Simultaneous Pose Estimation and Camera Calibration from Multiple Views

Tomas Izo W. Eric L. Grimson

Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139
{*tomas, welg*}*@csail.mit.edu*

Abstract

We present an algorithm to estimate the body pose of a walking person given synchronized video input from multiple uncalibrated cameras. We construct an appearance model of human walking motion by generating examples from the space of body poses and camera locations, and clustering them using expectation-maximization. Given a segmented input video sequence, we find the closest matching appearance cluster for each silhouette and use the sequence of matched clusters to extrapolate the position of the camera with respect to the person's direction of motion. For each frame, the matching cluster also provides an estimate of the walking phase. We combine these estimates from all views and find the most likely sequence of walking poses using a cyclical, feed-forward hidden Markov model. Our algorithm requires no manual initialization and no prior knowledge about the locations of the cameras.

1. Introduction

As seamless human-machine interfaces and automatic monitoring systems become more feasible, tracking and analysis of human motion will be crucial, since much of activity in indoor settings is comprised of moving people. Some of the most difficult challenges in extracting 3D information from observations of human motion stem from the complex kinematics of the human body and the inherent ambiguities in projected 2D observations of a 3D world. Researchers have dealt with these difficulties by constraining the problem to reduce the size of the parameter search space. This often results in tracking or body pose estimation systems that are specific to a particular camera setup or require non-automatic initialization, such as asking a human to identify certain features in the image.

In this paper, we present a system for extracting 3D pose information from observed 2D walking motion sequences in a manner invariant to the number and positions of the cameras. In essence, the task is equivalent to showing a series of images depicting a walking person and asking a human to identify the body pose shown in each image, for instance by shaping and orienting a three-dimensional model, as in figure 1.

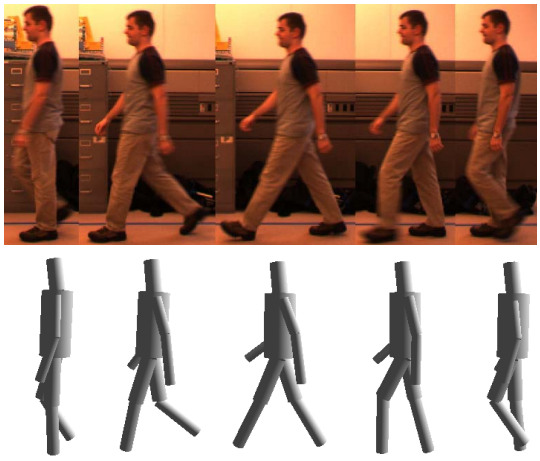

Figure 1. An example of a walking human in several phases of the walking cycle (top) matched against the most closely corresponding phases of a simple OpenGL stick-figure model (bottom). Here, the matching was done by a human observer. Our objective is to be able to do the same automatically using the added advantage of multiple points of view.

2. Related Work

Much of the previous research in articulated tracking has been concerned with a single camera view. Cham and Rehg used 2D Scaled Prismatic Models together with a multiple hypothesis approach to track human motion [3]. Bre-

Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW'04)
1063-6919/04 $ 20.00 IEEE

gler and Malik [2] represented 3D motion of articulated body parts by kinematic chains–series of twists (rotations and translations) of individual rigid parts around connecting joints. Their approach depended on manual initialization in the first frame. Howe, Leventon and Freeman [7] used a database of 3D motion snippets to estimate the depth of a number of manually initialized points by finding the maximum likelihood estimate for the motion snippet corresponding to a short sequence of frames from the input.

Other research groups have employed multiple camera setups. Mikic et al. [10] used four calibrated cameras to extract a volumetric model of the observed person. They then located the major body parts sequentially using volumetric templates. Cohen et al. [4] used correspondence between the medial axes of synchronized pairs of input silhouettes to estimate the epipolar geometry and subsequently fit to the silhouettes a 3D generalized cylinder. Grauman et al. [6] used a set of calibrated cameras to infer 3D joint positions using a learned "shape+structure" model specific to the camera setup.

A common assumption in multiple view setups is that the cameras can be easily calibrated, i.e. at least their relative positions can be well established. Eliminating this requirement would make the resulting algorithms more flexible and consequently more easily applicable in real world scenarios.

Finally, much of published research dealing with the tracking of human motion assumes that some sort of manual or semi-automatic initialization of the algorithm can be performed by the user.

3. Algorithm Overview

We tackle the problem of simultaneous camera calibration and walking pose estimation in two stages. First we construct a model of the appearance and dynamics of a human walk. We use commercial software to separate the walking cycle into a number of phases, and then sample from the 3D space around the person to generate examples of the 2D appearance of each walking phase. We cluster these examples into groups to make the appearance model more compact. We model the periodic dynamics of the walking cycle using a first-order, feed-forward, cyclical hidden Markov model (HMM).

Second, we develop an algorithm to fit this walking motion model to video sequences from multiple synchronized cameras. For each camera input, we apply a background subtraction algorithm to obtain a sequence of silhouettes for that view point. We then use a distance metric to find the appearance model cluster that best matches each silhouette. Given these matches, we infer the position of each camera with respect to the person. Finally, we use the matches from all cameras as input into the dynamic HMM model in order to hypothesize the walking phase at each frame in the input sequences.

With these appearance and dynamics models in place, our algorithm takes as input synchronized video sequences of a walking person from multiple views and outputs a 3D body pose estimate for each input frame and an estimated location for each of the cameras with respect to the person's direction of motion. The algorithm works as follows:

(1) Consider a system with n cameras, which collect n sequences of images, S_1 through S_n, where $S_i = I_{i1}...I_{iT}$.

(2) Each image I_{ij} is segmented using a background subtraction method. This produces n sequences of binary silhouettes: S'_1 through S'_n, where $S'_i = s_{i1}...s_{iT}$.

(3) Each silhouette s_{ij} is compared to the mean of each cluster in the appearance model using a distance metric. The cluster with the smallest distance is selected, thus producing n sequences of cluster matches: C_1 through C_n, where $C_i = c_{i1}...c_{iT}$.

(4) The camera positions of all of the training examples belonging to the clusters in sequence C_i form a cluster in the camera pose space. We fit an ellipse to this cluster and take the center of the ellipse to be the camera pose estimate for the i-th camera. If the appearance of the person is similar to two opposite viewpoints, we may observe two distinct camera pose clusters, in which cases we select the one that corresponds to the direction of the person's motion as observed in the image. Estimating the camera pose in this fashion reduces the effect of outliers on the result. Figure 2 shows the camera pose cloud for a typical input sequence. We assume that the input sequences are short enough so that the change in camera position with respect to the person is small. However, longer input sequences can be broken down into short segments that satisfy this assumption.

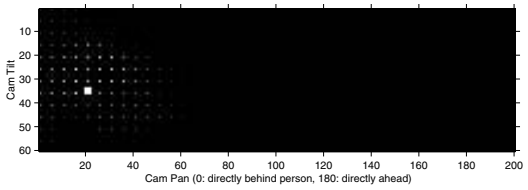

Figure 2. Distribution of camera poses over all of the examples in the matching cluster sequence corresponding to the input. Brighter dots indicate more examples with the corresponding camera position. The bright square indicates the correct answer.

(5) For each cluster in the sequence C_i, we compute the average walking phase over all of the training examples in that cluster. In this fashion, we obtain the sequence $\Psi_i = \psi_{i1}...\psi_{iT}$ of phase estimates for each camera. The se-

quences Ψ_1 through Ψ_n serve as inputs to our HMM, which outputs the most likely sequence of walking poses. The pose estimates correspond to 3D joint locations of our synthetic model, which we can then render from any view.

4. Modeling Motion Appearance

A view-independent appearance model of walking motion needs to sufficiently characterize those properties of the appearance that vary with time as well as the position of the observer. For a walking human, these are the body pose, i.e. the configuration of body parts at any given time as the motion progresses, and the camera pose, i.e. the position of the observer with respect to the walking person. The most straightforward motion appearance model that accounts for variations in these parameters is a collection of sample observations chosen so that the entire range of each parameter is well represented. To generate such a data set under controlled conditions, we used Curious Labs' Poser 5, a software package capable of modeling the human body in an anatomically correct fashion, to generate examples as follows:

(1) We divide the full walking cycle, i.e. the space $\mathcal{B}$ of body poses, uniformly in time into 20 phases, denoted ψ_i. A typical human walking cycle takes between 1 and 2 seconds to complete. Our recording equipment operates at 15Hz, and thus typically produces around 20 images from a single walking cycle.

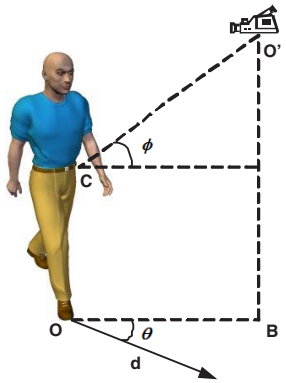

Figure 3. The camera position is determined by the pair (θ, ϕ), where θ is relative to the direction of motion and ϕ is relative to the ground-parallel plane crossing the centroid of the person.

(2) We define the camera pose relative to the person's direction of motion (see figure 3). Let θ be the angle between the direction of walking and the line connecting the projections of the person's centroid and the camera location onto the ground plane. Further let ϕ be the angle between the plane parallel to the ground plane and intersecting the person's centroid and the line connecting that centroid and the camera. The camera pose is the pair (θ, ϕ). We do not explicitly model the distance of the camera from the observed person; instead, we scale-normalize all input silhouettes to be of the same height and we keep track of the scaling factor. All lenghts extracted by our algorithm for a given input sequence are thus relative to other input sequences.

For our data set, we set the following conditions on the space $\mathcal{L}$ of camera poses:

$$\theta \in (0, 2\pi); \quad \phi \in (-\tfrac{1}{9}\pi, \tfrac{2}{9}\pi)$$

We restrict the range of ϕ to ensure that the view of the person is informative: views from extreme camera locations above and below the person may not provide enough information about the body pose. We sample from each of these ranges at intervals of $\frac{1}{36}\pi$, resulting in 864 different camera positions uniformly distributed over the space of all allowed camera poses.

Sampling from the space $\mathcal{B} \times \mathcal{L}$ results in 17,280 different examples. Example i of the data set consists of the quadruple $(\boldsymbol{s}, \psi, \theta, \phi)_i$, where the latter three parameters are the body phase and camera phase and $\boldsymbol{s}$ is the silhouette of the human body rendered in the given phase as seen from the given camera position. Additionally, for each example we save the vector $\boldsymbol{x}$ consisting of the corresponding 2D coordinates of 13 main 3D body joints of the model projected into the image. We use silhouettes rather than images because they are easy to extract from the input using motion segmentation techniques and they are invariant to color, pattern of clothing, color of skin, lighting of the scene, and many other parameters. While two images of different people in the same body pose as observed from the same view might look quite different, the corresponding silhouettes would look similar.

In essence, given a set of observations (i.e., silhouettes) $\boldsymbol{S}$, we are looking for the maximum likelihood estimate of the parameter vector $\boldsymbol{P} = (\psi, \theta, \phi)$:

$$\hat{\boldsymbol{P}} = \underset{\boldsymbol{P}}{\operatorname{argmax}}(L(\boldsymbol{S}|\boldsymbol{P})) \tag{1}$$

where $L(\boldsymbol{S}|\boldsymbol{P})$ is the likelihood of the data (silhouettes) given the model parameters P.

4.1. Clustering Examples Using EM

An appearance model consisting of a collection of examples is impractical because finding the maximum likelihood values for the parameters amounts to comparing an observation to each of the ca. 18,000 examples. We make the appearance model more compact by grouping the examples into clusters. While using the silhouettes as the criteria

for the clustering would be the most straightforward solution, it would make the maximum-likelihood estimation of the cluster parameters computationally expensive due to the large feature vectors and covariance matrices. We instead take advantage of higher order information, namely the 2D locations of the major body joints which led to the corresponding rendering of the silhouettes (see also [11]).

The goal is to assign to each 2D joint coordinate vector $\boldsymbol{x}$ a label $l, 1 < l < N$ classifying that feature vector as belonging to cluster C_l, where N is the number of clusters. To further constrain the problem, we assume that the cluster densities can be approximated by multivariate Gaussians, i.e. the probability of a feature vector $\boldsymbol{x}_i$ is

$$p(\boldsymbol{x}_i|\boldsymbol{\Theta}) = \sum_{j=1}^{N} \frac{P(j)}{(2\pi)^{d/2}|\boldsymbol{\Sigma}_j|^{1/2}} e^{-\frac{1}{2}(\boldsymbol{x}_i-\boldsymbol{\mu}_j)^T\boldsymbol{\Sigma}_j^{-1}(\boldsymbol{x}_i-\boldsymbol{\mu}_j)} \quad (2)$$

where $P(j)$ are the priors, $\boldsymbol{\Sigma}_j$ and $\boldsymbol{\mu}_j$ are the covariance matrix and mean of the j-th cluster, and $\boldsymbol{\Theta}$ is a vector encompassing the parameters of all N Gaussian distributions.

Given this assumption the expectation-maximization algorithm leads to the following update equations [1]:

$$P^t(j) = \frac{1}{N}\sum_{i=1}^{n} p(j|\boldsymbol{x}_i, \boldsymbol{\Theta}^{t-1}) \quad (3)$$

$$\boldsymbol{\mu}_j^t = \frac{\sum_{i=1}^{n} \boldsymbol{x}_i p(j|\boldsymbol{x}_i, \boldsymbol{\Theta}^{t-1})}{\sum_{i=1}^{n} p(j|\boldsymbol{x}_i, \boldsymbol{\Theta}^{t-1})} \quad (4)$$

$$\boldsymbol{\Sigma}_j^t = \frac{\sum_{i=1}^{n} p(j|\boldsymbol{x}_i, \boldsymbol{\Theta}^{t-1})(\boldsymbol{x}_i - \boldsymbol{\mu}_j^t)(\boldsymbol{x}_i - \boldsymbol{\mu}_j^t)^T}{\sum_{i=1}^{n} p(j|\boldsymbol{x}_i, \boldsymbol{\Theta}^{t-1})} \quad (5)$$

where the superscript t indicates the value of the corresponding variable at time (iteration) t.

Our clustering algorithm works as follows:

(1) We initialize the computation by "guessing" the first estimate of the parameter vector, $\boldsymbol{\Theta}^0$. We randomly select N examples to serve as estimates for the mean vectors $\boldsymbol{\mu}_j^0$. We cluster the remaining examples into N groups by assigning example $\boldsymbol{x}_i$ to cluster j whenever $j = \text{argmin}_k(|\boldsymbol{x}_i - \boldsymbol{\mu}_k^0|)$. We then calculate the covariance matrix for each of the resulting clusters, thus obtaining estimates for $\boldsymbol{\Sigma}_j^0$. Finally, the prior estimates $P(j)$ are set to the ratio of the size of the corresponding cluster to the size of the entire data set.

(2) We iteratively update the triple $(P(j), \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$ using equations 3, 4 and 5.

(3) We repeat step 2 until the change in the estimates for the mean vectors $\boldsymbol{\mu}_j$ falls below a threshold ϵ:

$$\sum_{j=1}^{N} ||\boldsymbol{\mu}_j^t - \boldsymbol{\mu}_j^{t-1}|| < \epsilon$$

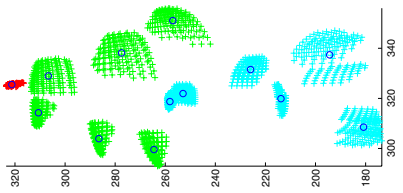

Figure 4. A plot of one of the appearance model clusters. Points represent the body joint locations projected into the image for all of the examples in the cluster. Circles show the superimposed mean of the cluster.

4.2. Number of Clusters

To determine the optimal number of clusters, we performed a faster, k-means clustering for different values for k and analyzed the resultant partitioning using scatter criteria. Define the within-scatter matrix as [5]:

$$\boldsymbol{S}_W = \sum_{j=1}^{N}\sum_{\boldsymbol{x}\in C_j} (\boldsymbol{x} - \boldsymbol{\mu}_j)(\boldsymbol{x} - \boldsymbol{\mu}_j)^T \quad (6)$$

where C_j denotes the j-th cluster and μ_j is its mean. Intuitively, the within-scatter matrix is a measure of tightness of clusters. A common way of evaluating the scatter matrix is by its trace, i.e. the sum of the diagonal elements:

$$tr[\boldsymbol{S}_W] = \sum_{j=1}^{N}\sum_{\boldsymbol{x}\in C_j} ||\boldsymbol{x} - \boldsymbol{\mu}_j||^2 \quad (7)$$

We can see from equation 7 that the trace of the within-scatter matrix is simply the sum-of-squared-errors for the clustering. Thus we should want to minimize this quantity.

Experiments show that the best value for k in terms of the trace criterion is 100, since the within-scatter is smallest for this number of clusters. We used this number of clusters for the EM algorithm outlined above.

To demonstrate the quality of the clustering, Figure 4 shows a plot of one of the clusters with all of its members superimposed and its mean highlighted. We can see that the examples in this group form a relatively tight cluster around the mean. Despite the tightness of the clusters, however, the cluster means are fairly evenly distributed in the camera pose space (Figure 5). Consequently, matching clusters to incoming silhouettes will provide us with information about the position of the camera with respect to the person.

5. Matching Silhouettes to Clusters

We represent each cluster by its mean silhouette. The value of a pixel in the mean silhouette corresponds to the

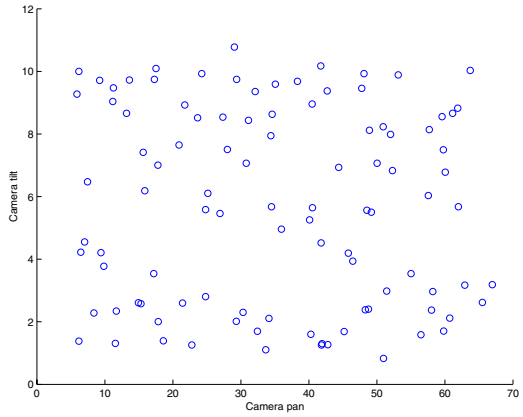

Figure 5. The distribution of mean camera poses. Each circle corresponds to the average camera position for a cluster of body poses. Axis labels are angles in degrees divided by 5.

probability that the pixel is 'on' in a silhouette picked randomly from the cluster.

To compare the binary input silhouette to the continuous cluster mean, we first normalize the two to be of the same height using isotropic scaling and then register them by aligning their centroids and principal axes. We define the distance measure on the registered silhouettes as follows. First let s_b, s_c be the binary and continous silhouette, respectively. Then let $s_1 = s'_b.s_c$ and $s_2 = s_b.s'_c$. Here s'_b and s'_c denote the "inverted" silhouettes, i.e. every non-zero pixel becomes 0 and every pixel with value 0 becomes 1. The operator '.' denotes scalar multiplication on the silhouettes represented as matrices of pixels. Then let the distance between the silhouettes s_b, s_c be the function

$$d(s_b, s_c) = sum(max(s_1, s_2)) \tag{8}$$

where the maximum is taken for each pixel pair. This distance is similar to the Hamming distance (number of pixels to be flipped to change one silhouette into the other), where each flipped bit is weighted by the number of examples in the cluster that differed from the binary silhouette.

6. Modeling Motion Dynamics with HMMs

Discrete Markov Models (MMs) are useful for describing systems that can be characterized as being at any given time in one of a finite number of distinct states. First-order MMs take advantage of the further constraint that the next state depends only on the current state of the system. Suppose that we discretize the human walking motion into p distinct, sequential phases, $\psi_1, ..., \psi_p$. The probability of the person being in phase ψ_j at time $t+1$ depends only on the state (i.e. phase) at time t. Thus, we can use a first-order discrete Markov model to describe the characteristic periodic progression through the phases of the walking cycle. Since, however, the state of the system (i.e. the current walking cycle phase) is not directly observable, we use a first-order Hidden Markov model. At each time instance t, the HMM generates an "observation," with some probability distribution $p_{q(t)}$, where $q(t)$ is the current state at time t. Our observations are generated by the camera pose estimation step of our algorithm. For an input sequence of T consecutive frames from c cameras, the camera pose estimation step outputs, in addition to the estimated camera position, the sequence of walking phase estimates $\mathbf{\Psi} = \boldsymbol{\psi}_1, ..., \boldsymbol{\psi}_T$ with two properties. First, each observation $\boldsymbol{\psi}_i = [\psi_{i1} ... \psi_{ic}]$ is a vector consisting of a walking phase estimate corresponding to each of the c cameras. Second, the walking phase estimates ψ_{ij} are either close to the true walking phase for that frame, or close to its *opposite* phase. If a full walking cycle were divided into p parts, the opposite of the i-th phase would be phase j, where $j = (i + p/2) \bmod p$. Opposite phases sometimes have very similar appearance.

Using this information and our knowledge of the motion, we use the following HMM to model the progression through the phases of a human walking cycle (figure 6):

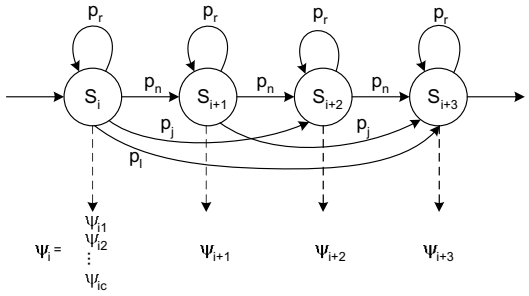

Figure 6. A cyclical, feed-forward hidden Markov model describing the progression of estimated walking phases from multiple views.

(1) There are p hidden states, each corresponding to a walking phase. We use $p = 20$, which is the number of walking phases in our appearance model.

(2) The HMM has a cyclical structure corresponding to the periodic nature of the motion. Moreover, the model can transition from state to state only in the forward direction, since we are only concerned with forward motion. To accomodate various speeds of motion, it is also possible for the model to remain in the same state or skip up to two states. These restrictions lead to the following properties of

the transition matrix $A = (a_{ij})$:

$$a_{ij} = \begin{cases} p_r & \text{if } j = i \\ p_n & \text{if } j = (i+1) \bmod p \\ p_j & \text{if } j = (i+2) \bmod p \\ p_l & \text{if } j = (i+3) \bmod p \\ 0 & \text{otherwise} \end{cases}$$

where a_{ij} is the probability of transition from state i to state j. We set the values for p_r, p_n, p_j and p_l based on experimentation with various data sequences. The best results were usually produced with $p_r < p_n$, which pushes the model towards moving to the next state rather than staying in the same one. We also set p_j to be only slightly smaller than p_n, since it is quite common to skip states. Similarly, the probability of skipping two states is slightly smaller than the probability of skipping just one. We use the same parameter values for every sequence. However, it is possible to automatically adjust these parameters based on the speed and frequency of the motion, both of which can be observed in the input sequence.

(3) Because all of the phases of the walking cycle are equally likely to be the first in the incoming video sequence, we chose a uniform prior distribution on the initial state.

(4) Each state in the model produces its output by drawing a sample from a continous observation density that models the fact that opposite phases of the walking cycle have similar appearance and, consequently, that a hidden state of the HMM can generate an observation in which some of the observed phases are opposite to the true phase. Thus, we use a mixture density with 2^c components, where c is the number of cameras. For state i, the k-th component of the mixture is a Normal distribution $\mathcal{N}_{ik}(\boldsymbol{\mu}_{ik}, \boldsymbol{\Sigma}_{ik})$:

$$\boldsymbol{\mu_{ik}} = [\mathrm{f}(i, e_1), ..., \mathrm{f}(i, e_c)], \text{ where } (e_1 e_2 ... e_c) = (k)_2$$

$$f(i, e_j) = \begin{cases} i & \text{if } e_j = 0 \\ (i + p/2) \bmod p & \text{if } e_j = 1 \end{cases}$$

$$\boldsymbol{\Sigma}_{ik} \text{ is diagonal}$$

The first two properties require some further explanation. Take $(e_1 e_2 ... e_c)$ to be the binary representation of k (with any necessary leading zeros). The mean $\boldsymbol{\mu}_{ik}$ then consists of the corresponding arrangement of either phase i or its opposite phase for the observation in each camera. We have 2^c mixture components because there are 2^c such possible arrangements. We choose a diagonal covariance matrix because the elements of the mean vector, corresponding to individual cameras, should be independent and have equal variance. With these modifications, hidden states generate visible states according to the following relationship:

$$P(v(t) = \boldsymbol{\psi}_j | q_i(t)) = \sum_{k=1}^{2^c} \frac{1}{2^c} \mathcal{N}_{ik}(\boldsymbol{\mu}_{ik}, \boldsymbol{\Sigma}_{ik})(\boldsymbol{\psi}_j). \quad (9)$$

where $v(t) = \boldsymbol{\psi}_j$ denotes the event that the visible state generated at time t was $\boldsymbol{\psi}_j$.

Given this model, we use the Viterbi algorithm [13] to generate the most likely sequence of states given our observations from multiple cameras. Combined with the estimated position of each camera with respect to the subject, this generates a view of the 3D model in the estimated body configuration as viewed from the estimated angle.

7. Experiments

To collect data, we used a pair of Point Grey Dragonfly cameras. Each of the cameras ran at 15Hz and produced images of 640x480 pixels. The cameras were automatically synchronized so that image acquisition in the two cameras took place within at most 20μs.

We segmented each incoming image using the adaptive background subtraction technique from [9], which is based on the method from [12]. We first used 100 frames of the scene without the person or any other moving objects to construct a probabilistic model of the background. We then used the method from [9] to assign to each pixel the binary value 1 whenever that pixel was classified as foreground, i.e. belonging to the person and the value 0 otherwise, thus converting each image sequence into a sequence of silhouettes. The size of the cropped silhouettes was roughly 120x250 pixels.

Figure 8 and 9 show the results of the body pose estimation part of our algorithm for two synchronized cameras observing the same person walking through a room. The top row shows the input silhouettes, the second row shows the best matching cluster and the third row shows the example from our model that corresponds to the camera pose and walking phase estimated by our algorithm. The results from both views closely match the input. The fourth row of figure 8 also shows silhouettes generated by rendering the 3D model from a different viewpoint to demonstrate that the extracted information is, in fact, 3D.

To analyze the accuracy of the camera pose estimation algorithm, we generated 100 synthetic sequences of walking people by varying the walking style parameters (leg lift, arm swing and stride) as well as the size of the person and location of the camera. Figure 7 shows that the majority of the camera pose estimates fell within 15 degrees from the correct answer.

The most computationally intensive part of our algorithm is the construction of the appearance model, which is only performed once. The background segmentation can be optimized to run in real time. The remaining steps of the algorithm take approximately 20 seconds per frame and camera in our Matlab implementation on a 2.0 GHz Pentium 4.

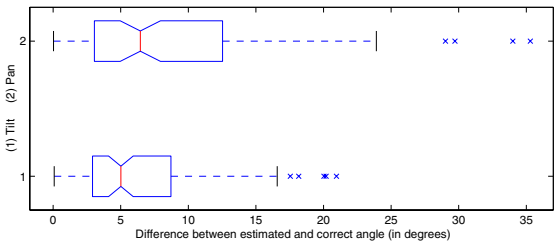

Figure 7. Distribution of error in the camera pose estimation for 100 randomized sequences. Center lines in boxes indicate the median, outer lines delimit the uppper and lower quartiles. Dashed lines extend to delimit the range of the data, with the outliers shown as 'x'.

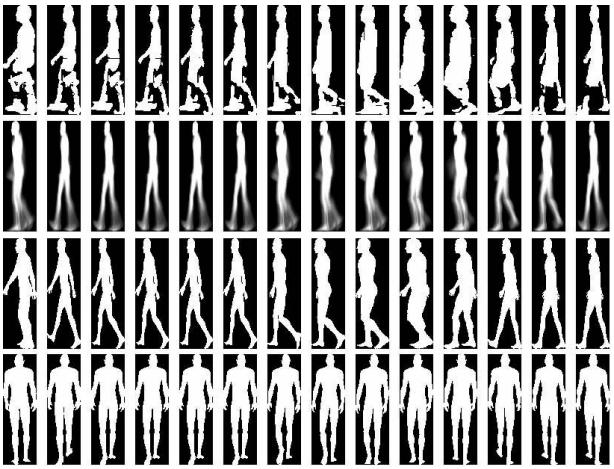

Figure 8. Top row: input silhouettes from the first camera angle of the recorded test sequence. Second row: best matching clusters. Third row: examples generated using the estimated camera position and body pose. Bottom row: 3D model rendered from a new viewpoint.

7.1. Using feedback to improve silhouettes

One of the most immediate uses of the output of our algorithm is to improve the quality of the silhouettes obtained from background subtraction. Figure 10 shows the result of using the extracted higher level information to improve silhouettes. To obtain the improved silhouettes, we took the synthesized renderings of our model in the extracted poses, registered them with the original silhouettes and used them to "force" the corresponding pixels to become foreground with higher likelihood than in the straight-forward adaptive backgrounding scheme. The resulting silhouettes appear to have many fewer missing parts. Since it has been shown that better silhouettes lead to higher recognition rates in identification by gait [8], our system could be used to further improve these rates.

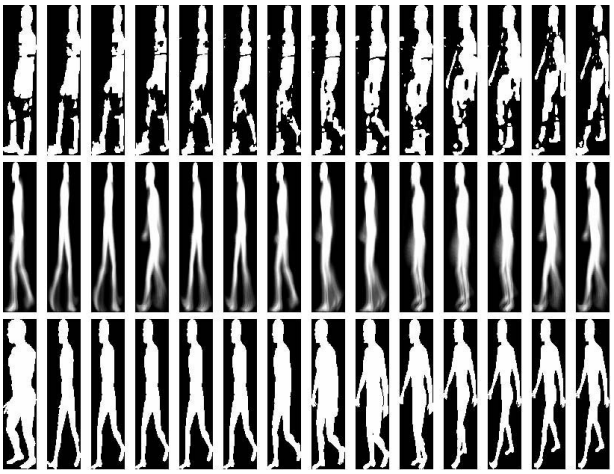

Figure 9. As in the first three rows of figure 8, but for the second camera view.

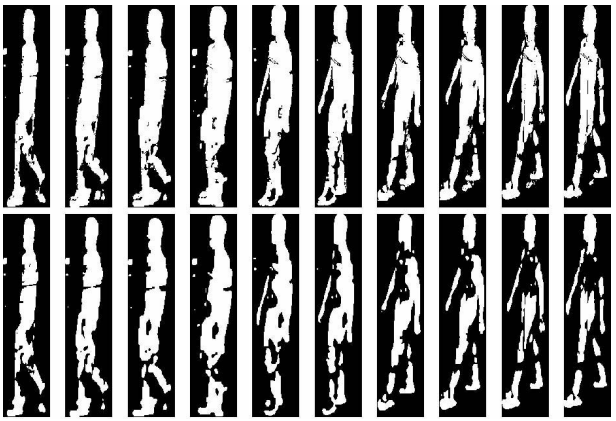

Figure 10. Bottom row: original silhouettes from background subtraction. Top row: improved silhouettes.

8. Conclusions

We have demonstrated a system which, given video input of a walking person from multiple synchronized, uncalibrated views, generates a sequence of 3D poses of a synthetic model that best match the input as well as provides estimates of the positions of the cameras. The major advantage of the system is that it requires no prior knowledge

about the locations of the cameras and no manual initialization. The drawback is that the generated 3D poses are specific to the model rather than the observed person. However, such information is nevertheless of immediate use, as it provides a way to compare walking sequences of people observed from different views. For each such sequence, our algorithm can be used to find the matching sequence of 3D model poses. The observed motion can then be evaluated against the extracted 3D model poses, thereby using our synthetic appearance model as a common reference point. The information extracted by our system can further be used as the initial estimate of the true 3D pose of the person, subject to further refinement and subsequent analysis or identification. Finally, the appearance model fitting can be used to propagate higher level information into a background subtraction algorithm and thus significantly improve the quality of silhouettes.

References

[1] J. Bilmes. A gentle tutorial of the EM algorithm and its application to parameter estimation for gaussian mixture and hidden markov models. Technical Report ICSI-TR-97-021, International Computer Science Institute, Berkeley, California, 1998.

[2] C. Bregler and J. Malik. Tracking people with twists and exponential maps. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Santa Barbara, CA, June 1998.

[3] T.-J. Cham and J. Rehg. A multiple hypothesis approach to figure tracking. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Ft. Collins, CO, June 1999.

[4] I. Cohen, G. Medioni, and H. Gu. Inference of 3D human body posture from multiple cameras for vision-based user interface. In *5th World Multi-Conference on Systemics, Cybernetics and Informatics, Orlando, Florida*, 2001.

[5] R. Duda, P. Hart, and D. Stork. *Pattern Classification*, section 10.7.3, Scatter Criteria, pages 544–546. Wiley-Interscience, New York, NY, second edition, 2001.

[6] K. Grauman, G. Shakhnarovich, and T. Darrell. Inferring 3D structure with a statistical image-based shape model. In *International Conference on Computer Vision*, Nice, France, October 2003.

[7] N. Howe, M. Leventon, and W. Freeman. Bayesian reconstruction of 3D human motion from single-camera video. In *Neural Information Processing Systems*, Denver, CO, November 1999.

[8] L. Lee, G. Dalley, and K. Tieu. Learning pedestrian models for silhouette refinement. In *International Conference on Computer Vision*, Nice, France, October 2003.

[9] J. Migdal. Robust motion segmentation using markov thresholds. Master's thesis, MIT, Dept. of Electrical Engineering and Computer Science, September 2003.

[10] I. Mikic, M. Trivedi, E. Hunter, and P. Cosman. Articulated body posture estimation from multi-camera voxel data. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Hawaii, December 2001.

[11] R. Rosales and S. Sclaroff. Inferring body pose without tracking body parts. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Hilton Head Island, SC, June 2000.

[12] C. Stauffer and W. Grimson. Adaptive background mixture models for real-time tracking. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Ft. Collins, CO, June 1999.

[13] A. Viterbi. Error bounds for convolutional codes and an asymptotically optimal decoding algorithm. *IEEE Trans. Informat. Theory*, IT-13:260–269, 1967.