Identifying Humans by Their Walk and Generating New Motions Using Hidden Markov Models

CPSC 532A Topics in AI: Graphical Models and
CPSC 526 Computer Animation
December 15, 2004

Andrew Adam
andyadam@cs.ubc.ca

Saleema Amershi
samershi@cs.ubc.ca

1. INTRODUCTION

The problems we consider in this work are that of recognizing a human figure from 3D motion capture data of a person walking, and also of generating new movement data useful for movies, simulations or games. What is it that could make a human distinguishable from another by their walk? The answer is gait. Previous work in this area has shown that if all points on the body are considered it is possible to differentiate between individuals.

We attempt to recognize people by modeling each individual's gait using a Hidden Markov Model (HMM). The HMM is a good choice for modeling a walk cycle because it can model sequential processes. The data that we consider comes from the Carnergie Mellon University (CMU) motion capture database, but in theory any data that drives or represents a human figure could be used. We build each HMM by converting motion capture data from .amc format to Matlab matrices (*Converter*), using feature vectors from that data to produce cluster sets (*Clusterer*), initializing the model with appropriate values based on the cluster set (*Initializer*) and then improving the model parameters to best represent the original data (*Trainer*).

Once we have an HMM for each person, we can make use of the model in two applications: identifying individuals from new unseen data (*Identifier*) or from training data so as to test the model's accuracy (*Tester*), and generating new pieces of motion (*Generator*).

The rest of this paper is organized as follows: Section 2 discusses our work in the context of previous work on motion recognition and generation. In Section 3 we introduce and describe our approach to recognition and generation. Section 4 gives an overview of our results (further results can be examined by running our matlab code and viewing our previously generated files). In Section 5 we discuss the limitations of our system and describe avenues for future extension.

2. BACKGROUND and RELATED WORK

Distinguishing people by the way they walk is part of a greater problem of recognizing people using different types of biometrics. The background for our idea comes from Kale et al. [7] where silhouettes of a human figure are extracted from video images and used to build an HMM and then to recognize the person. This system would be useful where fingerprinting or eye scanning is unsuitable, and where there is surveillance cameras or with night vision goggles. Unfortunately its accuracy depends on the style of clothing the person wears. Instead of using a silhouette we will use 3D motion capture data but apply HMM's in a similar way.

HMMs appear in many previous works, Rabiners paper [15] describing their use in speech recognition being a classic example. This paper discusses what an HMM is and why they are useful and goes on to state that such models are "rich in mathematical structure

and hence can form the theoretical basis for use in a wide range of applications." [12] also tries to extend the HMM framework for the application of recognizing speech. Alongside this, [3] is a presentation on dimensionality reduction and feature selection and how they work, and also the application to human tracking.

There are many other works that attempt recognition from 2D or 3D data. [18] uses video data and HMMs to try and recognize which sign language symbol is being used at a particular moment in time. They manage to recognize 99% of the symbols when colored gloves are worn and 92% without gloves, from a 40 word lexicon. [9] attempts to add HMMs when recognizing activities that have a predefined context and inherent semantics, such as in the card game Blackjack. They extract behavior and strategies in real-time. [21] attempts to decode different types of gestures from video, using a technique developed by the same authors in [22] called a Parametric HMM (PHMM), that is used to model parameterized gestures (gestures that show a meaningful variation). An example of this would be a pointing gesture where the important parameter is direction, and therefore would be parameterized by the Cartesian coordinates that indicate direction. The PHMM recognizes the gesture and also estimates the quantifying parameters using the Expectation-Maximization algorithm (EM).

[16] also attempts classification from video sequences, and uses mixed discrete/continuous states to couple perception with classification. A spline contour is used to track the contour of the person, along with mixed state condensation filtering. Another work uses a switching linear dynamic system (SLDS) to analyze and track a human figure [13] cast in the framework of Dynamic Bayesian Networks (DBN's) [11]. [2] uses HMM's on video and the EM algorithm for human gait recognition and [14] is a more narrow recognition problem which attempts to recognize types of movements such as tennis stroke, again using an HMM. [20] is a Masters thesis by Donald O. Tanguay Jr. at MIT, which recognizes different types of motion gestures in video.

[23] presents a 3D recognition scheme and efficiently computes the 3D appearance using a region-based coarse stereo matching algorithm. An unsupervised learning scheme is carried out to capture the cluster structure of these feature volumes, then the image sequence of a gesture is converted to a sequence of symbols that initialize the cluster identities of each image pair. Two schemes are used (forward HMM's and Neural Networks) to model the dynamics of the gestures, and they achieve a recognition accuracy of 96%. [5] creates what the authors call a "Continuous Human Movement Recognition" framework, which uses multiple HMM's to infer the movement skill from an alphabet of "dynemes" (units of full body skills.) Also used is a "clone-body-model" which is dynamically sized and texture mapped for more robust tracking of humans on video of both edges and texture regions.

There are also many examples of work that generate new movement data from existing data. [1] is an example of stylistic motion synthesis that uses learning across a varied set of choreographed dancing data to create new dancing motions. They achieve this using a "Stylistic HMM (SHMM)", which can also be driven by video, scripts or even by noise to generate new choreography and synthesize virtual motion capture in many different styles. [4] classifies human movement compactly by using "primitives", by filtering, segmenting and applying the Principle Component Algorithm (PCA) but in this case uses it for generating movement in robotics. [6] synthesizes a walking human motion that follows a sample trajectory and also generates a "synthetic partner" for a dancer who is acquired through motion capture. [19] synthesizes novel motion sequences from a database of motion capture examples, providing "the flexibility of keyframe animation with the realism of motion capture data." Finally, [17] constructs new parametric models out of captured motion capture data, using a multi-step approximation for the interpolation function and motion data compressed by PCA, for use in real-time applications. They can vary the model by age, height, weight and gender.

3. SYSTEM OVERVIEW

We aim to model an individual's walk using a Hidden Markov Model (HMM) with discrete hidden states and continuous observations from the states. An HMM can model a walk cycle because it can model sequential stochastic processes, or states, where the

probability of a state depends on previous states. For a walk cycle we use a left-right HMM, where a state can only be reached by itself or by a single other state. The states are hidden and one can only observe a sequence of observations generated from a sequence of states. In our case we observe skeletal joint configurations of a person, and assume they are generated from the states by a Gaussian probability density function. That is we assume that at each consecutive time step a new skeletal pose is generated from some state. Figure 1 shows an example cyclic left-right HMM.

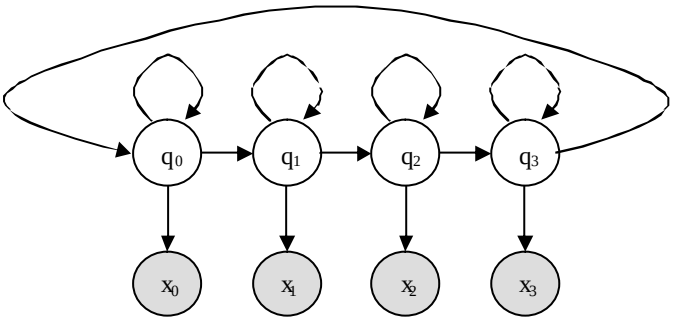

Figure 1. Cyclic Left-Right Hidden Markov Model

In Figure 1, the q_i nodes represent hidden states and the shaded x_i nodes the observations from these states. An HMM is parameterized by:

- transition probabilities a_{ij}, $1 \le i,j \le N$, which are the probabilities of transitioning from state q_i to state q_j in successive time slices,
- initial probabilities π_i, $1 \le i \le N$, which are the probabilities of a sequence starting in any state q_i, and
- observation probabilities $b_i(x)$, $1 \le i \le N$, which are the probabilities of observing x given being in state q_i.

These parameters can be summarized by $\lambda = (p, \mathbf{A}, B)$ where p is a vector of probabilities and **A** is a matrix of probabilities. For our system we will be modeling continuous Gaussian observations and so we can replace B with a Gaussian probability density function $N(\mathbf{x}; m, S)$ where m and S are the mean and covariance of the distribution.

Our goal is to automatically learn the parameters $\lambda = (p, \mathbf{A}, N)$ from motion capture data of walking sequences. Each person's walk will be modeled by one identically structured HMM. The unique parameters learned will be used to recognize people from new observation sequences. By performing random walks on the HMM we will also be able to produce new walking motions for each person. An overview of our system is shown in Figure 2. In the rest of this section we will explain our approach by describing each of the modules in Figure 2 in further detail.

3.1 Identification

3.1.1 Converting

For our system we use Acclaim formatted motion capture data from the CMU motion capture data base. The Acclaim format consists of *i)* a .asf file specifying skeleton information, and *ii)* a .amc file containing motion information for a sequence of frames. Each frame specifies the position and orientation of the root of the skeleton (6 dimensions) and joint angles for each joint in a skeleton (56 dimensions). For our system, we exclude skeletal information and ignore the root position and orientation, and focus recognition based on how a person is moving using only the joint angle information in the amc files. All of the individuals we use have structurally similar skeletons.

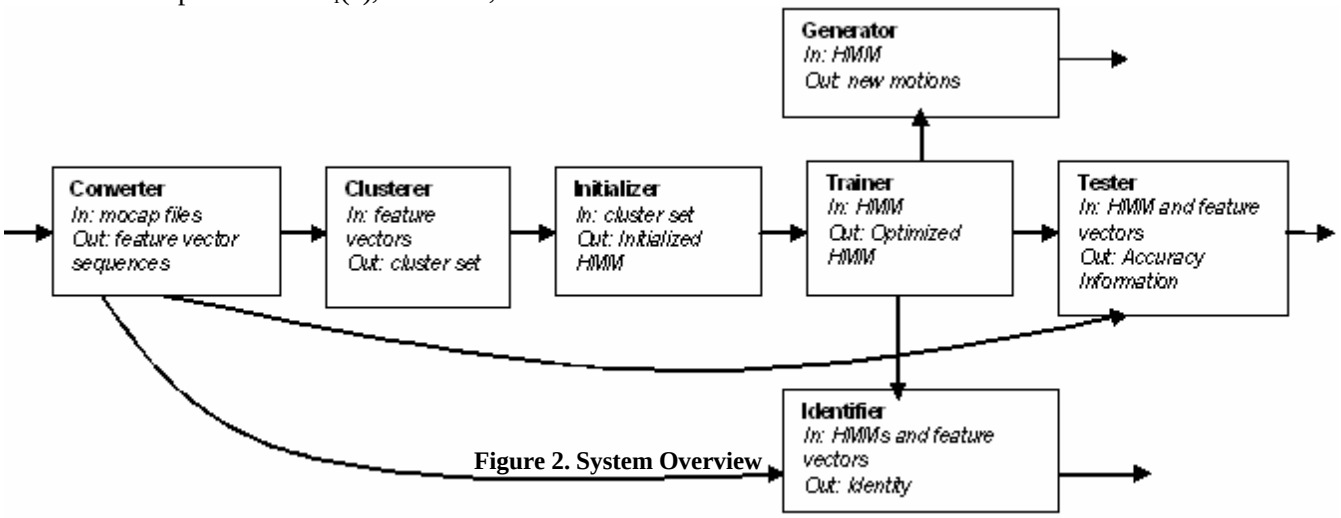

Figure 2. System Overview

We extract a 56 dimensional feature vector for every frame in a sequence and then preprocess the data by centering the feature vectors about the mean pose and normalizing.

3.1.2 Clustering

After extracting the necessary features, we identify clusters of data which will correspond to the states in our HMM. Each feature vector can be considered as a point in a 56 dimensional joint space. And a walk can be considered as cycles through similar poses. These similar poses (or points near each other in the joint space) can then be grouped into clusters, so as to partition the joint space.

To find these data clusters we employ the K-means clustering algorithm. The K-means algorithm identifies k vectors, each corresponding to the mean of a cluster in the joint space. Given a number, k, of clusters to find, the K-means algorithm iteratively identifies cluster means, m_i for i = 1,..,k, and assigns data points, $\mathbf{x}_n$, to clusters. We initialize the algorithm by randomly selecting feature vectors from our data to be the cluster means. For each feature vector, the squared distance between the feature and each cluster mean ($\| \mathbf{x}_n - m_i \|^2$) is then computed, and the feature vector is assigned to the cluster which minimizes the distance to its mean. After each feature vector in the data is assigned to a cluster, the algorithm re-computes the cluster means: $m_i = 1/N_i \Sigma \mathbf{x}_n$. The algorithm then reassigns the data to the new clusters (represented by the new means) and so on for a maximum of 100 iterations.

For each HMM we identify four clusters using K-means. We found that this is an adequate and natural representation for a walk cycle in our system. Figure 3 shows an example of the joint configurations corresponding to the four cluster means found for a person by our system.

3.1.3 Intializing

We can now initialize the parameters, $\lambda = (p, \mathbf{A}, N)$, of our HMM. Recall that we want to model the joint configurations of a walking sequence as continuous observations from discrete states. We model these continuous observations using:

$$b_i(\mathbf{x}) = N(\mathbf{x}; m_i, s_i), 1 \le i \le N$$

where $b_i(\mathbf{x})$ models the probability density function of observation $\mathbf{x}$ generated from state i as a Gaussian. The four partitions (N = 4) of the feature vectors found by K-means are used to initialize the Gaussian parameters m and s (the mean vector and covariance matrix respectively). The identified cluster means correspond directly to the mean vectors m_i. The form of s_i is spherical where the covariance is set to be the squared distance to the next closest mean vector.

We evenly distribute probability to the priors p of the HMM. This means that we assume it is equally likely

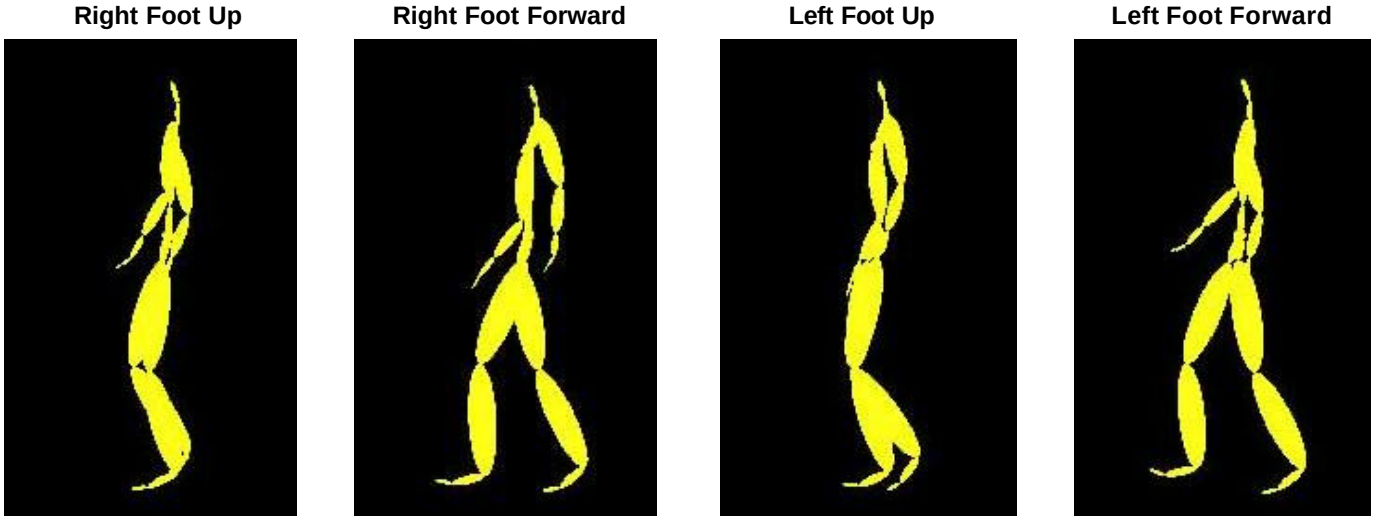

Figure 3. Four Cluster Means Found Using K-Means

to begin in any state of our HMM. EM (see Section 3.1.4) will adjust this prior to reflect the actual initial state.

The transition matrix **A** specifying the probability of transitioning from any state to any other state, is initialized based on frequency counts from the feature vector data. Each feature vector corresponds to a single frame in a sequence. From this and the partition information found in the last section, we can count how many times we transition from one state to another in consecutive frames. That is the element a_{ij} ($1 \le i,j \le N$) in **A** estimates the probability $p(q_{t+1}^{j} \mid q_t^{i})$ of transitioning from state q_i to state q_j in consecutive frames by counting the number of times in the sequence that we transition from state q_i to q_j, divided by the total number of times we are in state q_i.

3.1.4 Training

To improve our initial model parameters $\lambda = (p, \mathbf{A}, N)$, we use the expectation-maximization (EM) algorithm. Our goal is to maximize the complete log likelihood $p(\mathbf{X}, q \mid \lambda)$ that a sequence **X** of data vectors **x** was generated by our HMM by adjusting the HMM parameters λ. However, we do not have complete data because the state variables q_i are hidden. So we must maximize (the M step) the expected complete log likelihood

$$\text{argmax}_{\lambda} \Sigma_q \, p(q \mid \mathbf{X}, \lambda) \log p(\mathbf{X}, q \mid \lambda)$$

with respect to the parameters λ, where $p(q \mid \mathbf{X}, \lambda)$ is estimated in the E step. That is, in the E step we calculate the sufficient statistics for the HMM parameters λ using the forward-backwards algorithm, and then in the M step we compute the maximum likelihood (ML) estimates of the parameters. These ML parameters are the improved HMM parameters. We can then repeat the E and M steps using the new parameters to improve our model.

3.1.5 Testing/Identifying

We can create and train an HMM for several different people using the approach described in Sections 3.1.1-3.1.4 for each individual. Once we have a set of HMMs, recognition of a person from a new sequence of observations **x** amounts to calculating the log likelihood of **x** given each of the models ($\log p(\mathbf{x} \mid \lambda)$ for each HMM). The model that gives the highest probability is the model that most likely generated the new sequence **x.** To calculate $p(\mathbf{x} \mid \lambda)$ we can marginalize out the hidden states q from the joint distribution $p(\mathbf{x}, q \mid \lambda)$. That is, the likelihood is the sum of the likelihoods of **x** for every possible sequence of states q in the model

$$\log p(\mathbf{x} \mid \lambda) = \Sigma_q \log p(\mathbf{x}, q \mid \lambda).$$

Using the forwards step of the forwards backwards algorithm, we can efficiently compute this probability. In the forwards algorithm we can accumulate or store probabilities of sequences up to any time step t. Then we can update the probability for the next time step t+1 by multiplying the previously stored probabilities with the next transition and observation probabilities. The probabilities we store are

$$\alpha_t(i) = p(x_1, x_2, \ldots x_t, q_t^{i} = q_i \mid \lambda)$$

which are the probabilities of observing the sequence **x** up to time step t and ending up in state q_i given the model. At each successive time step we need only to update α by

$$\alpha_{t+1}(j) = [\Sigma_i \, \alpha_t(i) \, a_{ij}] \, b_j(x_{t+1})$$

where we sum the first equation for $\alpha_t(i)$ over all possible end states q_i multiplied by the transition probabilities of going from that end state to the current state q_j. We then only need to multiply by the probability of observing the current observation x from state q_j. This will in effect compute the likelihood of the data for an HMM.

We compute the log likelihood of the data for each HMM in our set and then identify a person by choosing the HMM that produced the highest value. We use this same procedure for testing each HMM against the training data (the data that was used to create and train the HMM) and for recognizing new data.

3.2 Motion Generation

For motion generation, a trained HMM for an individual is taken and the means of the clustered states are retrieved. Each HMM was built with four states which resulted in two mean vectors with outstretched legs and two with (roughly) straight legs, which is how we expected it (see Figure 3). To create a new walk we worked out the number of frames to generate by using the values in the transition

matrix **A**. The diagonal probability values in the transition matrix are large (~0.97), which correspond to staying in the current state for a certain length of time. Then in each row, there is one small probability (~0.03), the index of which corresponds to which unique state follows the current state. The two remaining values are zero to signify that the other states are impossible as followers (as necessary for the left-right HMM). By dividing one by the small value we get a figure (~35) which tells us the number of frames there should be between transitions. Also the order of the states is deduced from the indices of the small values:

$$\begin{pmatrix} 0.97 & 0 & 0 & 0.03 \\ 0 & 0.963 & 0.037 & 0 \\ 0.02 & 0 & 0.98 & 0 \\ 0 & 0.031 & 0 & 0.969 \end{pmatrix} \quad \begin{matrix} 1 \text{ to } 4 \\ 2 \text{ to } 3 \\ 3 \text{ to } 1 \\ 4 \text{ to } 2 \end{matrix}$$

State Order = 1 4 2 3

We average the number of frames between transitions to work out approximately how many frames there should be between each state. This number of frames is created by linearly interpolating between each mean state vector. The difference between the values of each state is worked out and divided by the number of transitions to create an alpha value which can be added to each frame to create a new corresponding position.

Also we have to reinsert position data that we removed for the identification process. We choose a start position and add the number of frames in a single walk cycle to that position in the test sequence to see how far the character has traveled, and we use this to interpolate between the start and end of our new sequence points.

4. RESULTS

We ran our system using four different test subjects, person 7, person 35, person 39 and person 44 from the CMU database as they had lots of walking sequences to work with. Each HMM was created and trained using three walking sequences from the subject. We tested each HMM against these training sequences and also against four unseen sequences per subject. We tested two different types of covariance for the clusters, spherical and PPCA, and set the number of clusters to four.

The results for each of the different types of covariance were different. Spherical achieved 6 out of 12 correct when tested against the training data. PPCA got three wrong therefore 9 out of 12. Spherical got correct classification of 10 out of 16 of the unseen files, and PPCA got 13 out of 16. So, while it seems PPCA is better, further tests may reveal that each one is better with a certain style of walk.

In terms of generation we took the HMM for each subject and then ran our generator on them, producing one walk cycle for each of the four subjects.[1]

5. DISCUSSION AND FUTURE WORK

In our system we used a 56 dimensional feature vector for our HMM observations. This is a large amount of data which may contain much redundant information. To minimize redundancy and compress the data we could have used Principal Component Analysis (PCA) to extract the significant components of the data. PCA is a common technique used for compression that computes the eigenvectors of the high dimensional space. By projecting our high dimensional data onto a lower dimensional space spanned by some number of eigenvectors, *n*, corresponding to the *n* largest eigenvalues found, we could have reduced this dimensionality and still retain the significant information needed for recognition. For generation we would need to invert this reduction to project back to our high dimensional space and retrieve the new motion data. However, the high dimensionality features used in our system did not substantially reduce performance and so we left this out.

Some joint angles contain more significant information about a walking motion. For example, the hip rotation may contain more variation then the ankle rotations when a person walks. In the future we could add weighting information to the feature vectors to give more importance to these more significant joints.

[1] To view our results, see our code and generated motions at http://www.cs.ubc.ca/~andyadam/project/project.zip

We used a standard left-right HMM for our system in which the state durations are modeled by a probability $p_i(d) = (a_{ii})^{d-1}(1- a_{ii})$, where a_i is the self transition probability for state q_i, and d is the number of consecutive self transitions. A variable duration HMM in which we explicitly specify the state durations would more naturally represent a walking motion because this would allow us to explicitly state the duration that an HMM must be in a certain state before transitioning to the next. That is, we do not want to jump from one state to the next without generating an appropriate number of observations from one state. In the future we could implement this type of HMM and compare performance to our current system.

The generated output that we produced could be improved in many ways. The generated motion is not ideal because we get a lot of foot sliding and floating figures. We could have interpolated the position data in a better fashion to keep the new walk along the floor. We also could have eliminated foot-sliding by using inverse kinematics and constraints to force the characters feet to be planted when in contact with the ground so as to have the character move in a more realistic manner such as in [8]. With more time we would make these additions to our system.

In theory our system could also be used for recognizing different types of motion rather than recognizing different people from the same type of motion. However this was not tested, but could easily be added as an extension in the future. We also could extend our system to be used in conjunction with vision systems that can track human figures in video sequences. For example [10], takes several 2D views of the human body, locates joint positions, and then uses them to estimate the body configuration and pose in 3D space. This appears to be an effective way of pulling 3D data out of 2D video so perhaps it, or similar systems, could be used as input to our system in the future.

6. REFERENCES

[1] Brand, M. and Hertzmann, A. Style Machines. *SIGGRAPH 2000,* ACM Press (2000), 193-192.

[2] Bregler, C. Learning and Recognizing Human Dynamics in Video Sequences. Proc. IEEE Conf. Comp. Vision and Pattern Recognition 1997, 7 Pages.

[3] Cohen, I. et al. Feature Selection Using Principle Feature Analysis. Slide Presentation.

[4] Fod, A., Mataric, M.J. and Jenkins O.C. Automated Derivation of Primitives for Movement Classification. *Autonomous Robots 12*, 1 (2002), 39 – 54.

[5] Green, R. and Guan, L. Quantifying and Recognizing Human Movement Patterns from Monocular Video Images - Part I: A New Framework for Modeling Human Motion. *IEEE Trans. on Circuits and Systems for Video Technology 14*, 2 (2004), 179-190.

[6] Hsu, E., Gentry, S. and Popovic, J. Example-Based Control of Human Motion. *SIGGRAPH/Eurographics Syp. on Computer Animation 2004*, ACM Press (2004), 69-77.

[7] Kale, A., Cuntoor, N. and Kruger, V. Gait-based Recognition of Humans Using Continuous HMMs. *IEEE Int. Conf. on Automatic Face and Gesture Recognition 2002*, IEEE Computer Society (2002), 336.

[8] Kovar, L., Gleicher, M. and Schreiner J. Footskate Cleanup for Motion Capture Editing. *SIGGRAPH 2003*, ACM Press (2002), 97-104.

[9] Moore, D. and Essa, I. Recognizing Multitasked Activities using Stochastic Context-Free Grammar. *SIGART 2002*, AAAI (2002), 770-776.

[10] Mori, G. and Malik, J. Estimating Human Body Configurations using Shape Context Matching. *ECCV 2002*, 666-680.

[11] Murphy, K. Dynamic Bayesian Networks. 2002.

[12] Ostendorf, M., Digalakis, V. and Kimball O. From HMMs to Segment Models: A Unified View of Stochastic Modelling for Speech Recognition. *IEEE Trans. Speech Audio Process 4*, 5 (1996), 360- 378.

[13] Pavlovic, V. and Rehg, J. Impact of Dynamic Model Learning on Classification of Human Motion. *CVPR 2000*, Pages 788-795.

[14] Petkovic, M., Jonker, W. and Zivokic, Z. Recognizing Strokes in Tennis Videos Using Hidden Markov Models. *Proc. of Int.*

Conference on Visualization Imaging, and Image Processing 2001.

[15] Rabiner, I. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proc. of IEEE 77*, 2 (1989), 257-286.

[16] Rittscher, J., Blake, A., and Roberts, S. J. Towards the automatic analysis of complex human body motions. *IVC(20) 12*, (2002) 905-916.

[17] Soo Lim, I. and Thalmann, O. D11: Generative Models from Capture Motion.

[18] Starner, T. and Pentland, A. Real-Time American Sign Language Recognition from Video Using Hidden Markov Models. *SCV95*, 1995

[19] Tanco, I. and Hilton, A. Realistic Synthesis of Novel Human Movements from a Database of Motion Capture Examples, *Proc. of the IEEE Workshop on Human Motion HUMO 2000*.

[20] Tanguay D. Jr. Hidden Markov Models for Gesture Recognition, Masters Thesis at Massachusetts Institute of Technology, Cambridge, June 1993

[21] Wilson A. and Bobick, A. Nonlinear PHMMs for the Interpretation of Parameterized Gesture. *Proc of the IEEE Conference on Computer Vision and Pattern Recognition 1998*, Page 879.

[22] Wilson, A. and Bobick, A. Parametric Hidden Markov Models for Gesture Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 21, 9 (1999).

[23] Ye, G., Corso, J. and Hager, G. Gesture Recognition Using 3D Appearance and Motion Features, 2004.