Developing Context Sensitive HMM Gesture Recognition

Kingsley Sage
A. Jonathan Howell
Hilary Buxton

CSRP 563

April 2003

ISSN 1350–3162

Cognitive Science
Research Papers

Developing Context Sensitive HMM Gesture Recognition

Kingsley Sage, A. Jonathan Howell, and Hilary Buxton

School of Cognitive and Computing Sciences,
University of Sussex, Brighton BN1 9QH, United Kingdom.

Abstract. We are interested in methods for building cognitive vision systems to understand activities of expert operators for our ActIPret System. Our approach to the gesture recognition required here is to learn the generic models and develop methods for contextual bias of the visual interpretation in the online system. The paper first introduces issues in the development of such flexible and robust gesture learning and recognition, with a brief discussion of related research. Second, the computational model for the Hidden Markov Model (HMM) is described and results with varying amounts of noise in the training and testing phases are given. Third, extensions of this work to allow both top-down bias in the contextual processing and bottom-up augmentation by moment to moment observation of the hand trajectory are described.

1 Introduction

In cognitive science, it is well known that simple moving light displays contain sufficient information for meaningful interpretation of activity, as in the early work of Johansson [12]. Given this human ability, we use hand trajectory data alone for our gesture analysis here. Further, we use a definition of a gesture as a tri-phasic sequence of atomic hand movements as in our earlier work [10]. The essential elements of cognitive vision must be supported here: *memory* organisation of representations for the gestures; *reasoning* about these representations for flexible decisions and actions, including the resolution of ambiguity [17]; *control* of online visual processing for efficient interpretation [19]; and *learning* of both the task-relevant representations and how to use them [6].

Gestures, as above, require matching against their representations in the online system to find the interpretation or model that best explains the observation sequence. These representations have to account for the uncertain and probabilistic nature of gesture instances, including variations in the spatiotemporal evolution of the trajectories. These variations can be handled by techniques such as dynamic time warping in the matching, as in [4, 13]. However, the temporal templates required in this approach are subject to the problem of self-occlusion, as well as requiring the whole of the gesture to be completed before it can be recognised. The HMM approach, using the Viterbi algorithm to overcome the

problems of variability, is popular as it offers more sophisticated matching for many kinds of time-varying signals [15]. HMMs consist of a series of states, which can capture the intrinsic structure of our gestures from a set of training examples. They are further characterised by the probabilistic transitions between the states and the set of probabilities that a particular state gives rise to a particular observation. That is, they are a member of the class of generative models, as we can see from early work [8, 16].

HMMs also offer the advantage of unsupervised segmentation of continuous data streams, where the beginning and end is unknown. The Viterbi algorithm, mentioned above, is a kind of dynamic programming algorithm and is used to capture the maximum probability as well as the state sequence. This is of particular importance in sign language interpretation, which follows the lead of Starner and Pentland [18] to emphasise the continuous recognition of gestures [1]. However, the Viterbi algorithm requires calculation of observation probabilities for each state and time step in the Forward-Backward procedure, which is difficult to scale up. New avenues are being explored, based on the condensation algorithm [11], such as the incremental scheme proposed by Black and Jepson [2]. However, in this paper we are concerned with flexibility and robustness to noise, which affect how effective the gesture interpretation can be in a particular context, rather than its efficiency.

For an effective scheme, we need to consider the need to acquire the generic gesture models and contextually augmented versions of them, which can be used in the matching process above. Typically machine learning approaches have required hand labelling of the trajectory data or at least pre-segmentation into classes as in the TDRBF approach [10]. HMMs have an associated learning method, the Baum-Welch algorithm [15], which is a specific variant of the Expectation Maximisation (EM) algorithm [7]. In previous work [8], HMM trajectory prediction from entry regions through intermediate states to the re-fuelling or baggage-handling regions was augmented by updates on the position of vehicles from lower level vision for a known vehicle type. In general, scene context and aspects of the top-down interpretation or bottom-up visual information from moment to moment can be used to augment processing in an HMM without going to a full hierarchical Bayesian network [5]. In our recent work, additional context variables were introduced into the training data, as in [3]. These additional variables cause separate Gaussian components to be generated in feature space such that each context has an independent representation [6]. This work is extended here.

In the following, the HMM model is first formally described in section 2. Then in section 3, some results from the generalisation of the generic gesture models and the state-structure captured are described, together with considerations of how robust the recognition is under noise in training and testing trajectory sequences. In section 4, the extensions of the contextual processing are described, together with preliminary results from top-down contextual bias in the interpretation and on-line augmentation from bottom-up observations. Fi-

nally, in sections 5 and 6, the implications of the work for task control and system integration are then discussed with conclusions and suggestions for further work.

2 Hidden Markov Model for gesture recognition

A Hidden Markov Model (HMM) is a doubly stochastic process, i.e. there is an underlying stochastic process that is not observable (hidden) but can only be observed through another set of stochastic processes that produce the sequence of observed symbols [15]. The HMM is characterised by a triple $\lambda = (\pi, A, B)$ where A is a square $N * N$ matrix of probabilities for transitions between N discrete hidden states, π is a vector of probabilities describing the initial state of the model (at time $t = 0$) and B is a $N * M$ matrix accounting for the mapping between the N hidden states and the M output (observable) symbols.

There are three general problems we may solve using HMMs. Given a set of observation symbols O and a model λ we can calculate the probability of that sequence $p(O|\lambda)$ (forward evaluation). Given O and λ we can deduce the most likely sequence of hidden states (Viterbi decoding). Finally, and most relevant for what follows, given O we can estimate model parameters λ that maximise the probability of O. The most common form of HMM model parameter estimation is the Baum-Welch algorithm (described in [15]) which is an iterative non-globally optimal procedure for maximum likelihood estimation.

To capture gesture models, we use a continuous output HMM with training observation sequences represented as six-valued vectors (two sets of 3-D hand velocities – see next section) with the observation symbols modelled as 6-component mixture of Gaussian functions. We then vary the number of internal discrete hidden states to explore the underlying dimensionality of the training set (which corresponds approximately to the number of distinct gesture phases) and to demonstrate the ability of the HMM to distinguish the learned gesture from other gestures.

The gesture data used for the experiments in this paper was the *Terminal Hand Orientation and Effort Reach Study* Database created by Human Motion Simulation (HUMOSIM) at the Center for Ergonomics, University of Michigan, USA. 3-D hand trajectory data was collected from 22 subjects of varying gender, age, and height. Nineteen of the subjects were right-handed and two were left-handed. 210 target locations and hand orientations were used, giving a total number of 4,410 trials and the 8,820 reach movements.

This gesture data describes target and task orientated hand trajectories rather than communicative gestures (such as waving and pointing). HUMOSIM tasks represented are limited to the manipulation of targets at a number of positions and heights relative to the user.

Four towers were used in the HUMOSIM hand trajectory data, from 45° left of the subject to 90° right, each of which had three 'pods' as targets. There is further variation in the targets, as each of the pods has five cubes, each of which can use four hand orientations. For the experiments in this paper, we consider only tower/pod combinations (12 in all). Each trial produced a file of

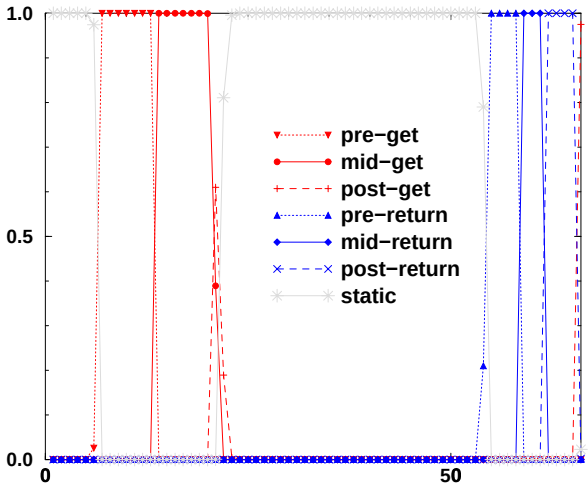


Fig. 1. Gesture phase classification for HMM trained with targets in Tower 0 (45° left of subject), when tested with a single complete hand trajectory also from Tower 0 values for output units for each gesture phase class (y-axis) at each timestep (x-axis). The probability values for each gesture phase correspond to the values associated with the HMM hidden nodes at each timestep.

3-D coordinates for two points (six values each time step) on the subject's hand. For each trial, data was collected at 25Hz for a sequence consisting of five distinct phases:

- Start with a static hand placed at a 'home location':
- A movement toward the target, which we term *get*;
- A static phase while the hand is at the target;
- A second movement, away from the target, which we term *return*;
- A final static phase at the home location.

Each resulting datafile contained 80–135 timesteps. The 3-D location data was pre-processed by differencing it from one time step to the next (relative motion or velocity data).

3 Results

We examined three aspects of HMM performance. We start by demonstrating how transitions between hidden states can be readily visualised facilitating direct comparison with other approaches. We then show how HMMs can generalise the gesture trajectory data, and how that generalisation degrades gracefully in the presence of noise. Section 4 shows how context information can be used to control HMM classifier performance.

We devised a simple technique based on modified Viterbi decoding to facilitate comparison between the transition between the HMM hidden states and six functional gesture phases defined for a comparable TDRBF model ([6]).

Analysis of the fit between the probability of the model parameters given the training data and the number of hidden states shows that the fit reaches an optimal point where low numbers of hidden states are matched against ability to generalise. In our particular task, this point was typically where the HMM had seven hidden states (similar to [14]).

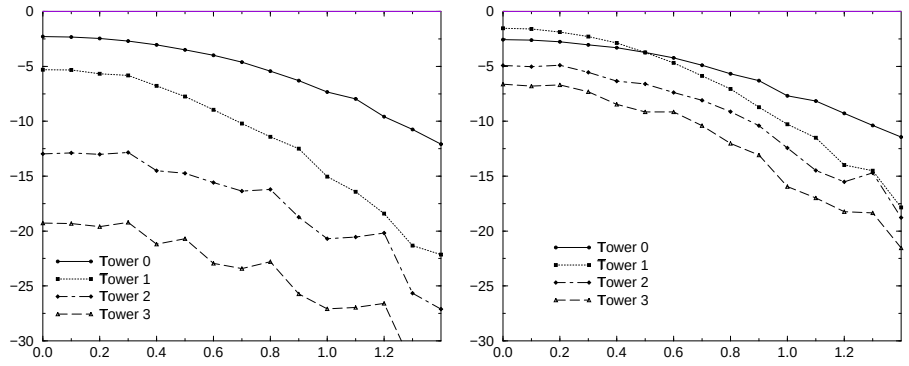
To test the trained HMM we presented complete trajectory files from targets not used for training. Fig. 1 shows the results for an HMM trained with 19 trajectories from a target on Tower 0 (45° left), when tested with another trajectory on the same tower. The seventh hidden state explicitly represents timesteps of minimal motion, interpreted as the 'static' gesture class.

3.1 Generalisation of the trajectory data

In the previous section we used the term 'classify' to refer to an analysis of the qualitative transitions between hidden states. In this section we make quantitative comparisons between different HMMs and so use the term classify to refer to a measure of fit between a learned model and a test observation. The usual quantitative measures used for HMMs are the terminal $P(O|\lambda)$ and the log likelihood per symbol *llps*. The measure of model used here is the log_{10} of the mean (non-log) likelihood per observation symbol, or *lmlpos*. This measure varies in the range $[0, -\infty)$ where 0 represents a perfect fit between the model and the testing data at each time step and $-\infty$ represents a zero fit. We use *lmlpos* rather than *llps* as the former represents the arithmetic mean of likelihood per symbol *lps* over a number of testing examples rather than taking an average of a set of log metrics which would constitute a geometric mean.

We wanted to explore the effect of noise on the trajectory data on classifier performance. We assumed that the original trajectory data represented a ground truth. In order to model noise in the data we superimposed independent but identically distributed Gaussian noise on each of the six data dimensions. We generated noise vectors with a given Root Mean Square (RMS) error. The RMS error corresponds to the standard deviation (i.e. square root of the variance). We then smoothed the noise vectors (average of current, previous and next timestep values) to provide a degree of temporal correlation and re-scaled the vectors to a given RMS error and then added these noise vectors to the trajectory data.

We first trained four separate HMMs, one each for trajectory data for each of the four towers with a constant pod setting and no training noise. We preprocessed the training data further slightly by removing timesteps where the sum of the absolute values across the six difference values was less than 1 cm. We then generated testing data by taking sections of the training data and superimposing testing noise at a given RMS level and classified this training data using the learned models. The results obtained using Tower 0 and Tower 1 as testing data are shown in Fig. 2(a) and (b) respectively.



(a) (b)

Fig. 2. Level of *lmlpos* fit for hand trajectories using HMM models for Towers 0-3 of as a function of RMS test set noise with test data for (a) Tower 0, (b) Tower 1.

Table 1. Classification rates for HMM models trained with various configurations of towers data when used to classify test towers data with varying amounts of noise.

Training Towers	No Test Noise, % Correct				0.6cm RMS Test Noise, % Correct			
	Tower 0	Tower 1	Tower 2	Tower 3	Tower 0	Tower 1	Tower 2	Tower 3
0	100	0	0	0	100	0	0	0
1	0	100	0	0	72	28	0	0
2	0	0	100	0	0	0	100	0
3	0	0	0	100	0	0	0	100
0 + 1	100	100	0	0	100	100	0	0
2 + 3	0	0	100	100	0	0	100	100

In Fig. 2(a) we see an excellent example of tower generalisation. At testing noise 0.0, we see that the Tower 0 model is, on average, the preferred model with the others models following in tower order at successively lower levels of confidence as measured by *lmlpos*. Similarly, in Fig. 2(b) at testing noise 0.0, we see that the Tower 1 model is, on average, the preferred model with the other models following in a logical order (with the tower 3 model being the least preferred).

As an alternative to measuring model fit using *lmlpos*, we can also measure the classification rate. That is, if we classify an individual test trajectory, we get a preferred model (the one that yields the highest *lmlpos*). If that highest value corresponds to the tower associated with the test trajectory then that is a 'correct' classification. For any set of hand trajectories we can therefore calculate a classification rate expressed as the percentage of test trajectories that are correctly associated with their target towers.

Table 1 shows the classification rates achieved with testing data set noise of 0.0 and 0.6cm RMS, both for single and composite tower arrangements. For

any given level of test set noise, generalisation is demonstrated as a diagonal classification rate matrix. Off-diagonal elements represent errors, in that the expected model was not the preferred model for some of the trajectory data. 100% classification is seen in all combinations, except Tower 1 with noise, where there is a degeneracy which causes misclassifications. This is shown in more detail in Fig. 2(b). The performance of the Tower 1 model can be seen to deteriorate more quickly than the other models in the presence of testing set noise. The reason for this is that Tower 1 was directly in front of the subjects meaning that the training data exhibits a very low variance along the x-axis. The Gaussian mixture model therefore has a low variance along that axis and so testing examples that deviate significantly from the Gaussian component mean (such as those created in the application of noise to generate training sets) produce a low probability membership estimate. This underlines the need to check for degeneracies in the application of the modelling process to the task.

The problem is easily overcome in a practical system by combining training data for towers 0 and 1 and tower 2 and 3 ([6]).

4 Top down and bottom up context control

In previous work [8], HMM trajectory prediction from entry regions through intermediate states to the re-fuelling or baggage-handling regions was augmented by updates on the position of vehicles from lower level vision for a known vehicle type. In general, scene context and aspects of the top-down interpretation or bottom-up visual information from moment to moment can be used to augment processing in an HMM without going to a full hierarchical BBN or DBN. In our case, additional context variables could be introduced into the training data (for example, to indicate the target tower) [3]. These additional variables would cause separate Gaussian components to be generated in feature space such that each context would have an independent representation. These context variables are abstract in the sense that there is no difference in implementation between top down and bottom up control.

In order to demonstrate context control using HMMs, we selected four trajectory data sets, one each for the four towers with constant values for pod and cube.

4.1 Using a single independent context control variable

For this experiment we augmented the six value vectors generated for relative position with a single context value (a pseudo probability value). This single independent context value represents a gesture-context relationship with two context classes. $context_1$ is the value specified in each vector and $context_2$ is implied as $1 - context_1$. We assume that this context is provided by an external agent (perhaps an object classifier) but for this experiment the training context value is generated directly from a normal distribution about a mean, with a single context value generated for each time step.

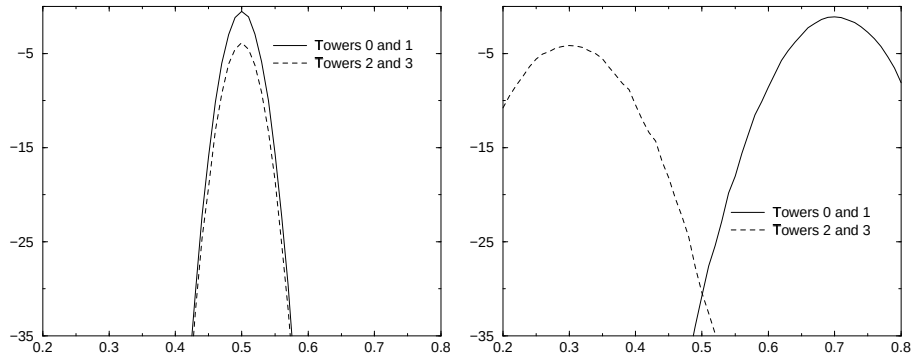


Fig. 3. Classification with HMM of tower 0 and 1 trajectories as a function of context where training context (a) $context_1 = context_2 = 0.5$, (b) $context_1 = 0.7$ and $context_2 = 0.3$.

We then grouped Towers 0 and 1, and 2 and 3 together and trained 2 composite HMMs using the same seven hidden state structure as before. We then classified the Towers 0 and 1 data using the two models and plotted a measure of model fit as a function of the context value.

For the first test, we set the mean $context_1 = context_2 = 0.5$. The results are shown in Fig. 3(a), the y-axis showing the *lmlpos* value and the x-axis $context_1$. We see that for all test values of context that the model for Towers 0 and 1 is preferred over the model for Towers 2 and 3 but that the confidence of that fit is maximised where the testing context matches the training context and falls away either side of that value. To see the real value of context control, we then set the mean $context_1 = 0.7$ and $context_2 = 0.3$. The results for this are shown in Fig. 3(b). This time we see that when classifying the Tower 0 and 1 examples, the Towers 0 and 1 model is preferred when the context is around 0.7, but that as the context approaches 0.3, the model for Towers 2 and 3 is preferred, albeit at a lower level of confidence.

4.2 Using multiple independent context control variables

For this experiment we augmented the six value vectors generated for relative position with two independent pseudo probabilistic context values $context_1$ and $context_2$. As before, we assume that these contexts are provided by an external agent but for this experiment the training context values are generated directly from a normal distribution about a mean, with two context values generated for each time step. We then used the same towers training grouping as for the previous experiment and classified the Towers 0 and 1 data using the two models and plotted the measure of model fit as a function of the context.

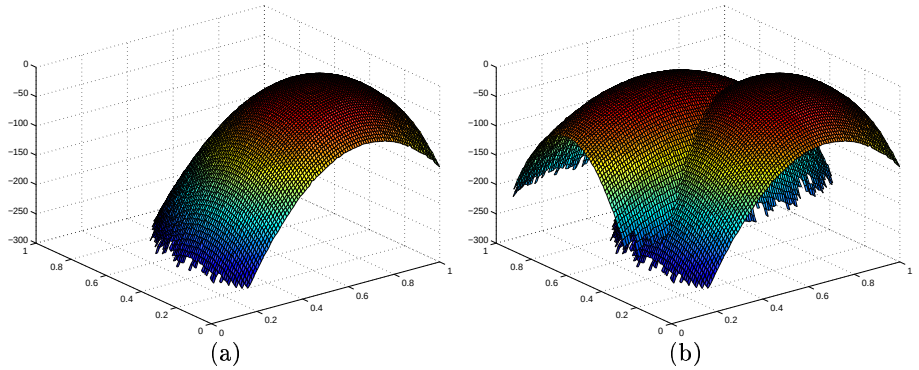


Fig. 4. Classification with HMM of tower 0 and 1 trajectories as a function of context where (a) Towers 0 and 1 training $context_1 = 0.7$ and $context_2 = 0.3$, (b) Towers 0 and 1 as for (a) and Towers 2 and 3 $context_1 = 0.4$ and $context_2 = 0.6$.

For the first test we trained the Towers 0 and 1 model with a mean $context_1 = 0.7$ and $context_2 = 0.3$. We then simply plotted the Towers 0 and 1 test data as a function of context. The results are shown in Fig. 4(a), the z-axis showing the *lmlpos* value, the x-axis $context_1$ and the y-axis $context_2$. We see that the model fit forms a continuous surface with a peak at the training context values.

For the second test, we retained the previous Towers 0 and 1 model and trained the Towers 2 and 3 model with a mean $context_1 = 0.4$ and $context_2 = 0.6$. We then plotted both models as a functions of context. We then classified the Towers 0 and 1 test data against both models and the results are shown in Fig. 4(b). We can see that the two surfaces intersect along a classification boundary that defines the values of context at which a switch in model preference takes place. As before, the peak of the Towers 2 and 3 surface (maximum fit) is lower than for the Towers 0 and 1 surface, as it represents a greater degree of generalisation.

5 Summary

We have demonstrated that:

- Modified Viterbi decoding can be used in some circumstances to extract meaning from a HMM forward evaluation trellis in terms of transitions between discrete functional gesture phases, facilitating direct comparison with other model types.
- HMMs are well suited to building gesture models with graceful deterioration in classification performance with increasing parameter distance from the learned model prototype and noise.
- Context control variables can be added at the learning stage to build HMM classifiers that are context sensitive. Such context control can be either top down or bottom up. In either case further context qualifier parameters (such

as confidence) could be used to determine model performance when context data is either unavailable of limited quality (e.g. a very early estimate from a predictive cueing process). However there is a trade-off between improvements in performance through context control and the size of the training set.

6 Conclusion

The HMM based learning can discover the temporal structure of the 3D hand gesture trajectories here from data clustering alone. The association of different interpretations with different contexts is also learnt and can allow more effective discrimination boundaries in the online system. Performance on the learning and generalisation tasks were robust to noise and scale well with task complexity, however the training with the Baum-Welch algorithm does not scale so well. The HMM developed here was coded in Matlab, thus it is premature to give computational costs but these will be established in future work. As in the discussion above, there is great potential for contextual processing using the HMM for attentive processing in the ActIPret system.

We have also proposed an approach to task control within the ActIPret system using a Dynamic Decision Network (DDN), e.g.. [9], in the Activity Reasoning Engine. However, we also want distributed control in the lower levels and one way of imposing this is by conditional probability matrices to activate the processing within each lower component, using priority metrics. Initially, it is proposed to hand code utility/task relevance nodes (e.g. watch/ignore) that determine the priority metric. In the longer term, in the context of a complete system, we hope to learn these dynamic dependencies. It may be that we can determine task-relevance automatically in this way, using a uniform Bayesian approach.

Acknowledgements

The authors gratefully acknowledge the invaluable help provided by the Laboratory for Human Motion Simulation (HUMOSIM) at the University of Michigan, USA in allowing us access to their 'Terminal Hand Orientation and Effort Reach Study, 2000' hand trajectory database; also framework concepts and funding from the EU ActIPret project.

References

1. B. Bauer, H. Heinz, and K. Kraiss. Video-based continuous sign language recognition using statistical methods. In *IEEE International Conference on Automatic Face and Gesture Recognition*, pages 440–445, Grenoble, France, 2000.
2. M.J. Black and A. Jepson. A probabilistic framework for matching temporal. In *European Conference on Computer Vision*, pages 343–356, Cambridge, UK, 1996.

3. D. M. Blei and P. J. Moreno. Topic segmentation with an aspect hidden Markov model. In *Proceedings of the 24th International Conference on Research and Development in Information Retrieval (SIGIR 2001)*, pages 343–348, New York, 2001.
4. A. Bobick and J. Davies. The recognition of human movement using temporal templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23:257–267, 2001.
5. H. Buxton and S. Gong. Visual surveillance in a dynamic and uncertain world. *Artificial Intelligence*, 78:431–459, 1995.
6. H. Buxton, A.J. Howell, and K. Sage. The role of task control and context in learning to recognise gesture. In *Cognitive Vision Workshop*, Zürich, Switzerland, 2002.
7. A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from imcomplete data using the em algorithm. *Journal of the Royal Statistical Society*, 39:185–197, 1977.
8. S. Gong and H. Buxton. On the visual expectations of moving objects: A probabilistic approach with augmented hidden Markov model. In *European Conference on Artificial Intelligence*, pages 781–785, Vienna, Austria, 1992.
9. R. Howarth and H. Buxton. Conceptual descriptions from monitoring and watching image sequences. *Image and Vision Computing*, 18:105–135, 2000.
10. A.J. Howell and H. Buxton. Learning gestures for visually mediated interaction. In *British Machine Vision Conference*, pages 508–517, Southampton, UK, 1998.
11. M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. In *European Conference on Computer Vision*, pages 343–356, Cambridge, UK, 1996.
12. G. Johansson. Visual perception of biological motion and a model for its analysis. *Perception and Psychophysics*, 14:201–211, 1973.
13. S. McKenna and S. Gong. Gesture recognition for visually mediated interaction using probabilistic event trajectories. In *British Machine Vision Association*, Southampton, UK, 1998.
14. D. J. Moore, I. A. Essa, and M. H. Hayes. Exploiting human actions and object context for recognition tasks. In *IEEE International Conference on Computer Vision*, pages 80–86, Vancouver, Canada, 1999.
15. L.R. Rabiner. A tutorial on hidden Markov models. *Proceedings of the IEEE*, 77:257–286, 1989.
16. J. Schlenzig, E. Hunter, and R. Jain. Recursive indentification of gesture using Hidden Markov Model. In *Workshop on Applications of Computer Vision*, Sarasota, Florida, 1994.
17. J. Sherrah and S. Gong. Resolving visual uncertainty and occlusion through probabilistic reasoning. In *British Machine Vision Association*, Bristol, UK, 2000.
18. T. Starner and A. Pentland. Real-time american sign language recognition using hidden Markov models. In *International Symposium on Computer Vision*, pages 265–270, Coral Gables, FL, 1995.
19. M. Walter, A. Psarrou, and S. Gong. Data driven model acquistion using minimum description length. In *British Machine Vision Association*, Manchester, UK, 2001.