

# Learning Behaviour Models of Human Activities

Aphrodite Galata, Neil Johnson and David Hogg  
School of Computer Studies, University of Leeds, Leeds LS2 9JT, UK  
{afro, neilj, dch}@scs.leeds.ac.uk      <http://www.scs.leeds.ac.uk/vision/>

## Abstract

In recent years there has been an increased interest in the modelling and recognition of human activities involving highly structured and semantically rich behaviour such as dance, aerobics, and sign language. A novel approach is presented for automatically acquiring stochastic models of the high-level structure of an activity without the assumption of any prior knowledge. The process involves temporal segmentation into plausible atomic behaviour components and the use of variable length Markov models for the efficient representation of behaviours. Experimental results are presented which demonstrate the generation of realistic sample behaviours and evaluate the performance of models for long-term temporal prediction.

## 1 Introduction

In recent years, challenging problems such as human-computer interaction, automated visual surveillance and the realistic animation of human motion, have led to an increased interest in providing machines with the ability to learn and use models of human behaviour [14, 3, 13, 8]. Of particular interest is the modelling and recognition of human activities involving highly structured and semantically rich behaviour such as dance, aerobics, and sign language [6, 17, 18].

In this paper an activity is viewed as a sequence of primitive movements with a high-level structure controlling the temporal ordering. Others have used a similar approach to perceiving human activities. Bobick and Ivanov [4] used a context-free parsing mechanism together with HMMs modelling low-level behaviour primitives for the recognition of activities, using a hand-coded stochastic context-free grammar to represent *a priori* knowledge of the high-level structure of an activity. Bregler [5] used a framework for the probabilistic decomposition of human dynamics at different levels of abstraction, modelling complex gestures as successive phases of simple movements using an HMM.

Unfortunately, HMMs do not encode high order temporal dependencies easily. Local optima are frequently encountered by iterative optimisation techniques when learning HMMs with many free parameters, and thus model topology and size are often highly constrained prior to training. We propose the use of variable length Markov models (VLMM) [15, 9] as a simple, yet powerful and efficient mechanism for capturing behavioural dependencies and constraints.

Two approaches are described for modelling behaviour using VLMMs at different temporal scales and the learnt behaviour models are used to demonstrate both the generation of realistic sample activities and the robust prediction of future behaviour.

## 2 Augmented configuration space

Behaviours are viewed as smooth trajectories in an *augmented configuration space* which describes both  $d$ -dimensional object configuration  $\mathbf{C}_t$  and its first derivative  $\dot{\mathbf{C}}_t$ . The inclusion of derivatives helps resolve ambiguity and facilitates the use of the models in the performance of generative tasks.

The first stage of the modelling process involves the acquisition of sequences,  $F_j$ , of regularly sampled augmented configuration vectors,  $\mathbf{F}_t \in [0, 1]^{2d}$ . Each sequence describes the temporal evolution of a behaviour:

$$F = \{\mathbf{F}_0, \mathbf{F}_1, \dots, \mathbf{F}_m\}, \quad (1)$$

where

$$\mathbf{F}_t = (\mathbf{C}_t, \lambda \dot{\mathbf{C}}_t), \quad (2)$$

and  $\lambda$  balances the contribution of derivatives when using the Euclidean distance as a dissimilarity measure.

In order to generate a discrete representation of behaviours, each augmented configuration vector  $\mathbf{F}_t$  is replaced by the nearest prototype  $\mathbf{p}_i$  from a finite set  $P = \{\mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_n\}$  of prototypical augmented configurations. Prototypes are derived using robust vector quantisation (see Johnson and Hogg [13] for details), and are further sub-sampled to produce a more uniform distribution of prototypes over the space of observed behaviours. Each behaviour is therefore represented by a sequence of prototypes  $\{\mathbf{p}_{i_0}, \mathbf{p}_{i_1}, \dots, \mathbf{p}_{i_m}\}$ .

### 2.1 Experimental data

For the experiments described in Sections 3 and 4, individuals performing exercise routines were tracked using a simple contour tracker [12, 1]. Object configuration is represented by the  $n$  control points of a closed uniform B-spline approximating the silhouette boundary. Control points are evenly spaced around the silhouette and ordered relative to a consistent point of reference (Figure 1(a)).

Integration of data from different training sequences is enabled by transforming control points into object centred coordinates and normalising such that each component of the transformed control points lies in the interval  $[0, 1]$ . The evolving silhouette boundary is thus represented by configuration vectors  $\mathbf{C}_t = (x_1(t), y_1(t), \dots, x_n(t), y_n(t)) \in [0, 1]^{2n}$  (i.e.  $d = 2n$ ).

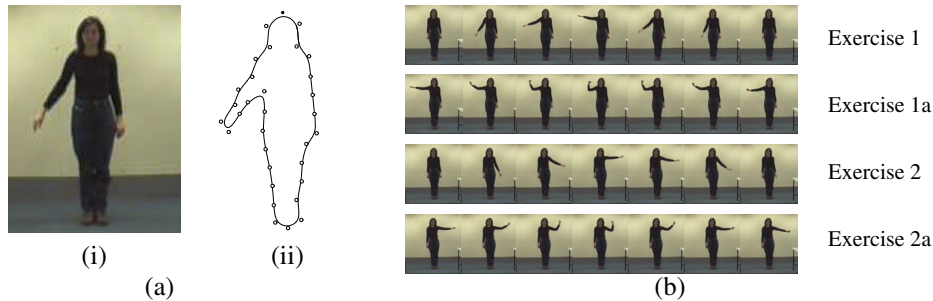


Figure 1: (a) Spline-based shape representation (b) Structure of exercise routine.

Training data was generated from a single sequence of an individual performing an exercise routine. This exercise routine comprises two exercises, each repeated four times and followed by four repetitions of a sub-exercise (see Figure 1(b)). The shape was represented by  $B$ -splines with 32 control points resulting in 64-dimensional configuration vectors  $\mathbf{C}_t$ . The training set  $F^{\text{shape}}$  of 128-dimensional augmented configuration vectors  $\mathbf{F}_t$  was generated using a scaling factor  $\lambda = 10$  and a set  $P^{\text{shape}}$  of 71 prototypes in the augmented configuration space was learnt from the training data set.

### 3 Modelling behaviour using variable length Markov models

We are interested in building models of behaviour which are able to support both recognition and generative capabilities such as the prediction of future behaviours or the generation of realistic sample behaviours. Since the description of behaviours as ordered sequences of prototype vectors in the augmented configuration space is a discrete representation, the addition of generative capabilities can be achieved during a further learning phase in which memory conditioned probabilities of transitions between prototypes are estimated using a variable memory length Markov model (VLMM) [9].

#### 3.1 Variable length Markov models

VLMMs have been used successfully for text compression [7, 2] and more recently in language modelling to improve the accuracy of speech and handwriting recognisers [15, 9, 10]. Their advantage over a fixed memory Markov model is the ability to locally optimise the length of memory required for prediction. This results in a more flexible and efficient representation which is particularly attractive in cases where we need to capture behavioural dependencies at a large temporal scale.

Assume  $w$  is a string of tokens used as a memory to predict the next token  $a'$  according to an estimate  $\hat{P}(a'|w)$  of  $P(a'|w)$ . The main idea behind the variable length modelling method is that if the output probability  $\hat{P}(a'|aw)$  that predicts the next token  $a'$  is significantly different from  $\hat{P}(a'|w)$ , then the longer memory  $aw$  may be a better predictor than  $w$ . The Kullback-Leibler divergence [15] is used to measure the additional information that is gained by using the longer memory  $aw$  for prediction instead of the shorter memory  $w$ :

$$\Delta H(aw, w) = \hat{P}(aw) \sum_{a'} \hat{P}(a'|aw) \log \frac{\hat{P}(a'|aw)}{\hat{P}(a'|w)}. \quad (3)$$

If  $\Delta H(aw, w)$  exceeds a given threshold  $\epsilon$ , then the longer memory  $aw$  is used, otherwise the shorter memory  $w$  is considered sufficient for prediction.

The transition probabilities and priors are derived from estimates of  $P(a_n|a_1a_2 \dots a_{n-1})$  and  $P(a_1a_2 \dots a_n)$  calculated for various values of  $n$  ( $n = 1, 2, \dots, N$ ). The estimates are given by:

$$\hat{P}(a_n|a_1a_2 \dots a_{n-1}) = \frac{v(a_1a_2 \dots a_{n-1}a_n)}{v(a_1a_2 \dots a_{n-1})}, \quad (4)$$

$$\hat{P}(a_1a_2 \dots a_n) = \frac{v(a_1a_2 \dots a_n)}{v_0}, \quad (5)$$

where  $v(a_1 a_2 \dots a_n)$  is the number of times the string of tokens  $a_1 a_2 \dots a_n$  appears in the training data and  $v_0$  is the total length of the training sequences.

The training algorithm involves building a *prefix tree* [9] where each node corresponds to a string up to a predetermined length  $N$ . The transition frequencies are counted by traversing the tree structure repeatedly with strings of length  $N$  where the strings are generated by sliding a window of fixed length  $N$  along a training sequence of tokens. Transition probabilities are computed using equation 4 and a pruning procedure is then applied while the prefix tree is converted to a *prediction suffix tree* [15]. For each node  $n_p$  in the prefix tree, a corresponding node  $n_s$  in the suffix tree is created if and only if the Kullback-Leibler divergence between the probability distribution at  $n_p$  and the probability distribution at its ancestor node in the prefix tree is larger than threshold  $\epsilon$ . Finally, the suffix tree is converted to an automaton representing the trained VLMM. A more detailed description on building and training variable length Markov models is given by Ron *et al.* [15].

Thus, a VLMM is equivalent to a *Probabilistic Finite State Automaton* (PFSA) represented by  $M = (Q, \Sigma, \tau, \gamma, \pi)$  where  $\Sigma$  is a finite *alphabet* –the set of tokens– and  $Q$  is a finite set of model states. Each state corresponds to a token string of length at most  $N$ , ( $N \geq 0$ ), representing the *memory* for a conditional transition of the VLMM. The *transition function* is  $\tau : Q \times \Sigma \rightarrow Q$  and  $\gamma : Q \times \Sigma \rightarrow [0, 1]$  is the *output probability function* representing the memory conditioned probabilities of the next *token*  $a \in \Sigma$ . Finally,  $\pi : Q \rightarrow [0, 1]$  is the probability distribution over the *start states*. The functions  $\gamma$  and  $\pi$  are such that for every  $q \in Q$ ,  $\sum_{a \in \Sigma} \gamma(q, a) = 1$  and  $\sum_{q \in Q} \pi(q) = 1$ .

### 3.2 Modelling behaviour at the prototype level

In our first approach, temporal dependencies in behaviour are represented by a VLMM over the prototypes. The training sequences  $F_j$  are converted into sequences of prototypes by observing the closest prototype, in a *nearest neighbour* sense, to the current training vector at each time instant. Only transitions between different prototypes are considered for training. The output sequences are used to train a VLMM represented by the PFSA  $M_p = (Q_p, P, \tau_p, \gamma_p, \pi_p)$ .

#### 3.2.1 Behaviour Generation

The trained model  $M_p$  represents the learnt behaviour and has generative capabilities. Behaviour generation is achieved by traversing the PFSA, selecting either the most likely transition (maximum likelihood behaviour generation) or sampling from the transition distribution (stochastic behaviour generation) at each state, and emitting the corresponding prototype vectors. This results in an ordered set  $G$  of prototype vectors  $\mathbf{p}_{q_r}$  which are the output of the transitions  $q_{r+1} = \tau_b(q_r, \mathbf{p}_{q_r})$  between states  $q_r, q_{r+1} \in Q_p$ .

The time interval between generated prototypes is initially unspecified and in order to generate an output sequence of vectors in the augmented configuration space at video frame rates, an interpolant of  $G$  must be sampled. Since non-linear changes may occur between prototypes separated by large time intervals, an approximation for the time interval is found and a Hermite interpolation is used (see [12] for details). An ordered set of vectors in the augmented configuration space can be produced by sampling this interpolant at data frame rate.

### 3.2.2 Prediction of future behaviour

To use the model  $M_p$  for behaviour prediction, it is first necessary to locate the current model state. Since model states may encode a history of previous behaviour, the model is initially used in a recognition mode, accepting successive prototypes representing observed behaviour and making the corresponding state transitions. Having located the current model state, prediction of future behaviour can be achieved using the model either as a stochastic or a maximum likelihood behaviour generator.

A VLMM model needs to be able to handle the problem of *unseen events* – cases where token sequences which might have not appeared previously in the training corpus, appear during the prediction process. Therefore, if the model  $M_p$  is presented with a prototype  $\mathbf{p}_i \in P$  while is in a state which emits this prototype with probability zero, we return to the initial model state, lose all the previous memory and predict  $\mathbf{p}_i$  with the prior probability  $\hat{P}(\mathbf{p}_i)$ . This *backing-off* method is simple but effective, although other more complex methods of handling unseen events could be employed [11].

### 3.2.3 Assessing predictor performance

For each prediction model a set of root mean square (RMS) errors are calculated to quantify the mean performance in predicting the value of the output vector in configuration space on each future time instant:

$$E_T = \sqrt{\frac{\sum_{j=1}^n |\mathbf{C}_{t+T} - \mathbf{C}_{t+T}^*|_j^2}{n}}, \quad (6)$$

where the error in predicting forward by  $T$  time steps is averaged over predictions generated on every frame of every test sequence and  $\mathbf{C}_{t+T}^*$  denotes the *ground truth* vector in configuration space at time  $t+T$  as given by the test data. Errors are only updated if both a prediction and the ground truth exist for the particular  $T$ .

The mean performance of the models should represent a probabilistic weighting of the errors given by all possible predictions. In general, it is not possible to enumerate the entire set of possible predictions from a particular model state due to the possibility of cycles within the transition structure. Instead, mean performance is calculated using Monte Carlo simulation, generating a large number of stochastic predictions on each frame and allowing their relative frequency to provide probabilistic weighting within the calculation of RMS errors. For the experiments presented in this paper, 50 stochastic predictions were generated on each frame.

### 3.2.4 Experiments

Using the set of 71 prototypes  $P^{\text{shape}}$  (see Section 2.1) in the augmented configuration space as an alphabet, variable length Markov models have been trained to capture activity behaviour for different values of memory length  $N$ . A test sequence of an individual performing an exercise routine is used to demonstrate predictor performance. The test sequence comprises the same two exercises and sub-exercises as the ones in the training data but in this case each exercise and sub-exercise is repeated three times.

Figure 2 demonstrates predictor performance of the variable length behaviour model  $M_p$  for various values of  $N$  where  $N$  is the maximum possible length of memory for each model state. For  $N$  equals 4, 14 and 19 a VLMM model with 113, 232 and 265 states

## BMVC99

respectively was learned using in all cases a value  $\epsilon = 0.0001$ . Each plot in the graph demonstrates mean predictor performance over a range  $t + T$  ( $1 \leq T \leq 70$ ) of future time instants, averaged over all stochastic predictions for all frames in the test data set.

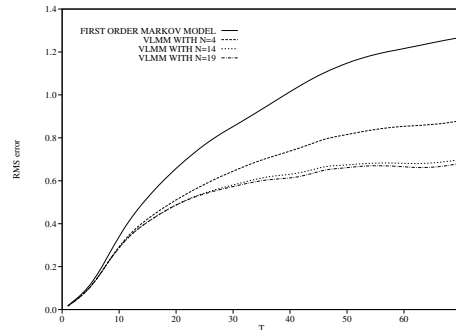


Figure 2: Behaviour prediction results using prototypes as alphabet.

As can be seen from the prediction graph, substantially better results are obtained in comparison to a first order Markov model. Learning temporal behaviour dependencies at the prototype level in the augmented configuration space using a variable length Markov model results in an efficient behaviour model representation that captures accurately local behaviour dependencies and has good generative capabilities. However, in order to capture more of the large scale structure of an activity and thus infer possible higher level syntactic information, a hierarchical approach in behaviour modelling as described in the next section can yield better results.

## 4 Learning structured behaviour models

Motion in human activities has different characteristics at different time scales, usually carrying syntactic and semantic information at larger temporal scales. Capturing the variability of behaviour at different temporal scales could be facilitated by using multiple memory mechanisms, thus providing a more powerful means of modelling structured and semantically rich behaviours of human activities.

In this section a hierarchical memory mechanism is proposed that has a detailed history for recent past behaviour together with a history of important states or simple movements performed during the long term past behaviour. Using such a mechanism, it is possible to capture the larger scale temporal dependencies and therefore infer possible high level syntactic or semantic information about the studied behaviour.

### 4.1 Temporal segmentation

Temporal segmentation into plausible atomic movements involves identifying suitable break points at which to partition an activity. Given the physical constraints posed by the human body, any change in the type of human movement usually causes dips in velocity. We wish to exploit this by using minima in the magnitude of configuration change as clues for performing semantic temporal segmentation.

Activity behaviour, as described in Section 2, is represented by a sequence of prototypes in the augmented configuration space, describing the temporal evolution of behaviour. We use the magnitude of the first derivative  $\dot{\mathbf{C}}_t$  of a prototypical object configuration to identify a subset of prototypes that could represent suitable segmentation points of activity behaviour in the augmented configuration space. We call this the set of *key prototypes*  $K$ :

$$K = \{\mathbf{k}_1, \mathbf{k}_2, \dots, \mathbf{k}_{k-1}, \mathbf{k}_k\} \subset P. \quad (7)$$

A threshold value is chosen by inspection and prototypes with first derivative magnitude below this threshold are chosen as key prototypes. These are then utilised to facilitate the learning of larger scale temporal dependencies in activity behaviour.

#### 4.1.1 Atomic behaviour components

Having identified a set of key prototypes, a set of *atomic behaviour components*  $\alpha_{ij}$  is acquired, where each  $\alpha_{ij}$  represents the range of behaviour observed between key prototypes  $\mathbf{k}_i$  and  $\mathbf{k}_j$  for  $i \neq j$ . Each atomic behaviour component  $\alpha_{ij}$  comprises a set of  $m$  *template sequences*  $\alpha_{ij}^l$ ,  $1 \leq l \leq m$ , where  $m$  is, in general, different for each atomic behaviour component. Each such template is an ordered set of  $n$  augmented configuration prototypes

$$\alpha_{ij}^l = \{\mathbf{p}_{1l}, \mathbf{p}_{2l}, \dots, \mathbf{p}_{nl}\}, \quad (8)$$

where  $n$  is, in general, different for each template sequence.

Template sequences are generated from the training corpus via a process of sequence clustering and merging. Initially, the set of all training sub-sequences spanning the transition between prototypes  $\mathbf{k}_i$  and  $\mathbf{k}_j$  is acquired. This set is then partitioned into a number of clusters of self-similar sequences using dynamic time warping [16] to assess sequence similarity. Finally, a single template sequence is generated from each cluster via a process of sequence re-sampling, averaging, and re-quantisation.

The relative probability  $P(\alpha_{ij}^l)$ ,  $\sum_l P(\alpha_{ij}^l) = 1$ , of each template sequence within the atomic behaviour component  $\alpha_{ij}$  is derived from the training corpus by observing the relative frequency with which the templates are matched.

## 4.2 Inferring a higher-level behaviour grammar

Temporal segmentation into plausible atomic behaviour components enables the learning of a higher level behaviour model that efficiently captures temporal ordering and constraints between constituent atomic behaviour components. This is achieved using a VLMM to capture the memory conditioned probabilities of transitions between atomic behaviour components. For convenience, the VLMM  $M_k = (Q_k, K, \tau_k, \gamma_k, \pi_k)$  is trained using the set of key prototypes as an alphabet.

Although key prototypes are used as the alphabet, the memory conditioned probabilities of transitions between atomic behaviour components are implicit within the model. Suppose  $P_{M_k}(\mathbf{k}_j | s\mathbf{k}_i)$  denotes the probability of observing key prototype  $\mathbf{k}_j$  conditioned on the observation of history  $s\mathbf{k}_i$ , where  $s$  denotes a key prototype history. Since  $\alpha_{ij}$  is the atomic behaviour component representing the transition between  $\mathbf{k}_i$  and  $\mathbf{k}_j$ , clearly  $P_{M_k}(\alpha_{ij} | s\mathbf{k}_i) = P_{M_k}(\mathbf{k}_j | s\mathbf{k}_i)$ . Within such a behaviour model, the probability  $P_{M_k}(\alpha_{ij}^l | s\mathbf{k}_i)$  of observing template sequence  $\alpha_{ij}^l$ , conditioned on the observation of a key prototype history  $s\mathbf{k}_i$ , is given by:

$$P_{M_k}(\alpha_{ij}^l | s\mathbf{k}_i) = P(\alpha_{ij}^l) P_{M_k}(\alpha_{ij} | s\mathbf{k}_i). \quad (9)$$

Figure 3 illustrates part of a sample VLMM encoding the high-level structure of a behaviour. Also illustrated is the sub-model of an atomic behaviour component defined between two key prototypes that is implicitly built within the high-level model.

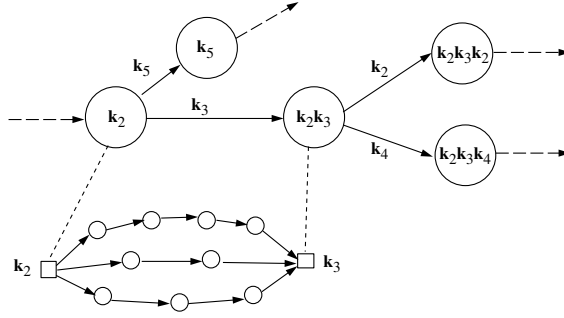


Figure 3: Structured behaviour model

### 4.3 Behaviour generation

The learnt structured behaviour model has stronger generative capabilities compared with the model in Section 3.2.1. This is due to the fact that temporal behaviour dependencies are encoded at a higher level offering a more powerful, longer duration memory mechanism.

Behaviour generation is achieved by traversing the model's automaton  $M_k$ , generating a key prototype at each step and replacing each  $k_i k_j$  subsequence with a  $k_i \alpha_{ij}^l k_j$  subsequence. The choice of template  $\alpha_{ij}^l$  representing the atomic behaviour  $\alpha_{ij}$  is achieved by either choosing the template that maximises equation 9 or sampling from the set of possible templates  $\alpha_{ij}^l$ .

Entirely hypothetical sequences can be generated using the start state distribution  $\pi_k$  to select an initial model state. The selection of the start state is based on either sampling from the start state distribution or identifying the most probable state. The start state distribution is approximated by the relative frequency of starting at a particular state of the VLMM model in the training data.

### 4.4 Prediction of future behaviour

In order to use the model  $M_k$  for behaviour prediction, it is necessary not only to locate the current model state (as in section 3.2.2), but also to identify the atomic behaviour template currently being observed and to locate the current position within this template. Once the atomic behaviour has been found, the subsequent model state is implicitly identified.

Suppose that, at time  $t$ , it has been established that the current state of the model  $M_k$  is  $q_c$ , having memory  $s\mathbf{k}_t$ , and that the prototype sequence  $O_t = \{\mathbf{o}_1, \mathbf{o}_2, \dots, \mathbf{o}_{t-1}, \mathbf{o}_t\}$  has been observed since the last observed key  $\mathbf{k}_j$ . In order to select the subsequent model state, it is necessary to identify the atomic behaviour template currently being traversed.

A Bayesian approach is taken where the estimated transition probabilities of the learned model  $M_k$  are used as *priors*. The posterior probability that atomic behaviour template  $\alpha_{ij}^l$

represents the observation sequence at time  $t$ , taking into account the history of the high-level behaviour model, is approximated by:

$$P(\alpha_{ij}^l | O_t, s\mathbf{k}_i) \propto P(O_t | \alpha_{ij}^l) P_{M_k}(\alpha_{ij}^l | s\mathbf{k}_i), \quad (10)$$

where  $P_{M_k}(\alpha_{ij}^l | s\mathbf{k}_i)$  is given by equation 9 and  $P(O_t | \alpha_{ij}^l)$  is the likelihood of template  $\alpha_{ij}^l$  giving rise to the current observation sequence.

All the atomic behaviour template sequences from all the possible atomic behaviour components defined from the last observed key prototype  $\mathbf{k}_i$  to any other key prototype  $\mathbf{k}_q$  are considered. For each such template sequence  $\alpha_{iq}^l$ , a position  $\xi \leq n$  is found such that the distance or the cost to align  $\{\mathbf{p}_{t_1}, \mathbf{p}_{t_2}, \dots, \mathbf{p}_{t_\xi}\}$  with  $\{\mathbf{o}_1, \mathbf{o}_2, \dots, \mathbf{o}_t\}$  is minimised using dynamic time warping. This minimum cost is denoted by  $c_t(i, q, r)$ . Utilising the cost function, it is possible to approximate the likelihood  $P(O_t | \alpha_{ij}^l)$  with the relative probability of atomic behaviour template  $\alpha_{ij}^l$  giving rise to the observation sequence  $O_t$ :

$$P(O_t | \alpha_{ij}^l) = 1 - \frac{c_t(i, j, l)}{\sum_{q,r} c_t(i, q, r)}. \quad (11)$$

Selecting the atomic behaviour template for which equation 10 is maximised identifies the subsequent key prototype  $\mathbf{k}_j$  and thus the subsequent state of the VLMM.

Unseen events are handled by detecting cases in which the maximum probability is less than a threshold  $\theta$ . In these cases, we lose all previous history of the high-level behaviour model and predict the next key prototype using only the likelihood function given by equation 11.

Having located the next state  $q_{c+1}$  of the VLMM model  $M_k$  and the position  $\xi$  within the atomic behaviour template currently being traversed, generation of future behaviour is possible. The generated behaviour comprises the remaining template behaviour sequence  $\{\mathbf{o}_{\xi+1}, \mathbf{o}_{\xi+2}, \dots, \mathbf{o}_{t-1}, \mathbf{o}_t\}$  and the behaviour generated from the state  $q_{c+1}$  using the model  $M_k$  either as a stochastic or a maximum likelihood behaviour generator as described in the previous section.

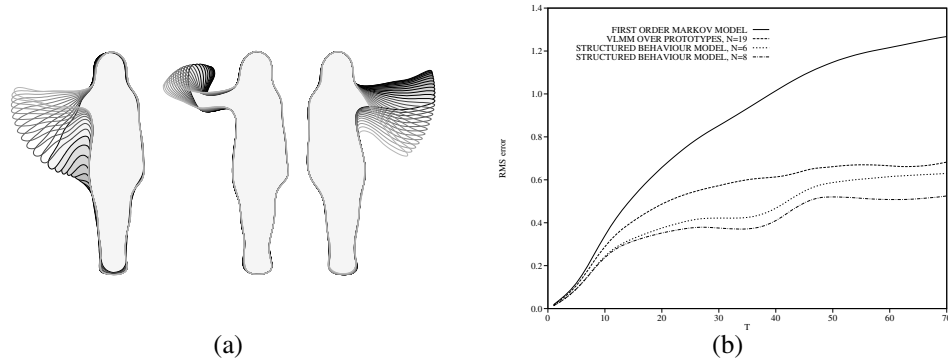


Figure 4: (a) Behaviour extrapolation (b) prediction results.

Figure 4(a) illustrates an example of maximum likelihood future behaviour extrapolation at selected time instants during the generation of a sample exercise routine. In each figure, recent behaviour is illustrated by a set of filled contours, the shade of which indicates recency, the lightest being the current shape. The first 12 frames of each extrapolation

are illustrated by a set of unfilled contours overlaying the recent behaviour, the shade of which indicates the progression of behaviour, the lightest being the furthest advanced.

## 4.5 Experiments

Using the training data and the set of prototypes  $P^{\text{shape}}$  described in Section 2.1, structured behaviour models have been learned with VLMMs capturing the structure of activity at a higher level using atomic behaviours as an alphabet. From the set of 71 prototypes  $P^{\text{shape}}$ , 5 prototypes have been identified as key prototypes and 8 atomic behaviour components representing the range of behaviour between key prototypes have been learned. VLMMs  $M_k$  were trained using memories of maximum length 6 and 8 with a threshold  $\epsilon = 0.0001$ , resulting in models with states 35 and 45 respectively. The test sequence from section 3.2.4 is again used here to assess prediction performance.

Figure 4(b) illustrates predictor performance using the learned structured behaviour model for various values of  $N$ . The performance of the behaviour model described in section 3.2. is also illustrated for comparison, and the structured behaviour models can be seen to give consistently better prediction, clearly indicating the increased ability of the model to encode the complex, long-term temporal dependencies.

## 5 Conclusions

Two novel methods have been proposed for the automatic acquisition of statistical models for structured and semantically rich behaviours. The use of variable length Markov models provides an efficient mechanism for learning complex behavioural dependencies and constraints.

Capturing behaviour at a low level using a VLMM with prototypical configurations as an alphabet, results in a model that accurately encodes local behaviour dependencies. However, such a model is unable to capture dependencies at multiple temporal scales. Using a VLMM at a higher level of abstraction, with constituent atomic behaviours as an alphabet, it is possible to automatically infer a stochastic model of the high-level structure of a behaviour. The learned structured behaviour model encodes behavioural dependencies at two temporal scales thus providing a more powerful model.

Both models have good generative capabilities and can be utilised for behaviour recognition, for the automatic generation of realistic object behaviours and for improving the robustness and efficiency of object tracking systems.

## References

- [1] A. Baumberg and D. Hogg. Learning Flexible Models from Image Sequences. In *European Conference on Computer Vision*, volume 1, pages 299–308, 1994.
- [2] T. Bell, J. Cleary, and I. Witten. *Text Compression*. Prentice Hall, 1990.
- [3] A. Bobick. Movement, activity, and action: The role of knowledge in the perception of motion. In *Philosophical Transactions Royal Society London*, 1997.
- [4] A. Bobick and Y. Ivanov. Action Recognition using Probabilistic Parsing. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 196–202, 1998.
- [5] C. Bregler. Learning and Recognising Human Dynamics in Video Sequences. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 568–574, 1997.

## BMVC99

- [6] L. Campbell and A. Bobick. Recognition of Human Body Motion Using Phase Space Constraints. In *International Conference on Computer Vision*, pages 624–630, June 1995.
- [7] G. Cormack and R. Horspool. Data Compression using Dynamic Markov Modelling. *Computer Journal*, 30(6):541–550, 1987.
- [8] D. Gavrilu. The Visual Analysis of Human Movement: A Survey. *CVIU*, 73(1):82–98, 1999.
- [9] I. Guyon and F. Pereira. Design of a Linguistic Postprocessor using Variable Memory Length Markov Models. In *International Conference on Document Analysis and Recognition*, pages 454–457, 1995.
- [10] J. Hu, W. Turin, and M. Brown. Language Modelling using Stochastic Automata with Variable Length Contexts. *Computer Speech and Language*, 11(1):1–16, 1997.
- [11] F. Jelinek. *Statistical Methods for Speech Recognition*. MIT Press, 1998.
- [12] N. Johnson, A. Galata, and D. Hogg. The Acquisition and Use of Interaction Behaviour Models. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 866–871, 1998.
- [13] N. Johnson and D. Hogg. Learning the Distribution of Object Trajectories for Event Recognition. *Image and Vision Computing*, 14(8):609–615, 1996.
- [14] A. P. Pentland. Machine Understanding of Human Motion. Technical Report 350, MIT Media Laboratory Perceptual Computing Section, 1995.
- [15] D. Ron, S. Singer, and N. Tishby. The Power of Amnesia. In *Advances in Neural Information Processing Systems*, volume 6, pages 176–183. Morgan Kaufmann, 1994.
- [16] H. Sakoe and S. Chiba. Dynamic Programming optimization for spoken word recognition. *IEEE Trans. Acoustics, Speech and Signal Processing*, 26:623–625, 1980.
- [17] T. Starner and A. Pentland. Visual Recognition of American Sign Language Using Hidden Markov Models. In *International Workshop on Automatic Face and Gesture Recognition*, pages 189–194, 1995.
- [18] C. Vogler and D. Metaxas. ASL Recognition Based on a Coupling Between HMMs and 3D Motion Analysis. In *International Conference on Computer Vision*, pages 363–369, 1998.