directions leading to poor classification. At the same time, using too many symbols will prevent the HMM from generalizing, leading to poor classification because none of the models will correspond well with the training sequences. Fig. 10 and 11 show the classification performance as a function of the number of observation symbols. Remember that the observation symbols are defined differently between the OD and MD HMMs and values are thus not comparable. For the OD HMM the observation symbols correspond to the cluster centers obtain from the k-means clustering of the nominal motion directions of the training data, whereas for the MD HMMs the observation

symbols are taken from  $M \cdot D$  predefined bins of size  $1/M$  giving a total of  $M^D$  different possible observations, where  $M$  is the number of discrete observation symbols and  $D$  is the dimensionality of the MD HMM.

### B. The LHMM

Fig. 12 shows a 2D trajectories with 4 gestemes,  $G = \{l_1, l_2, l_3, c_1\}$ . The “mental model” of this task is that the gestemes should be performed in a sequential-left-to-right (SLR) fashion with the  $c_1$  gesteme appearing twice, the task having five different states  $S_1, \dots, S_5$ .

A task level HMM is now trained on the output of the gesteme classifiers. That is, the trajectory is classified by the gesteme classifiers (online) and the sequence of winning gestemes are used as input to the task-level HMM which is trained in order to extract the task-model. Fig. 12 (right) shows classification results. The dashed lines indicate the switch from one state to the next. Note that there are only four gestemes

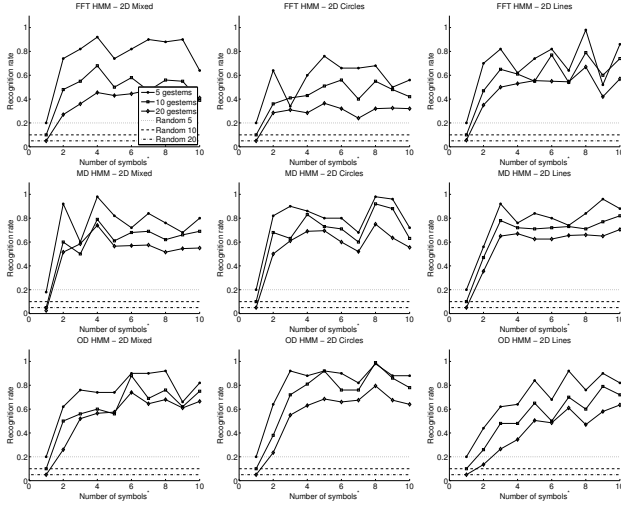


Fig. 11. Classification as a function of the number of symbols. Note that the number of symbols have different meaning for OD and MD HMMs.

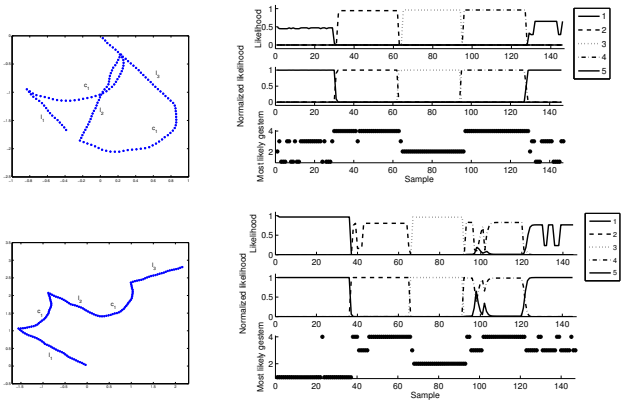


Fig. 12. Example tasks with 5 states and 4 gestemes.

recognized in the bottom plot whereas there is five states in the top and middle plots since the gesteme  $c_1$  is associated with two states. Even though the gesteme classifiers are sometimes confused, the task-level HMM is still capable of determining the correct state. This is because the misclassification of the gesteme classifiers are consistent with training data and thus the task-level HMM expects some misclassification. Furthermore the discriminant power of the LHMM is much better than that of the HMM, i.e. the difference between the most probable and the second most probable state is in general much larger for the LHMM.

## V. DISCUSSION AND CONCLUSION

In this paper, we have evaluate Layered Hidden Markov Models (LHMM) for motion intention recognition based on action-primitives or gestemes. Three different HMM models were used at the gesteme level: one-dimensional HMM, multi-dimensional HMM and multi-dimensional HMM with Fourier transform. These three models were evaluated with respect to the number of gestemes, the influence of the number of

training samples, the effect of noise and the effect of the number of observation symbols in both 2D and 3D tasks. One observation was that the OD HMM shows better performance with respect to noise sensitivity and we conclude that this is due to the low dimensionality and k-means clustering of the pre-processing step. We have also shown that the discriminant power of the LHMM is much better than that of the HMM since the difference between the most probable and the second most probable state is in general much larger for the LHMM.

## REFERENCES

- [1] C. S. Hundtofte, G. D. Hager, and A. M. Okamura, "Building a Task Language for Segmentation and Recognition of User Input to Cooperative Manipulation Systems," in *Proc. of the 10th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pp. 225–230, 2002.
- [2] M. A. Peshkin, J. E. Colgate, W. Wannasuphprasit, C. Moore, R. B. Gillespie, and P. Akella, "Cobot Architecture," *IEEE Trans. on Robotics and Automation*, vol. 17, pp. 377–390, 2001.
- [3] M. Li and A. Okamura, "Recognition of Operator Motions for Real-Time Assistance Using Virtual Fixtures," in *Proc. of the 11th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pp. 125–131, 2003.
- [4] D. Kragić, P. Marayong, M. Li, A. M. Okamura, and G. D. Hager, "Human-Machine Collaborative Systems for Microsurgical Applications," *Int. Journal of Robotics Research*, vol. 24, pp. 731–741, 2005.
- [5] A. Castellani, D. Botturi, M. Bicego, and P. Fiorini, "Hybrid HMM/SVM Model for the Analysis and Segmentation of Teleoperation Tasks," in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pp. 2918–2923, 2004.
- [6] D. Aarno, S. Ekvall, and D. Kragić, "Adaptive Virtual Fixtures for Machine Assisted Teleoperation Tasks," in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pp. 897–903, 2005.
- [7] W. Yu, R. Alqasemi, R. Dubey, and N. Pernalet, "Telemanipulation Assistance Based on Motion Intention Recognition," in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pp. 1121–1126, 2005.
- [8] R. Zöllner, O. Rogalla, R. Dillmann, and M. Zöllner, "Understanding Users Intention: Programming Fine Manipulation Tasks by Demonstration," in *Proc. of the Int. Conf. on Intelligent Robots and Systems*, pp. 1114–1119, 2002.
- [9] M. Kaiser and R. Dillmann, "Building Elementary Robot Skills from Human Demonstration," in *Proc. of the Int. Conf. on Robotics and Automation*, pp. 2700–2705, 1996.
- [10] R. H. Taylor and D. Stoianovici, "Medical Robotics in Computer-Integrated Surgery," *IEEE Trans. on Robotics and Automation*, vol. 19, pp. 765–781, 2003.
- [11] J. T. Nolin, P. M. Stemmiski, and A. M. Okamura, "Activation Cues and Force Scaling Methods for Virtual Fixtures," in *Proc. of the 11th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pp. 404–409, 2003.
- [12] P. Marayong, A. Bettini, and A. Okamura, "Effect of Virtual Fixture Compliance on Human-Machine Cooperative Manipulation," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 1089–1095, 2002.
- [13] S. Fine, Y. Singer, and N. Tishby, "The Hierarchical Hidden Markov Model: Analysis and Applications," *Mach. Learn.*, vol. 32, no. 1, pp. 41–62, 1998.
- [14] N. Oliver, A. Garg, and E. Horvitz, "Layered Representations for Learning and Inferring Office Activity from Multiple Sensory Channels," *Computer Vision and Image Understanding*, vol. 96, pp. 163–180, 2004.
- [15] D. Zhang, D. Gatica-Perez, S. Bengio, I. McCowan, and G. Lathoud, "Modeling Individual and Group Actions in Meetings: a Two-Layer HMM Framework," in *2nd IEEE Workshop on Event Mining: Detection and Recognition of Events in Video, In Association with CVPR*, 2004. IDIAP-RR 04-09.
- [16] F. J. Harris, "On the use of Windows for Harmonic Analysis with the Discrete Fourier Transform," vol. 66, pp. 51–83, 1978.
- [17] D. Aarno and D. Kragić, "Layered hmm for motion intention recognition," in *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS'06*, 2006.