Implicit Probabilistic Models of Human Motion for Synthesis and Tracking

Hedvig Sidenbladh*[1] Michael J. Black[2] Leonid Sigal[2]

[1] Computational Vision and Active Perception Laboratory (CVAP)
Dept. of Numerical Analysis and Comp. Sci. KTH, SE-100 44 Stockholm, Sweden
hedvig@nada.kth.se http://www.nada.kth.se/~hedvig/

[2] Department of Computer Science, Brown University, Box 1910
Providence, RI 02912, USA.
black@cs.brown.edu http://www.cs.brown.edu/~black/

Abstract. This paper addresses the problem of probabilistically modeling 3D human motion for synthesis and tracking. Given the high dimensional nature of human motion, learning an explicit probabilistic model from available training data is currently impractical. Instead we exploit methods from texture synthesis that treat images as representing an *implicit empirical distribution*. These methods replace the problem of *representing* the probability of a texture pattern with that of *searching* the training data for similar instances of that pattern. We extend this idea to temporal data representing 3D human motion with a large database of example motions. To make the method useful in practice, we must address the problem of efficient search in a large training set; efficiency is particularly important for tracking. Towards that end, we learn a low dimensional linear model of human motion that is used to structure the example motion database into a binary tree. An approximate probabilistic tree search method exploits the coefficients of this low-dimensional representation and runs in sub-linear time. This probabilistic tree search returns a particular *sample* human motion with probability approximating the true distribution of human motions in the database. This sampling method is suitable for use with particle filtering techniques and is applied to articulated 3D tracking of humans within a Bayesian framework. Successful tracking results are presented, along with examples of synthesizing human motion using the model.

1 Introduction

Probabilistic models of human motion provide a representation that can be used both for synthesizing novel animations and for constraining the search in Bayesian tracking algorithms [37]. While the learning of such models from training sets of 3D human motions (e.g. joint angles over time) is an active area of research, the problem is made difficult by the dimensionality of the human

* Address at the time of publication: Dept. of Data and Information Fusion, Swedish Defense Research Institute (FOI), SE-172 90 Stockholm, Sweden. hedvig@foi.se.

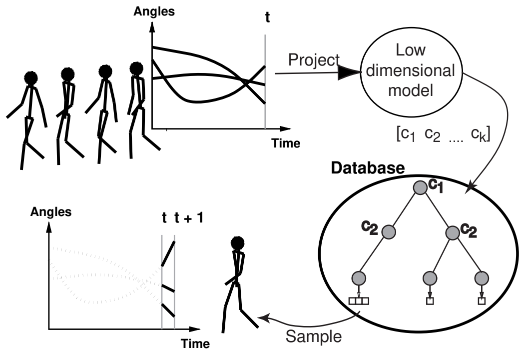


Fig. 1. Implicit probabilistic model of human motion (see text).

body, the variability in human motion, correlations among joint angles, and the correlations in motion over time. While recent work has seen some success at learning probabilistic models in small training sets or in supervised situations where the activities are known and clearly delimited, the general problem remains unsolved.

Rather than attempt to learn a general probabilistic model in the high dimensional space of human motions we exploit recent work on texture synthesis that treats an image as an *implicit probability distribution*. As with human motion, there has been some limited success at learning probabilistic models of the spatial statistics of images that allows texture synthesis [30, 32, 44]. Recent synthesis methods, however, replace the problem of learning with that of search [7–9, 14, 16, 40]. One incrementally constructs new textured regions by searching in the training images for example textures that have a similar neighborhood structure. The observation is that the important high order statistics are implicitly represented in the data and it is easier to *match* regions with similar statistics than it is to model them.

Here we extend these approaches to the problem of human motion modeling. The key idea with our motion model is to replace the problem of probabilistic *learning* with efficient probabilistic *search*. The idea is summarized in Figure 1. Given a large set of example human motions, over a time window of length d, we first construct a low dimensional model of the motion, by taking the time series of joint angles of length d and reducing the dimensionality using principal component analysis (PCA). Each length d subsequence in the set is then projected onto the resulting low-dimensional representation to give a vector of coefficients at each time instant. The database is then structured into a binary tree using these coefficients with the top node in the tree corresponding to the coefficient that captures the dimension of largest variance in the database. Lower levels in the tree capture the finer motion structure. Each of the leaf nodes contains an index into the motion database. This index gives the location of a time series corresponding to the body pose parameters (3D position and orientation of the body and the relative angles of all the joints).

Given a synthesized or observed motion from time $t - d$ to t, the goal is to predict (with the correct probability) the pose of the body at time $t + 1$. The synthesized or observed motion history is projected onto the subspace learned from the database examples. The coefficients obtained are then used to index into the database in a probabilistic manner, which approximates sampling from the distribution $p(\text{database example} \mid \text{synthesized motion})$. Once a example from the database is found, the pose at the next time instant in the database is taken to be the predicted motion at time $t+1$. The advantage of the tree representation is that a sample can be drawn in sub-linear time. This is particularly important for synthesis or tracking methods that exploit stochastic sampling such as the CONDENSATION algorithm [13, 18].

Below we relate this approach to recent work on texture synthesis and human motion modeling. After briefly presenting the Bayesian tracking framework in which the model is employed, we describe the database and the probabilistic tree search algorithm. We illustrate the implicit probabilistic model by using it to synthesize realistic motion sequences and by using it to provide a prior probability distribution over human motions for tracking. While our focus here is on developing a rigorous probabilistic model for Bayesian tracking, we expect the method to be useful for computer graphics applications and we will suggest extensions to make it more practical for human motion synthesis.

2 Related Work

There have been many approaches for modeling human motion. For narrow classes of motion such as walking, specific analytic functions of the joint angles have been proposed [17, 34]. For more general human motion, dynamical systems have been developed for tracking [41, 29] or animation (c.f. [5]). These systems can be computationally expensive and they may lack a clear probabilistic interpretation.

Instead of explicitly modeling the physics of the human body, one can *learn* statistical properties of human motion from 3D motion capture data. This approach has been used both for synthesis and tracking.[1] The learned statistical properties may be captured by wavelets [33], PCA [22, 37, 42, 43], polynomial basis functions of motion trajectories [12], or Hidden Markov Models (HMM) [3, 4, 26]. Full probabilistic models, however, remain difficult to learn given the dimensionality of the models and the limited availability of training data.

In contrast to traditional statistical learning methods, we make the most of available 3D motion capture data by keeping *all* of it. The challenge for probabilistic tracking is then to search it efficiently and to do so in a way that captures the underlying, implicit, probabilistic structure.

[1] It is worth noting that the goals of synthesis and tracking are somewhat different. In synthesis, a good deal of effort is expended to make sure that transitions are "smooth", visually pleasing, and physically meaningful. For tracking, what we need is a representative set of plausible motions. Image data will tell us which of these are reasonable.

In formulating this implicit motion model, we exploit recent work in texture synthesis. This work can be roughly viewed on a continuum from approaches that attempt to learn statistical models of texture [15, 30, 32, 44] to those that essentially treat an input texture as an implicit probabilistic model and use search to find matches between similar textures [7–9, 14, 16, 40].

The non-parametric texture models most similar to the approach here generate new textures from an example texture in roughly the following way. Given a randomly selected starting block of texture in the image, propagate out from it generating new texture blocks. For each new block in the image, examine any neighboring blocks that have already been generated and search the example image (or images) for similar textures. Find the k best such matches and then randomly choose the corresponding new texture patch from among them. The methods [8, 9, 14, 16, 40] all vary in how the blocks are represented, how similarity is determined, and how the search is performed.

The approach described here is a natural extension of this texture synthesis idea to sequences of joint angles. Previous extensions to time have focused on synthesis and prove unsuitable for Bayesian tracking [1, 35, 40]. For cyclic articulated motion, Pullen and Bregler [33] use a frequency decomposition of joint angles with a learned, non-parametric, kernel density estimate of the conditional statistics across frequency bands [7, 15]. Sampling from this model produces synthetic repetitive motions with natural variation. More closely related to the work described here is the work on video textures [35] which uses a pixel-based match metric to construct a matrix of probabilities that captures the similarities between frames in a video sequence and that can be used to transition between frames to construct an infinitely looping video sequence with apparently natural variation. In contrast to our approach, the method uses relatively short sequences of a single type of motion and, hence, all possible transitions can be pre-computed (and even optimized for display). Molina and Hilton [26] use a similar approach for modeling human motion. Since this type of motion is more diverse and has structure over longer time intervals, it is not possible to learn transition probabilities between all possible poses. Instead, a large set of example poses are grouped using vector quantization, and transition probabilities between the different groups are learned using an HMM formulation. A pose in the group found at each time step is selected based on constraints of smoothness over time. Our approach differs in that we do not *learn* transition probabilities between states. Instead, at each time step of the synthesis, we *search* among a large set of previously observed motions to find a plausible motion that fits with the synthesized motion history. We have thus replaced the problem of learning with that of searching a large database.

The most recent motion texture models for image sequences assume stationary statistics and use simple autoregressive models to capture temporal image change [10, 39]. It is not clear whether these methods will scale to the problem of representing general human motion where the assumption of stationarity is violated.

For general motions an example-based model such as ours will only be practical if efficient search methods can be employed. Moreover, if it is to be used for Bayesian tracking, it must have a sound probabilistic interpretation. There are two issues to be addressed: defining a match metric and a search procedure [21, 40]. Exhaustive search in a large dataset is prohibitive and an algorithm with sub-linear time complexity is required. Thus, the metric must be defined so as to allow a hierarchical search in the dataset. Matching can be be performed using wavelet coefficients [1, 21] and various pyramid representations [7, 15, 40]. Here we use a database-specific basis set learned using PCA. This provides an approximate representation of the data and has the property that the basis functions provide a decomposition of the data ordered by the variance accounted for. Various search approaches have been proposed and include kd-trees [2, 14], approximate nearest neighbor search (ANN) with PCA coefficients [16, 24, 27], dynamic space partitioning [28] and tree-structured vector quantization [16, 40]. Our approach extends these ideas to provide a probabilistic tree search using PCA coefficients.

It is worth noting that Chenney and Forsyth [6] have shown that samples such as those generated by our probabilistic tree search can be used to generate motions satisfying various external constraints.

3 Probabilistic Tracking Framework

The motion model described in this paper is employed in a probabilistic tracking framework. Given a model of a human, parameterized at time $t-1$ by the m parameters[2] $\boldsymbol{\phi}_{t-1} = [\phi_{1,t-1}, \ldots, \phi_{m,t-1}]^T$, the tracking of the model parameters over time can be formulated using Bayes' rule as

$$p(\boldsymbol{\phi}_t \mid \mathbf{I}_t) = \kappa\, p(I_t \mid \boldsymbol{\phi}_t) \int p(\boldsymbol{\phi}_t \mid \boldsymbol{\phi}_{t-1})\, p(\boldsymbol{\phi}_{t-1} \mid \mathbf{I}_{t-1})\, d\boldsymbol{\phi}_{t-1} \tag{1}$$

where I_t is the image at time t, $\mathbf{I}_t$ the image sequence up to t, and κ a normalizing constant independent of $\boldsymbol{\phi}_t$. At each time step, the *posterior* distribution $p(\boldsymbol{\phi}_t \mid \mathbf{I}_t)$ is estimated. This distribution is represented by a set of samples or particles, which are propagated in time using a particle filter [13, 18]. Each particle i represents a certain pose $\boldsymbol{\phi}_t^i$ of the human model, i.e. a certain location in the parameter space.

The distribution $p(I_t \mid \boldsymbol{\phi}_t)$ is the *likelihood* of observing the image I_t, conditioned on model configuration $\boldsymbol{\phi}_t$. In the particle representation, each model configuration $\boldsymbol{\phi}_t^i$ is projected into the image I_t and assigned a likelihood according to an image-model similarity measure. For details on the likelihood model and the tracking framework, the reader is referred to [36, 37].

[2] The model is a 3D articulated assembly of truncated cones, with 50 parameters comprising the position and velocity of the torso, the joint angles between the cones (limbs), and the angular velocities [36]. However, the framework applies to any parameterized model.

The motion model presented here, is used to formulate the *temporal prior* $p(\boldsymbol{\phi}_t \mid \boldsymbol{\phi}_{t-1})$. This conditional distribution is used to propagate the particles in time so that the correct part of the parameter space is covered at each time instant. Due to the particle representation, it is sufficient to design a motion model that allows *drawing samples* $\boldsymbol{\phi}_t^s$ from the distribution $p(\boldsymbol{\phi}_t \mid \boldsymbol{\phi}_{t-1})$. Details are described in the following section.

4 Implicit Probabilistic Motion Model

The motion database consists of multiple sequences of body pose parameters recorded with a 3D motion capture system; sequences include two male and two female actors walking, running, dancing, skipping and lifting.

Let $\boldsymbol{\psi}_i = [\psi_{1,i}, \ldots, \psi_{m,i}]^T$ be a recorded vector of m pose parameters of the human model, stored at index location i in the database. More specifically, the pose parameters are the global position and orientation of the model, as well as joint angles, and the velocities of all these parameters. Let $\Psi_i = [\boldsymbol{\psi}_i^T, \ldots, \boldsymbol{\psi}_{i-d}^T]^T$ be the dm-dimensional vector containing the sequence of the d vectors of parameters up to and including the angles at location i. Similarly, let $\boldsymbol{\phi}_t = [\phi_{1,t}, \ldots, \phi_{m,t}]^T$ represent the m pose parameters in a synthesized (or in a tracking framework, estimated) pose at time t, and let $\Phi_t = [\boldsymbol{\phi}_t^T, \ldots, \boldsymbol{\phi}_{t-d}^T]^T$ be the synthesized (or estimated) sequence between time $t-d$ and t. In all experiments here we take $d = 10$ [22] which corresponds to 1/3 sec.

In the standard formulation of the probabilistic motion model, the temporal prior $p(\boldsymbol{\phi}_t \mid \boldsymbol{\phi}_{t-1})$ satisfies a first-order Markov assumption. Here, instead of drawing samples $\boldsymbol{\phi}_t^s$ from the the Markov prior, we augment the state space to store the history, Φ_{t-1}, over the previous d time instants and draw samples from a distribution $p(\boldsymbol{\phi}_t \mid \Phi_{t-1})$ [3]. In the example based formulation, to draw samples $\boldsymbol{\phi}_t^s$ from $p(\boldsymbol{\phi}_t \mid \Phi_{t-1})$ we rewrite it as

$$p(\boldsymbol{\phi}_t \mid \Phi_{t-1}) = p(\boldsymbol{\phi}_t \mid \Psi_{i-1})\; p(\Psi_{i-1} \mid \Phi_{t-1}),$$

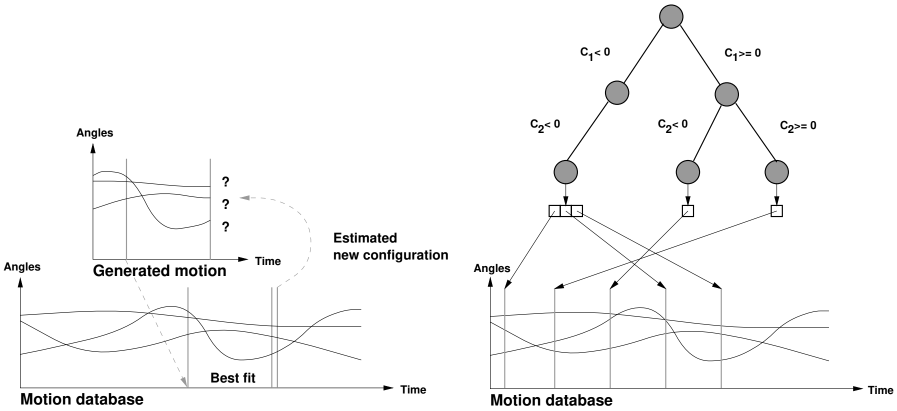
where

$$p(\boldsymbol{\phi}_t \mid \Psi_{i-1}) = \begin{cases} 1 & \text{if } \boldsymbol{\phi}_t = \boldsymbol{\psi}_i, \\ 0 & \text{otherwise.} \end{cases}$$

Thus, sampling from the prior $p(\boldsymbol{\phi}_t \mid \Phi_{t-1})$ corresponds to drawing samples Ψ_{i-1}^s from $p(\Psi_{i-1} \mid \Phi_{t-1})$, and selecting $\boldsymbol{\phi}_t^s = \boldsymbol{\psi}_i^s$, where $\boldsymbol{\psi}_i^s$ is the pose directly following the stored motion Ψ_{i-1}^s. The key idea is that sampling from $p(\Psi_{i-1} \mid \Phi_{t-1})$ is approximated by an efficient probabilistic search of the database of motions as described below.

First, the variance of each pose parameter in the database is computed and stored in an $m \times m$ diagonal covariance matrix Γ. Let Γ_d be the $dm \times dm$ covariance matrix created by storing d copies of Γ along the diagonal.

[3] The violation of the Markov assumption in (1) can be dealt with by treating $p(\boldsymbol{\phi}_t \mid \Phi_{t-1})$ as a proposal distribution and appropriately re-weighting samples in our particle filtering framework [19].



(a) The history of generated motion is compared with examples of motion, and the next time step is selected from an example whose history fits well with the generated history.

(b) Binary search tree structure. Each leaf contains a list of pointers to motion examples in the database with coefficients corresponding to the search path to that leaf.

Fig. 2. Indexing into the database (see text).

We define a generative model that states that the stored human motion data looks like the synthesized data plus Gaussian noise

$$\Psi_i = \Phi_t + \boldsymbol{\eta}(\Gamma_d) \tag{2}$$

where $\boldsymbol{\eta}(\Gamma_d)$ is an dm-dimensional vector of Gaussian noise. Then the probability that any length d segment, Ψ_i, in the database matches a synthesized sequence Φ_t is given by

$$p(\Psi_i \mid \Phi_t) = \kappa\, \mathrm{e}^{-\frac{1}{2}(\Psi_i - \Phi_t)^T \Gamma_d^{-1} (\Psi_i - \Phi_t)} \tag{3}$$

where $\kappa = (2\pi)^{-\frac{dm}{2}}(\det(\Gamma_d))^{-\frac{1}{2}}$. For simplicity, we assume a diagonal covariance matrix.

As illustrated in Figure 2a a naive approach to synthesis would compute $p(\Psi_i \mid \Phi_t)$ for every subsequence Ψ_i in the database. Then, $i = \arg\max_i p(\Psi_i \mid \Phi_t)$ would be chosen and the index $i+1$ would provide the new pose parameters, $\boldsymbol{\psi}_{i+1}$. However, this search strategy would have a time complexity of $\mathcal{O}(n)$ where n is the number of entries into the database, and would not scale to the large databases needed for tracking of human motion in general. A search strategy performing in sub-linear time has to be developed, which means that the database has to be structured to avoid comparison with all database elements.

This search exploits a low-dimensional model of the data. Let $\bar{\Psi} = (\sum_{i=1}^{n} \Psi_i)/n$ be a length dm vector representing the mean of all subsequences in the database of motions. Let $\hat{A} = [\hat{\Psi}_1, \cdots, \hat{\Psi}_n]$ be the $dm \times n$ matrix of all subsequences where the mean motion has been subtracted; that is, $\hat{\Psi}_i = \Psi_i - \bar{\Psi}$.

Performing singular value decomposition (SVD), we write $\hat{A}$ as $\hat{A} = U\Sigma V^T$ where the $dm \times n$ matrix U contains the principal components of $\hat{A}$ and Σ is a diagonal matrix in which the diagonal entries represent the standard deviation σ_l accounted for by each of the principal components $l = 1, \ldots, n$.

We select the first b principal components where $b = \lfloor \log_2(n) \rfloor$. Given our training set of $n \approx 50000$ time points, $b = 16$ and accounts for 89% of the variance in the training data; that is, $\sum_{l=1}^{b} \sigma_l^2 / \sum_{l=1}^{n} \sigma_l^2 \geq 0.89$. Let $\tilde{U}$ be the b first columns of U, and $\tilde{\Sigma}$ the matrix with the b first singular values σ_l, $l = 1, \ldots, b$ along the diagonal. Using this sub-space representation, we can approximate any subsequence, Ψ_i, in the database as

$$\Psi_i \approx \tilde{\Psi}_i = \bar{\Psi} + \tilde{U}(\tilde{U}^T \hat{\Psi}_i) = \bar{\Psi} + \tilde{U}\mathbf{c}_i \ . \tag{4}$$

where the vector of coefficients $\tilde{U}^T \hat{\Psi}_i = \mathbf{c}_i$ provides a "descriptor" for the motion parameters Ψ_i. Analogously, the generated motion Φ_t can be approximated as

$$\Phi_t \approx \tilde{\Phi}_t = \bar{\Psi} + \tilde{U}\mathbf{c}_t \tag{5}$$

where $\mathbf{c}_t = \tilde{U}^T \hat{\Phi}_t$, and $\hat{\Phi}_t = \Phi_t - \bar{\Psi}$.

4.1 Tree representation.

There are many ways to represent data such as ours for nearest neighbor search; e.g. given a probe motion (set of coefficients) find the k nearest neighbors. Our goals are somewhat different. For our tracking task, we seek a representation that can be searched such that each search result corresponds to a sample from some underlying distribution. These samples will only approximate the true distribution and the formulation trades off accuracy for efficiency. The method proposed here exploits the structure of our problem and may not be applicable to other search problems with a different probabilistic structure.

The motion examples in the database are sorted into a binary tree of depth b according to their coefficients $\mathbf{c}_i = [c_{i,1}, ..., c_{i,b}]^T$ (Figure 2b). The top node of the tree corresponds to the coefficient $c_{i,1}$. The database is split based on the sign of $c_{i,1}$ for each motion example i. Similarly, at the next level, the data is divided again at each node based on the sign of $c_{i,2}$. For each node (at depth l), the left subtree contains motion examples with eigencoefficient $c_{i,l} < 0$, while the right subtree contains samples with eigencoefficient $c_{i,l} \geq 0$. The process continues to the leaves of the tree at the depth b. Each leaf of the tree contains a list of pointers to the actual motion sequences in the database.

This process does not guarantee a balanced tree and, thus, some parents will have one child instead of two, and some leaves will contain more than one sample (see Figure 2b). In practice, the PCA representation results in a tree that is approximately balanced. Leaf nodes in our experiments contain between zero and several hundred samples. A balanced tree could be obtained by computing the median of the coefficients at each node and dividing based on that value [2].

4.2 Probabilistic search.

The tree described above can be searched in a probabilistic fashion so that searching the tree roughly approximates sampling from the distribution $p(\Psi_i \mid \Phi_t)$ (Equation (3)). The approach exploits the generative model above and uses the coefficients $c_{i,l}$ at each level l to randomly select the left or right subtree.

Observation 1 *If $\tilde{U}$ are the b axes with largest variation in the space of motion examples Ψ_i, $\bar{\Psi}$ the center of this space, $\mathbf{c}_i = \tilde{U}^T(\Psi_i - \bar{\Psi}) = \tilde{U}^T\hat{\Psi}_i$ the projection of Ψ_i into the sub-space spanned by $\tilde{U}$, and $\mathbf{c}_t = \tilde{U}^T(\Phi_t - \bar{\Psi}) = \tilde{U}^T\hat{\Phi}_t$ the projection of a generated motion Φ_t, then*

$$p(\Psi_i \mid \Phi_t) \approx p(\mathbf{c}_i \mid \mathbf{c}_t).$$

Explanation: $p(\Psi_i \mid \Phi_t)$ in Equation (3) can be written as

$$\begin{aligned} p(\Psi_i \mid \Phi_t) &= \kappa\, \mathrm{e}^{-\frac{1}{2}((\Psi_i-\bar{\Psi})-(\Phi_t-\bar{\Psi}))^T \Gamma_d^{-1}((\Psi_i-\bar{\Psi})-(\Phi_t-\bar{\Psi}))} = \\ &= \kappa\, \mathrm{e}^{-\frac{1}{2}(\hat{\Psi}_i-\hat{\Phi}_t)^T \Gamma_d^{-1}(\hat{\Psi}_i-\hat{\Phi}_t)} = \kappa\, \mathrm{e}^{-\frac{1}{2}(\hat{\Psi}_i^T \Gamma_d^{-1}\hat{\Psi}_i - \hat{\Phi}_t^T \Gamma_d^{-1}\hat{\Phi}_t)} \ . \end{aligned} \tag{6}$$

This can be characterized by the *Mahalanobis* distances [25] of the two motions from the learned feature space:

$$d(\Psi_i) = \hat{\Psi}_i^T \Gamma_d^{-1}\hat{\Psi}_i \ , \ d(\Phi_t) = \hat{\Phi}_t^T \Gamma_d^{-1}\hat{\Phi}_t \ .$$

Using the learned basis $\tilde{U}$ and the diagonal matrix of singular values $\tilde{\Sigma}$ of size $b \times b$, containing the b first diagonal elements from Σ, the Mahalanobis distance can be approximated by projecting it onto the subspace spanned by $\tilde{U}$ [25]:

$$d(\Psi_i) = \hat{\Psi}_i^T \Gamma_d^{-1}\hat{\Psi}_i \approx \hat{\Psi}_i^T \tilde{U}\tilde{\Sigma}^{-2}\tilde{U}^T\hat{\Psi}_i = (\tilde{U}^T\hat{\Psi}_i)^T\tilde{\Sigma}^{-2}\tilde{U}^T\hat{\Psi}_i = \mathbf{c}_i^T\tilde{\Sigma}^{-2}\mathbf{c}_i \ .$$

Similarly, $d(\Phi_t) \approx \mathbf{c}_t^T\tilde{\Sigma}^{-2}\mathbf{c}_t$. Inserting these approximations into (6) gives

$$p(\Psi_i \mid \Phi_t) \approx p(\mathbf{c}_i \mid \mathbf{c}_t) = \kappa\, \mathrm{e}^{-\frac{1}{2}(\mathbf{c}_i^T\tilde{\Sigma}^{-2}\mathbf{c}_i - \mathbf{c}_t^T\tilde{\Sigma}^{-2}\mathbf{c}_t)} = \kappa\, \mathrm{e}^{-\frac{1}{2}(\mathbf{c}_i-\mathbf{c}_t)^T\tilde{\Sigma}^{-2}(\mathbf{c}_i-\mathbf{c}_t)} . \ \square$$

Thus, the samples in the database can be compared to a generated motion using only the eigencoefficients $\mathbf{c}$. The error in the approximation can be estimated from the residual eigenvalues σ_l, $l = b+1, \ldots, dm$, i.e. the eigenvalues of the dimensions in U that are not included in $\tilde{U}$ [25].

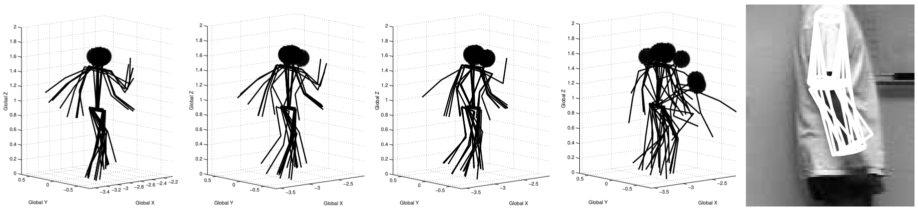
Search Algorithm. Given a "probe" motion, Φ_t, project it onto the basis set to compute the coefficients $\mathbf{c}_t$. Starting at the top of the tree with coefficient $c_{t,1}$ and proceeding down the tree for each level l, decide which branch to chose (left or right) based the probabilities

$$p_{\text{right subtree}} = p(c_{i,l} \geq 0 \mid c_{t,l}) = \frac{1}{\sqrt{2\pi\beta}\sigma_l}\int_{z=-\infty}^{c_{t,l}} \mathrm{e}^{-\frac{z^2}{2\beta\sigma_l^2}}\, dz \ , \tag{7}$$

$$p_{\text{left subtree}} = 1 - p_{\text{right subtree}} \ , \tag{8}$$

where β is a "temperature" parameter described below.

Assuming the Gaussian model from Observation 1, a branch at level l is selected with the probability that the coefficient value $c_{t,l}$ falls on that side of the tree. Since this choice is probabilistic, "sampling" from the tree many times will result in different paths through the tree to the leaf nodes. Note that for



(a) $\beta = 0.01$ (b) $\beta = 0.1$ (c) $\beta = 1.0$ (d) $\beta = 10.0$ (e) Tracking

Fig. 3. Sampling from the database. The temperature parameter β affects the randomness in the sampling. In each figure, 10 pose candidates $\boldsymbol{\phi}_{t+1}$ were sampled from the database using a generated motion Φ_t (a) Very low temperature. (b) Low temperature. (c) Neutral temperature. (d) High temperature. (e) Samples from an example of arm tracking (see also Section 6).

simplicity, σ_l is derived from the entire data set rather than being conditioned on the choices above.

When a leaf is reached using this probabilistic search, one of the examples in the leaf, Ψ_i^s, is selected. This can be done by computing $p(\mathbf{c}_i \mid \mathbf{c}_t)$ for each i in the leaf and then using a Monte Carlo sampling technique to chose a particular i. As suggested by Observation 1, this approximates sampling from the leaf using $p(\Psi_i \mid \Phi_t)$ (Equation (3)). Alternatively, one can sample uniformly from the leaf node; this works well in practice and is more efficient.

The new state, sampled from the example database, is defined as $\boldsymbol{\phi}_{t+1}^s = \boldsymbol{\psi}_{i+1}^s + \boldsymbol{\eta}$ where $\boldsymbol{\psi}_{i+1}^s$ is the pose directly following Ψ_i^s in time, and $\boldsymbol{\eta}$ is a small Gaussian noise term with empirically determined variance. This noise is added in both synthesis and tracking to generate samples that differ slightly from the training data. $\boldsymbol{\phi}_{t+1}^s$ is concatenated to the $d-1$ most recent entries in Φ_t^s to create Φ_{t+1}^s which is the motion history of particle s at time $t+1$.

The temperature parameter β controls the amount of randomness in the probabilistic tree search. Figure 3 illustrates the effect of β with samples drawn from the database using Equation (7). A generating motion example Φ_t was first selected from the database. Then, a probabilistic tree search was performed 10 times. For each of the 10 found database examples Ψ_i, the pose $\boldsymbol{\psi}_{i+1}$ directly following the motion example was selected and plotted. Thus, the variance within the 10 found poses reflects the accuracy with which the examples were found. A low temperature (e.g. 3a) results in samples that are similar to Φ_t. Alternatively, a very high temperature (e.g. 3d) will lead to an almost uniform sampling of the search tree, which means that the distribution over possible body poses will roughly approximate the *prior* probability distribution, $p(\Psi_i)$, over all motions. In a tracking application, the temperature parameter controls how strongly the motion prior guides the tracking. Furthermore, a lower temperature is typically needed if the model will be used for synthesis, since it is not guided by image measurements.

The total search cost for a single sample involves a logarithmic time tree search followed by a search that is linear in the number of elements in the leaf node. The relationship between the size of the leaf nodes, the depth of the tree,

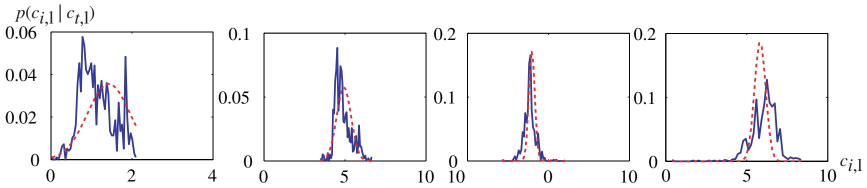


Fig. 4. Distribution over coefficients c_3, c_7, c_{11}, c_{15} obtained by probabilistic tree search (see text). Solid: empirical distribution from sampling. Dashed: Gaussian model.

and the size of the database requires further study. The overall cost, however is significantly better than linear in the size of the database. This efficiency is particularly important if the samples are to be used for particle filtering where predicting each particle involves sampling the tree with a different probe.

Observation 2 *The "true" distribution $p(\Psi_i \mid \Phi_t)$ or $p(\mathbf{c}_i \mid \mathbf{c}_t)$ is unknown. The probabilistic tree search provides an approximate model and samples from the tree search are more "realistic" than those from the Gaussian model in (3).*

Intuition: In the general case, there is no guarantee that the probabilistic tree search will approximate sampling from $p(\Psi_i|\Phi_t)$. For example, if the coefficients c_l for each basis direction l were actually normally distributed and statistically independent, then the tree search could provide a poor approximation to sampling the true distribution.

Given the nature of human motion however, PCA does not result in bases that make the coefficients statistically independent and, hence, these dependencies are represented in the branching structure of the tree. Intuitively, similar motions have similar sets of coefficients and these motions end up being grouped in the leaf nodes.

This can be seen in Figure 4 which plots the empirical distribution over a few coefficients using 1000 samples, Ψ_i, drawn for a single probe, Φ_t (with $\beta = 1$). For comparison, the figure also plots the distribution over each coefficient assuming independent Gaussian noise. This example illustrates the general agreement between the independent Gaussian model and samples drawn with the tree. By adopting our model however, we can also capture the non-Gaussian nature of the human motion data present in the database.

Further work needs to be done to formalize the tree search procedure, understand the nature of the approximation, and characterize the problems to which it is applicable.

5 Visualizing the Model (Synthesis)

Although our primary application for this motion model is as a prior to guide probabilistic tracking, the probabilistic model can also be used for synthesis. The goal of the visualization is to generate a new synthetic motion incrementally, by indexing into the database at each time step, using the generated motion at the previous d time steps.

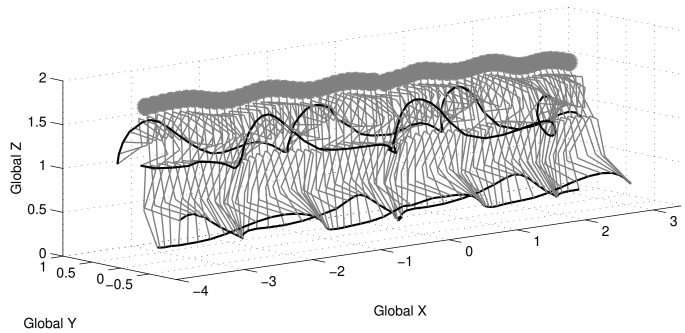


Fig. 5. Generated running motion, 2 s at 30 Hz. The pose every frame is shown in gray. Trajectories for hands and wrists are plotted in black.

Note that the synthesis does not have a goal function. The type of motion can be controlled up to a certain point by the choice of start motion. However, due to the probabilistic tree search there is a certain randomness in the synthesis.

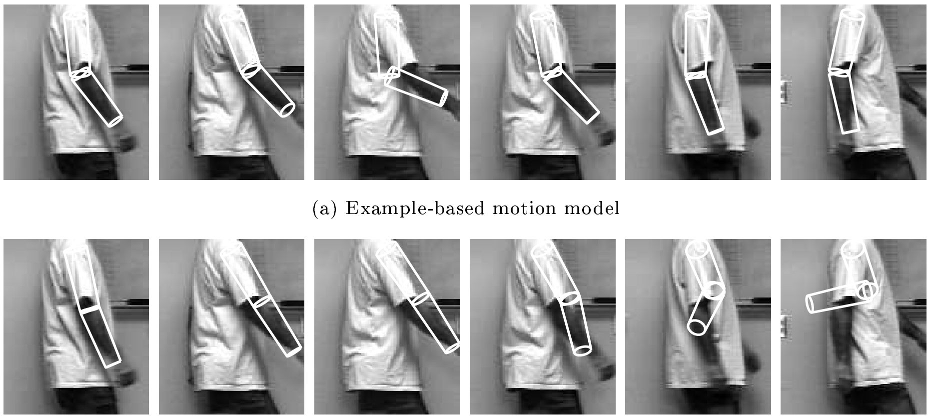
In Figure 5, a 2 second long running motion is visualized.[4] Note that the database contains several types of activities, not just running. The regularity with which the feet are planted (Figure 5) indicates that the periodicity is well preserved in the synthesized running sequence, even though the phase of the running cycle is not explicitly modeled.

To generate more varied and non-cyclic motion, a larger database of examples is needed to find smooth transitions between many different types of motion (β can be used to control the probability of changing motions). Finally, it is important to note that the motion model has no notion of gravity, friction, and position of the feet relative to the floor. To be able to generate plausible-looking motions for longer periods, constraints need to be introduced on the position and rotation of the human model so that it fulfills basic kinematic and dynamic requirements [6, 11]. Furthermore, in general it is desired to be able to edit the generated motion iteratively, and to introduce goals in the motion generation [26]. Therefore, the motion model is, in its present state, more suited for tracking than for motion synthesis. However, the example based scheme introduces fine realistic details that are often lost in a learned motion model.

6 Tracking Results

The example-based motion model is now evaluated in terms of its performance as a temporal prior for *monocular* 3D tracking, as described in Section 3. We compared it to a very general temporal model of constant (angular) velocity in the parameters, where all parameters are considered independent [36]. The generality of the constant velocity model allows tracking of any type of motion, but introduces problems in high-dimensional spaces. Using particle filtering methods as we do here is problematic given that the number of required particles is

[4] A movie of the generated motion in Figure 5, along with other synthesis examples, can be found at `http://www.nada.kth.se/~hedvig/mpegs/movies.html`.



(a) Example-based motion model

(b) Constant velocity model

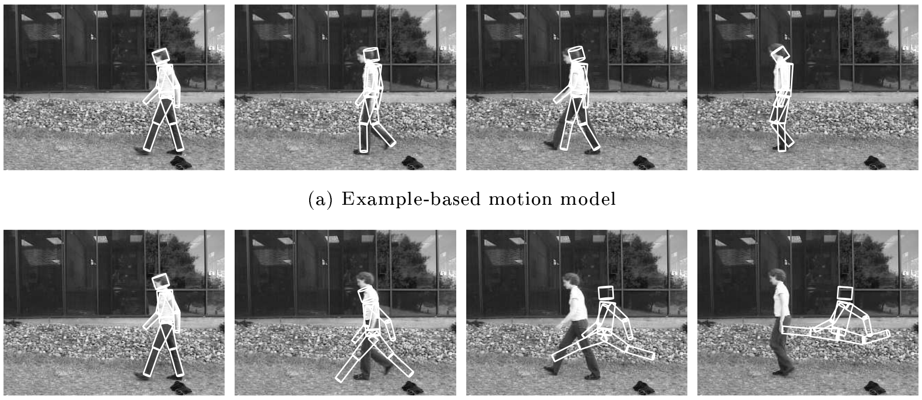
Fig. 6. Tracking an arm using 300 samples. Frames 5, 10, 15, 20, 25 and 30, with the expected value of the posterior distribution overlaid, are shown. (a) The example based model enables tracking with a small number of samples. (b) The same number of samples is insufficient using the constant velocity model.

$N \propto 1/\alpha^d$ where $\alpha \ll 1$ depends on the generality of the motion model, and d is the number of parameters [23]. The use of a strong prior model such as the one developed here is one way of coping with the dimensionality problem. See [38] for recent work on human motion tracking that places particles more effectively thus allowing more general motion priors.

As an initial illustration, a version of the database is built using only the joint angles and angular velocities of the right arm. The model has 8 DOF; 3 Euler angles in the shoulder, one angle in the elbow, and their angular velocities. The sequence used for tracking is stabilized so that the shoulder has the same position in all images and the configuration of the arm at frame 0 is manually set. To initialize the example-based temporal model, a linear search is performed in the database, and N arm motion samples, whose last time step correspond well with the manually set joint angles, are chosen using Monte Carlo sampling.

In Figure 6a, the arm is tracked with the example-based motion model as the temporal prior. $N = 300$ samples are used for particle filtering. In frame 15, the elbow of the model arm is more bent than the real elbow in the sequence. The reason for this is that the real arm position is not present in the database – none of the example subjects moved their arm to this position. This may be an artifact of the marker placement during motion capture. However, in frame 20, the arm is again correctly estimated. A typical set of particles is illustrated in Figure 3e. The variation in the set of sampled poses can be controlled by the temperature parameter β, as discussed in Section 4.

For comparison, the arm is also tracked using the constant velocity prior. The number of particles, N (and the likelihood model) are the same as in the previous tracking case, i.e. much lower than the number of particles used in similar experiments [36]. Since the constant velocity model is not able to predict the arm motion as well as the example-based model, the arm model loses track



(a) Example-based motion model

(b) Constant velocity model

Fig. 7. Tracking a walking human using 300 samples. Frames 5, 10, 15, 20, 25 and 30, with the expected value of the posterior distribution overlaid, are shown. (a) Even though the dimensionality is higher than in Figure 6 the example based model enables tracking with the same number of samples. (b) This number of samples is, again, insufficient using the constant velocity model.

after a few frames, due to the small number of particles. However, if the number of particles is raised to $N = 3000$, the constant velocity prior is sufficient.

If the number of parameters d is larger, the constant velocity model is too general (i.e. α is too low). Given that $N = 3000$ is sufficient to track an 8-dimensional arm model using this motion model, the relation $N \propto 1/\alpha^d$ gives an estimate of the number of particles needed to track a 50-dimensional model such as human. Using this estimate, $N \approx 10^{16}$.

The result of tracking a full 3D body using the example-based model is shown in Figure 7a. $N = 300$ particles were used for tracking. Since the database of motions is quite small ($n \approx 50000$), and the examples are taken from the motion of professional dancers (a quite biased selection in terms of motion pattern), the deviations in pose between the model and the human in the sequence is at times large. Apart from the pose deviations, the example-based motion model enables successful tracking of the person in this 30 frame sequence. This is in contrast to the smooth motion model for which $N = 300$ particles is far from sufficient (Figure 7b). Note that the example-based motion model is more versatile than learned models of walking [17, 34, 37] in that it can be used for all kinds of motion (present in the database), as well as transitions between activities. Furthermore, some generalization to novel motions is made possible by the addition of Gaussian noise to the samples.

7 Conclusions

Learning a concise probabilistic model of 3D human motion is a challenging task (though recent work [4] suggests that it is possible). Here we make the simple observation that a database of motion capture data can, itself, serve as an implicit probabilistic model in a way that is directly analogous to recent work on texture synthesis. Unlike work on texture synthesis, our goal is human tracking and, hence, we must model general human motion rather than narrow "textural" motions. This results in a large database of heterogeneous motions. Consequently, practical algorithms for either synthesis or tracking necessitate an efficient algorithm for searching the database. We have proposed a method for structuring the database as a binary tree based on the coefficients of a low-dimensional approximation to the data. Furthermore, we described a probabilistic search method that uses the binary tree to stochastically generate sample motions from the database that match a synthesized input motion. The algorithm has sub-linear complexity and is suitable for synthesizing novel motion sequences and for generating predictions in a Bayesian tracking framework. Further work needs to be done however to understand how well this heuristic search method approximates the true distribution, how it will scale to larger databases, and whether it can be applied to other problems with different underlying densities.

One advantage of this method over traditional learning methods is that it uses all the data and hence captures subtle variations that may be lost using other methods. It is also extremely simple to implement. The disadvantage however is that the representation does not readily generalize to new motions. To allow more variation in the generated motion, and to lower the number of needed example motion sequences, the method described here could be used to learn separate motion models for different parts of the body (as was illustrated with the arm tracking example in Section 6 and Figure 6). The natural tree structure of the body can be exploited in sampling the motions of the joints. More research is needed to model the correlations between these limbs and to perform Bayesian inference using such a model.

While our tracking results are preliminary, they suggest that an implicit motion model may prove effective for tracking high dimensional human motion. To reduce the problems with extrapolation from the database, the example-based motion priors could be combined with more general priors using mixture models. In a particle framework, this would correspond to propagating a certain portion of the particles with a general temporal prior, and another portion with the example-based temporal prior.

The model also shows promise as a method for generating varied synthesized motion with natural detail. However, to exploit the example-based method for synthesis, further work must be done. The most obvious extension is to perform some simple blending or smoothing of the synthesized poses [35, 31], and to introduce a predictive mechanism to avoid "dead ends" [35]. The issue of retargetting the synthesized motions to new actors remains open [11, 20, 31] as does the application of these techniques to facial motions or other time series data. Finally, realistic synthesis will require the addition of constraints such as contact between feet and the ground plane and, more generally, controllability/editability for animation [11]. The formulation of such constraints within our sampling framework seems plausible (cf. [6]) but remains an area for future research.

Acknowledgments. We thank Michael Gleicher for providing the 3D motion capture database and Yaser Yacoob for the walking image sequence used for the arm tracking experiment. We also thank Nancy Pollard, Fernando De la Torre, and Thomas Hofmann helpful discussions on motion synthesis, and Jan-Olof Eklundh for his support and encouragement.

HS was sponsored by the Foundation for Strategic Research under the "Center for Autonomous Systems" contract. MJB was supported by the DARPA HumanID Project (ONR contract N000140110886) and by a gift from the Xerox Foundation.

References

1. J. Bar-Joseph, R. El-Yaniv, D. Lischinski, and M. Werman. Texture mixing and texture movie synthesis using statistical learning. *IEEE Trans. Visualization and Computer Graphics*, 7(2):1120–135, 2001.
2. J. S. Beis and D. G. Lowe. Shape indexing using nearest-neighbour search in high-dimensional spaces. *CVPR*, pp. 1000–1006, 1997.
3. M. Brand. Shadow puppetry. *ICCV*, vol. 2, pp. 1237–1244, 1999.
4. M. Brand and A. Hertzmann. Style machines. *SIGGRAPH*, pp. 183–192, 2000.
5. A. Bruderlin and T. W. Calvert. Intelligence without representation. *Computer Graphics*, 23:233–242, 1989.
6. S. Chenney and D. A. Forsyth. Sampling plausible solutions to multi-body constraint problems. *SIGGRAPH*, pp. 219–228, 2000.
7. J. de Bonet. Multiresolution sampling procedure for analysis and synthesis of texture images. *SIGGRAPH*, pp. 361–368, 1997.
8. A. A. Efros and W. T. Freeman. Image quilting for texture synthesis and transfer. *SIGGRAPH*, pp. 341–346, 2001.
9. A. A. Efros and T. K. Leung. Texture synthesis by non-parametric sampling. *ICCV*, vol. 2, pp. 1033–1038, 1999.
10. A. Fitzgibbon. Stochastic rigidity: Image registration for nowhere-static scenes. *ICCV*, vol. 1, pp. 662-669, 2001
11. M. Gleicher. Retargeting motion to new characters. *SIGGRAPH*, pp. 33–42, 1998.
12. L. Goncalves, E. Di Bernardo, and P. Perona. Reach out and touch space (motion learning). *IEEE International Conference on Automatic Face and Gesture Recognition*, pp. 234–238, 1998.
13. N. Gordon. A novel approach to nonlinear/non-gaussian Bayesian state estimation. *IEE Proceedings on Radar, Sonar and Navigation*, 140(2):107–113, 1993.
14. P. Harrison. A non-hierarchical procedure for re-synthesis of complex textures. *WSCG*, 2001.
15. D. J. Heeger and J. R. Bergen. Pyramid-based texture analysis/synthesis. *SIGGRAPH*, pp. 229–238, 1995.
16. A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. H. Salisin. Image analogies. *SIGGRAPH*, pp. 327–240 2001.
17. D. C. Hogg. Model-based vision: A program to see a walking person. *IVC*, 1(1):5–20, 1983.
18. M. Isard and A. Blake. Condensation – conditional density propagation for visual tracking. *IJCV*, 29(1):5–28, 1998.
19. M. Isard and A. Blake. ICondensation: Unifying low-level and high-level tracking in a stochastic framework. *ECCV*, vol. 1, pp. 893–908, 1998.

20. J. K. Hodgins and N. S. Pollard. Adapting simulated behaviors for new characters. *SIGGRAPH*, pp. 153–162, 1997.
21. C. E. Jacobs, A. Finkelstein, and D. H. Salesin. Fast multiresolution image querying. *SIGGRAPH*, pp. 277–286, 1995.
22. M. E. Leventon and W. T. Freeman. Bayesian estimation of 3-d human motion from an image sequence. MERL Tech. Rep. TR–98–06, Cambridge, MA, 1998.
23. J. MacCormick and M. Isard. Partitioned sampling, articulated objects, and interface quality hand tracking. *ECCV*, vol. 2, pp. 3–19, 2000.
24. J. McNames. A fast nearest-neighbor algorithm based on a principal axis search tree. *PAMI*, 23(9):964–976, 2001.
25. B. Moghaddam and A. Pentland. Probabilistic visual learning for object representation. *PAMI*, 19(7):696–710, 1997.
26. L. Molina-Tanco and A. Hilton. Realistic synthesis of novel human movements from a database of motion capture examples. *IEEE Workshop on Human Motion*, pp. 137–142, 2000.
27. H. Murase and S. Nayar. Visual learning and recognition of 3-D objects from appearance. *IJCV*, 14:5–24, 1995.
28. S. A. Nene and S. Nayar. A simple algorithm for nearest neighbor search in high dimensions. *PAMI*, 19:989–1003, 1997.
29. V. Pavolvić, J. Rehg, T-J. Cham, and K. Murphy. A dynamic Bayesian network approach to figure tracking using learned dynamic models. *ICCV*, vol. 1, pp. 94–101, 1999.
30. K. Popat and R. W. Picard. Cluster-based probability model and its applications to image and texture processing. *IEEE Trans. Image Proc.*, 6(2):268–284, 1997.
31. Z. Popovic and A. Witkin. Physically Based Motion Transformation. *SIGGRAPH*, pp. 11–20, 1999.
32. J. Portilla and E. P. Simoncelli. A parametric texture model based on joint statistics of compex wavelet coefficients. *IJCV*, 40(1):49–71, 2000.
33. K. Pullen and C. Bregler. Animating by multi-level sampling. *IEEE Computer Animation*, pp. 36–42, 2000.
34. K. Rohr. Towards model-based recognition of human movements in image sequences. *CVGIP - Image Understanding*, 59(1):94–115, 1994.
35. A. Schödl, R. Szeliski, D. H. Salesin, and I. Essa. Video textures. *SIGGRAPH*, pp. 489–498, 2000.
36. H. Sidenbladh and M. J. Black. Learning image statistics for Bayesian tracking. *ICCV*, vol. 2, pp. 709–716, 2001.
37. H. Sidenbladh, M. J. Black, and D. J. Fleet. Stochastic tracking of 3D human figures using 2D image motion. *ECCV*, vol. 2, pp. 702–718, 2000.
38. C. Sminchisescu and B. Triggs. Covariance scaled sampling for monocular 3D body tracking. CVPR, vol. 1 pp. 447–454, 2001.
39. S. Soatto, G. Doretto, Y. Wu. Dynamic Textures. *ICCV*, pp. 439–446, 2001.
40. L-Y. Wei and M. Levoy. Fast texture synthesis using tree-structured vector quantization. *SIGGRAPH*, pp. 479–488, 2000.
41. C. R. Wren and A. P. Pentland. Dynaman: Recursive modeling of human motion. *IVC*, to appear.
42. Y. Yacoob and M. J. Black. Parameterized modeling and recognition of activities in temporal surfaces. *CVIU*, 73(2):232–247, 1999.
43. Y. Yacoob and L. Davis. Learned models for estimation of rigid and articulated human motion from stationary or moving camera. *IJCV*, 36(1):5–30, 2000.
44. S. C. Zhu, Y. N. Wu, and D. Mumford. Minimax entropy principle and its application to texture modeling. *Neural Computation*, 9:1627–1660, 1997.