

Simultaneous Pose Estimation of Multiple People using Multiple-View Cues with Hierarchical Sampling

Joel Mitchelson and Adrian Hilton
Centre for Vision, Speech, and Signal Processing
University of Surrey, Guildford, UK
j.mitchelson@surrey.ac.uk

Abstract

We present a novel method for dynamic estimation of pose of multiple people using multiple video cameras. Tracking is performed using a model-based approach and a set of cues which exploit both shape and colour information. For shape we propose a fast line-search method to incorporate multi-view constraints without the computational overhead of a voxel representation. The tracking algorithm is a new hierarchical stochastic sampling scheme. Results are presented using natural movements of up to two people sharing the same capture volume. Tracking is shown to be robust over a range of natural movements, including considerable occlusion. Processing times for tracking two people are at least as short as those for tracking one person using other stochastic schemes. The high performance and efficiency are attributed to the hierarchical search method and the accuracy of the cues in identifying suitable poses.

1 Introduction

The search is on for a fast and robust method for human pose estimation from video sequences. Such a scheme would allow the task of computer character animation from human movement (*motion capture*) to be greatly simplified, removing the need for markers or suits (see [8]). It has the potential to revolutionise human-computer interaction by allowing automatic gesture recognition, and aid in sports or biomedicine by providing a non-invasive means of analysis. Since all of these applications may involve motions of more than one person, it is important to look at issues involved in simultaneous pose estimation of multiple people.

Recent work has been aimed at improving efficiency and robustness of solutions to the problem of tracking a single person. Contributions include those from Deutscher *et al.* [1], Mikić *et al.* [10], and Sidenbladh *et al.* [13]. For a comprehensive survey of the whole area of visual motion capture up to the year 2000, see work by Moeslund [12].

This paper demonstrates the feasibility of tracking human pose during complex movements of multiple people in a multi-camera studio environment, with strong emphasis on efficiency. Section 2 describes a fast hierarchical stochastic search scheme which is a streamlined version of the approach given in previous work [11]. In section 3 we develop

a suitable structural and dynamical model of human motion. No strong motion priors [13] are used, so a large range of movements can be tracked. We develop a set of cues to use with our structural model which incorporate both shape and colour information. For shape we use a multi-view line search method, which avoids the need for a voxel representation as in [10]. Colour is modelled using a simple partitioning of HSV (hue-saturation-value) colour space [3], mapped onto the geometric model. We avoid explicit restrictions on appearance such as distinct colour for hands [15]. The scheme is evaluated using sequences of one and two people from multiple cameras.

2 Hierarchical Stochastic Sampling

2.1 Background

Suppose we have a physical articulated structure which we wish to track visually using video from one or more cameras. The *state* of the structure $\mathbf{X} \in \mathbb{R}^K$ refers to the parameters we wish to estimate. This may include global position, orientation, relative orientation of structural parts, and corresponding velocities and accelerations. We use a model-based approach to tracking, so we have a *fitness function*, $f : \mathbb{R}^K \mapsto [0, 1]$ dependent on the video input data. The aim of tracking is to update a state estimate to maintain high values of f as we receive new frames of input.

A standard method for doing this is the *particle filter* [7, 2]. This technique maintains a collection of N likely states $\{\mathbf{X}_i : i = 1 \dots N\}$, known as particles. Useful graphical notation for particle filters was developed in [9]. Using similar notation, we represent the *standard* particle filter (also known as a sampling-importance-resampling filter) as shown in figure 1. $\{\mathbf{X}\}$ represents the initial set of particles (assumed to be the output

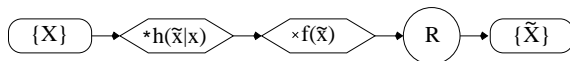


Figure 1: *Notation for a standard particle filter*

from the previous frame, or the result of manual initialisation). The first operation of the algorithm, denoted here by $*$, means application of stochastic dynamics according to some function h . Typically this means modifying each particle by a random amount in order to approximate the distribution of possible motion from one frame to the next. The next operation \times is the evaluation of some fitness function f for each new particle location. Finally R denotes resampling according to the newly evaluated fitness, leading to our updated set of estimates, $\{\tilde{\mathbf{X}}\}$. This process is repeated as frames arrive for processing at successive time increments. Unlike [7] we will use such techniques in a situation where $f(\mathbf{X})$ is not directly proportional to the probability that the image data was created using state \mathbf{X} . This means we cannot formally prove that our scheme is *asymptotically correct* [2], but instead we verify experimentally that our fitness function gives correct results.

It is known that particle filters are not efficient at tracking in high dimensional spaces. The method of Deutscher *et al* [1] is one of the most general solutions to the problem of dimensionality. A less general but highly efficient approach developed by MacCormick and Isard is *partitioned sampling* [9]. But this imposes a linear hierarchy of sampling which may not be related to the true structure. This asymmetry can cause performance

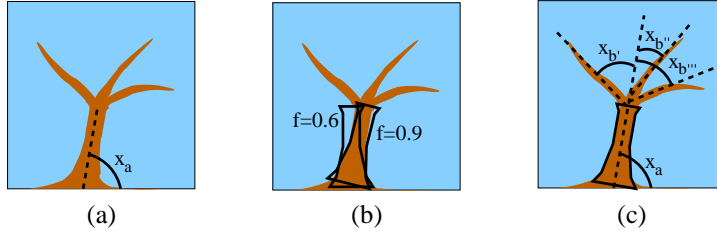


Figure 2: Tree tracking example (all images are from the same time instant)

to suffer when a small number of particles are used for a system with many degrees of freedom. We use a *parallel partitioning* of the space, so that the sampling scheme reflects the true nature of the hierarchy. When tracking using the simple human model described in [1], a pre-determined partitioning of the space is not possible, thus motivating the *soft partitioning* scheme. Contrary to this, we assume that a pre-determined partitioning *is possible* given a good enough model, thus allowing a more efficient scheme. Justification for this assumption in the case of human motion is given in section 3.

2.2 Theory

Our method assumes that we can parameterise a structure in such a way that some parameters are associated with very strong visual cues, and so can be localised independently of other parameters. A similar assumption is implicit in the standard partitioned sampling scheme. This is fundamentally different from the annealed particle filter approach of Deutscher *et al.* [1] which attempts no explicit partitioning of the space and instead uses several iterations per frame of an algorithm for sampling the *whole* space, based around the basic particle filter shown in figure 1.

To illustrate our approach, consider the ‘toy’ problem of tracking a tree, shown in figure 2. Let us say that the state of part of a structure can be represented by a vector of real numbers, \mathbf{x}_a . For the tree, \mathbf{x}_a might be the angle between the tree trunk and the ground as shown in 2(a). Now suppose that we have a fitness function $f(\mathbf{x}_a)$ which tells us how well a given state matches video data from a given time instant. Figure 2(b) shows what the output of a suitable fitness function might be for some candidate tree poses. As tracking proceeds we wish to maintain a collection of possible states, $\{\mathbf{x}_{a_i} : i = 1 \dots N\}$, one of which is close to the true state to within an acceptable margin of error. If we are able to do this without modelling any other parts of the structure, then we say that the part is *root distinct* with respect to f , and we call that part the *root part*.

We parameterise the state of our structure in layers. Layer *A* comprises just the parameters of the root part \mathbf{x}_a . Layer *B* may comprise more than one part. These are parts whose visual features allow their state to be determined once possibilities for the state of the root part are known. Each part in layer *B* will have its own state parameters $\mathbf{x}_{b'}$, $\mathbf{x}_{b''}$, $\mathbf{x}_{b'''}$, etc. In the example, figure 2(c) shows tree branches as layer *B* parts, and layer *B* parameters as the angles between each branch and the trunk. For more complex hierarchies we can go on to define further layers. We can form a complete state vector \mathbf{X} for the state of the structure by concatenating these: $\mathbf{X} = [\mathbf{x}_a \mathbf{x}_{b'} \mathbf{x}_{b''} \mathbf{x}_{b'''} \dots]$.

We now describe a modified particle filter to take advantage of such a hierarchy. Let

$\{\mathbf{X}_i : i = 1 \dots N\}$ be an initial estimate of possible poses for the whole structure. Let us split each particle into its component parameter vectors, to give sets (\mathbf{x}_{ai}) , (\mathbf{x}_{bi}) , $(\mathbf{x}_{b'i})$, etc. By construction, we can refine our estimates of (\mathbf{x}_{ai}) independently of the other parameters. This is done using the same idea as the standard particle filter (figure 1), and can be written as in figure 3.

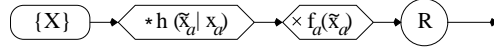


Figure 3: *Sampling scheme for layer A*

We must now refine layer *B* parameters. But here we are presented with a problem. The fitness of a set of layer *B* parameters depends on the layer *A* parameters with which they are associated. How do we know which of the resampled layer *A* parameters to pair with which initial layer *B* parameters? To most accurately preserve likelihood distributions, the 'correct' answer is to maintain the association throughout, so that when layer *A* parameters are eliminated or repeated by resampling, the corresponding layer *B* parameters are eliminated or repeated respectively. This is the approach favoured by partitioned sampling, but may cause unwanted depletion of layer *B* particles unless larger numbers of particles are used. An alternative is to use the layer *A* transitions as a dynamical model which can be used to drive layer *B* parameters. In other words, suppose the i^{th} element of our particle set has parameters \mathbf{x}_{ai} before filtering, and $\tilde{\mathbf{x}}_{ai}$ afterwards. We predict $\tilde{\mathbf{x}}_{bi}$ by sampling from a distribution of the form $h_b(\tilde{\mathbf{x}}_{bi} | \tilde{\mathbf{x}}_{ai}, \mathbf{x}_{ai}, \mathbf{x}_{bi})$. The reasoning behind this model is somewhat involved, but it will be seen that it produces good results. Fitness is evaluated for all new layer *B* parts from all particles.

We must now resample, but different parts of layer *B* may favour different layer *A* states. The next step is therefore a 'compromise' step. For each particle, we take the best fitness amongst all layer *B* parts, and use this to resample from layer *A* particles. The idea is that layer *A* states which allow layer *B* states of high fitness will be chosen. Then we resample layer *B* particles for each part, using the original fitness outputs. Finally, we re-associate layer *B* parameters with a layer *A* parameters. Where possible, each layer *B* particle is associated with its original parent layer *A* particle. Where this is not possible for a particular layer *B* particle, a random selection amongst available layer *A* particles is made. It is this compromise step which allows the parallel partitioning of state space, which is the new contribution of our sampling scheme. The complete algorithm is given in figure 4, where *C* denotes recombination of layer *A* and *B* states as described.

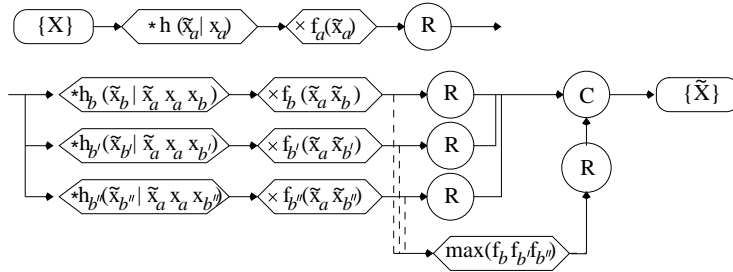


Figure 4: *Hierarchical Stochastic Sampling Algorithm*

3 Human Pose Estimation

We now apply hierarchical stochastic sampling to the problem of human pose estimation.

3.1 Model Creation

Here we define a parameterised model of human appearance. For each person to be tracked, we model limbs as rigid bodies attached to an underlying skeleton, whose pose can be represented with 22 parameters. This structure is shown schematically in figure 5(a). Truncated cones can be used for a reasonable approximation to the shape of arms and

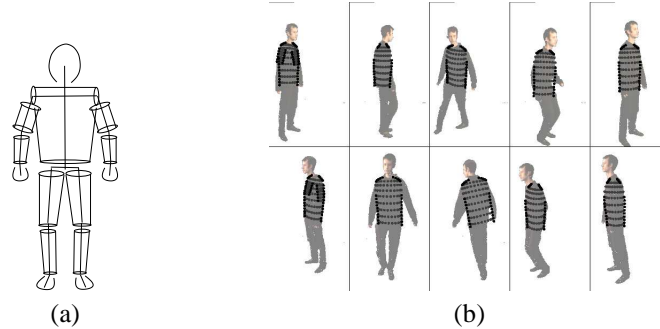


Figure 5: (a) *Geometry of Human Model*, (b) *Independent tracking of torso as the subject walks in a circle and then jumps. Two views are shown, and each column represents a different time, in increments of 25 frames (1s).*

legs. Hands, feet, and head are modelled using ellipsoids. The torso is a more complex shape, and we model it explicitly using a set of 3-dimensional surface points. A graphical user interface (GUI) and an initial set of input images are used to interactively adjust the geometry of the model, to fit each person we wish to track. This assumes known camera calibration using the standard *pinhole* model [6]. The detailed shape of the torso is computed using a volume carving method.

It is also possible to create a model for the colour of the body parts. We define a fixed grid of 3D points $(\mathbf{x}_i) \subset \mathbb{R}^3$ which cover the surface of each body part. For each point, an occlusion test is used to decide whether the point is visible in each camera view. A point is projected into each view in which it is visible, and the image colour is sampled over a small window of pixels. We take the mean colour over all window pixels from all views as a model for the colour \mathbf{c}_i of the point, but do not create a model for points whose colour variance across the pixels is too high. This prevents us from modelling regions which do not have uniform colour. Future work could use texture cues in such regions.

3.2 Cost Functions

Given an estimated pose of one part of a person, we need to evaluate the discrepancy between the model and a given set of input images, known as the *cost*. Again we assume input from multiple calibrated cameras.

Since our work is aimed at studio environments, we assume a plain background can be used for easy segmentation of people from background. This produces a set of silhouette images. Let $S_i \subset \mathbb{Z}^2$ be the set of 2D image points which lie on the silhouette from the i^{th} camera, and let $H_i(\mathbf{x}) : \mathbb{R}^3 \mapsto \mathbb{Z}^2$ be the pinhole projection function of world point \mathbf{x} into the i^{th} camera, rounded to the nearest pixel. Now let us define a three-dimensional volume V , constrained by our multiple-view silhouette measurements:

$$V = \{\mathbf{x} \in \mathbb{R}^3 : H_i(\mathbf{x}) \in S_i \forall i\} \quad (1)$$

Neglecting image noise, this is the largest volume which may be occupied by people or objects in the scene, and forms the basis for all measures of shape from silhouettes. This volume may be modelled explicitly by sampling over a voxel grid, as in [10], so that cost measures can be made using the shape of the voxel set. We prefer to use V implicitly by making measurements in multi-view image space.

If a body part fits the data perfectly, the volume occupied by the model must lie inside V . Equivalently, the projection of the model into the images must lie inside each silhouette. This gives us a first measure of cost. Let us approximate the surface of the model using a grid of 3D points, (\mathbf{x}_j) . Then we define the *silhouette overlap cost*, C_o as:

$$C_o = |\{j : H_i(\mathbf{x}_j) \in S_i \forall i\}| \quad (2)$$

Where $|\cdot|$ is the number of elements in the set. In practice, to reduce noise, we disregard points near the occluding boundary of the model for a given view. Minimising C_o is not sufficient to distinguish pose, since it only requires the model to be *somewhere* inside V . In the absence of occlusion from other objects, a perfect fit also implies that when the body part is projected into camera views, its occluding boundary co-incides with the occluding surface of V . This concept can be used to define another cost function. First note that the surface of V is completely defined by:

$$\sigma = \{\mathbf{x} \in \mathbb{R}^3 : H_i(\mathbf{x}) \in S_i \text{ for all } i \text{ and } H_i(\mathbf{x}) \in \hat{S}_i \text{ for some } i\} \quad (3)$$

Where \hat{S}_i is the boundary of silhouette pixels S_i . Let us project the body part into each camera view, and determine a set of points $(\mathbf{x}_{ik}) \subset \mathbb{R}^3$ and corresponding surface normals $(\mathbf{n}_{ik}) \subset \mathbb{R}^3$ at regular intervals on the occluding contour. We estimate the shortest distance from \mathbf{x}_{ik} to σ by searching along the line $\mathbf{x}_{ik} + t\mathbf{n}_{ik}$, $t \in \mathbb{R}$. The greater this distance, the further our model is from the data. We define the *edge proximity cost* function, C_e :

$$C_e = \sum_i \sum_k \min\{|t| : (\mathbf{x}_{ik} + t\mathbf{n}_{ik}) \in \sigma\} \quad (4)$$

It is possible to implement this search very efficiently by using an extension of the standard Bresenham line drawing algorithm [4] to multiple views.

The third cue we use is colour. For this we use the colour model $(\mathbf{x}_l, \mathbf{c}_l)$ for our body part as defined in section 3.1, where \mathbf{x}_l is the 3D location of each surface point according to the given pose. For the i^{th} view, let $[h_{il} \ s_{il} \ v_{il}]$ be the HSV colour [3] of the image pixel at location $H_i(\mathbf{x}_l)$. Let $[\bar{h}_l \ \bar{s}_l \ \bar{v}_l] = \mathbf{c}_l$. We discard points whose surface normals face away from the camera. We use a simple partitioning of the HSV colour space to determine discrepancy between model colour and actual colour. For this, we define four thresholds: κ_{black} , κ_{colour} , κ_{hue} , $\kappa_{\text{saturation}}$. These must be set manually, but

suitable defaults can be chosen which work in most situations. Model pixel and image pixel are said to match under any of the following conditions:

1. $v_{il} < \kappa_{\text{black}}$ and $\bar{v}_l < \kappa_{\text{black}}$
2. $v_{il} > \kappa_{\text{black}}$ and $\bar{v}_l > \kappa_{\text{black}}$ and $s_{il} > \kappa_{\text{colour}}$ and $\bar{s}_l > \kappa_{\text{colour}}$
3. $v_{il} > \kappa_{\text{black}}$ and $\bar{v}_l > \kappa_{\text{black}}$ and $s_{il} < \kappa_{\text{colour}}$ and $\bar{s}_l < \kappa_{\text{colour}}$ and $|h_{il} - \bar{h}_l| < \kappa_{\text{hue}}$ and $|s_{il} - \bar{s}_l| < \kappa_{\text{saturation}}$

The colour cost function C_c is equal to the proportion of sampled pixels for which these conditions do not hold. Note that this model exploits colour cues only where they are available - it has no specific requirements such as distinct hand colour [15]. Our separate cost functions can be combined into an overall cost:

$$C = \lambda_o C_o + \lambda_e C_e + \lambda_c C_c \quad (5)$$

The weighting factors λ_o , λ_e and λ_c are currently fixed values which are set manually. Future work may look at automated ways to choose these based on the input data.

3.3 Application of Hierarchical Sampling

To apply our hierarchical sampling method for tracking of human pose we must define: the fitness function for body parts; the parameterisation and its hierarchical partitioning; and a stochastic dynamical model for human movement.

The purpose of evaluating fitness is to sort poses which correspond well to images from those that do not. The conversion from cost value to fitness value is key to a successful tracker. A good candidate for this was found to be the positive part of a Gaussian function:

$$f = e^{-(\beta C)^2} \quad (6)$$

β is a constant set manually for each body part depending on the expected range of motion.

We found that over a wide range of movements, the torso as modelled here is root distinct. Figure 5(b), for example, shows the result of a single-layer search for torso pose as the subject walks in a circle and then jumps. It was verified that the estimated pose remained correct throughout. So we choose the torso as the root part, and the 6 degrees of freedom of the torso as layer A parameters. In this work we use just two layers, so layer B comprises 4 parts: left arm, right arm, left leg, right leg.

For the dynamics of torso translation we use a 3D Gaussian distribution. For torso rotation, we use an *exponential map* [5] of a 3D Gaussian distribution. The exponential map, or *axis-angle* is a map from \mathbb{R}^3 to $SO(3)$, whereby the direction of a 3D vector is taken as the axis of rotation, and its magnitude is taken as the rotation angle. It was found to help if torso rotation is with respect to its geometric centre. For arms and legs, we assume constant velocity of feet and hands with respect to the torso, plus a 3D Gaussian distribution. Inverse kinematics [14] is then used to retrieve elbow and knee positions. The angle of elbow and knee axes is a parameter of the inverse kinematics, and this too is modified according to a 1D Gaussian variate. It was found that over a variety of motions inverse kinematics allowed more effective modelling of dynamics than Euler angles as used in [1]. As with other stochastic schemes [1], the variance of all the Gaussian distributions of dynamics is set manually.

4 Results

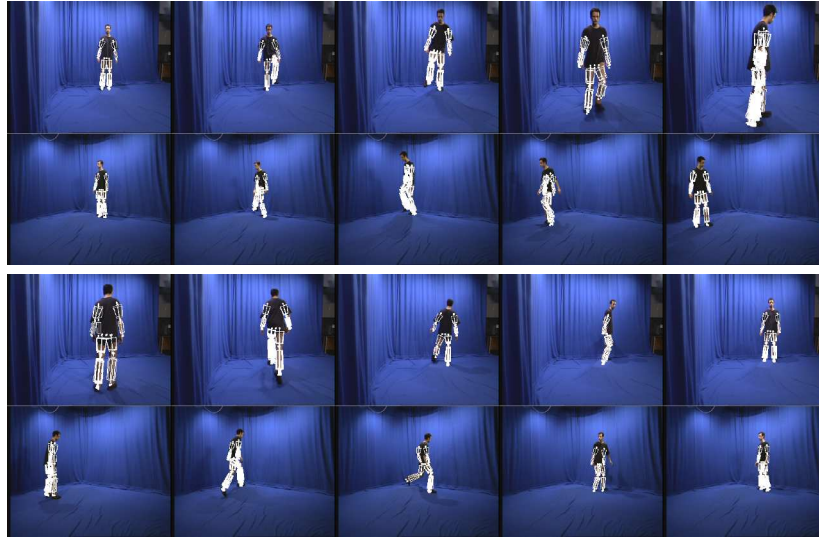


Figure 6: Tracking results overlaid on two views of WALK sequence, Shown at 20 frame (0.8s) increments

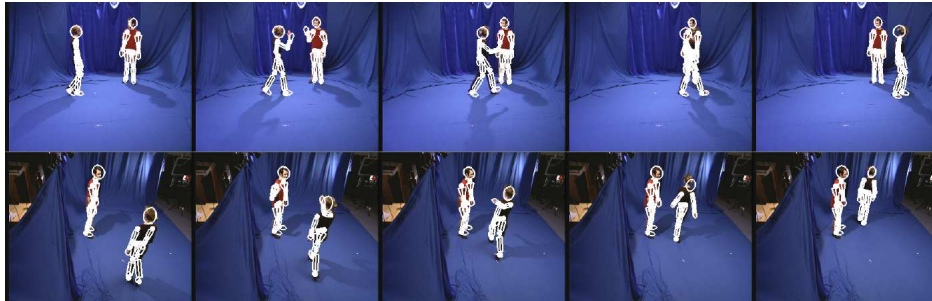


Figure 7: Tracking results overlaid on two views of WAVE sequence. Shown at 12 frame (0.48s) increments.

Data was acquired in a studio with blue-screen background and multiple calibrated cameras. Three sequences are presented here: WALK, WAVE and BUMP. For WALK and WAVE we use a tracker with 500 particles, for BUMP we use 1000. Maximum processing time was 15s per frame (1.7GHz Xeon, 5 cameras, 2 people, 1000 particles). In figure 6 the set of output particles from our tracker are overlaid on the WALK images. We see that the pose is successfully tracked throughout. In WAVE (figure 7) we see some inaccuracies due to occlusion as the people move close to each other, the tracker quickly recovers. Only the mean output is shown. In BUMP (figure 8) the characters brush very close to each other, and track of one arm of each person is lost, since they cannot be resolved from any view. Once again, we see that the tracker recovers after this occlusion.



Figure 8: Tracking results overlaid on two views of BUMP sequence. Shown at 10 frame (0.4s) increments.

If any of our set of cues were disabled, tracking of WAVE and BUMP was not possible using a reasonable number of particles.

5 Conclusions and Future Work

We have shown a tracking scheme capable of tracking multiple people. The ability to *maintain* track during complex movements compares well with existing methods for single person tracking. Very few constraints are put on the motions which can be tracked or the appearance of the people. Our processing time of 15s on a 1.7GHz Intel Xeon processor for 2 people using 5 cameras compares well with the 15s / 1GHz for one person using 3 cameras quoted in [1]. (Using 3 cameras and one person our processing times are around 3s). We attribute the efficiency to the cues we use, and the ability of the algorithm to exploit them. Use of hierarchical stochastic sampling reduces the computational complexity required to track the given articulated model, compared to alternatives such as annealed particle filtering [1] and partitioned sampling [9].

Due to the stochastic method used, our tracking output is not yet as smooth as required for realistic animation. This may be improved in future by intelligent filtering of output, and possible use of more cues such as texture. We also hope to maintain high accuracy even during occlusion. This may be achieved by explicit modelling of the occlusion, and a more detailed dynamical model. It may be possible to make the choice of parameters for dynamics automatic and adaptive to the data. The use of a plain background is a limitation of the system, but for applications such as TV production work, plain blue-screen environments are readily available to end-users.

Finally, we aim to produce a scheme which runs in real-time. There is room for more optimisation in the code, but we may require further refinement of the sampling scheme, and the ability to selectively evaluate only the most useful cues. Nevertheless, by demonstrating that multiple people can be tracked in reasonable time periods, we can see that real-time pose estimation of multiple interacting people remains feasible for the future.

References

- [1] J. Deutscher, A. Davison, and I. Reid. Automatic partitioning of high dimensional search spaces associated with articulated body motion capture. In *Proc. Conference of Computer Vision and Pattern Recognition*, 2001.
- [2] A. Doucet, J. de Freitas, and N. Gordon. *Sequential Monte-Carlo Methods in Practice*. Springer-Verlag, 2000.
- [3] J.D. Foley, A. van Dam, S.K. Feiner, and J.F. Hughes. *Computer Graphics: Principles and Practice in C (2nd Edition)*. Addison-Wesley, 1995.
- [4] A. Glassner, editor. *Computer Graphics Gems*. Academic Press, Inc., 1990.
- [5] F.S. Grassia. Practical parameterization of rotations using the exponential map. *Journal of Graphics Tools*, 3(3):29–48, 1998.
- [6] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.
- [7] M.A. Isard and A. Blake. Visual tracking by stochastic propagation of conditional density. In *Proc. 4th European Conference of Computer Vision*, pages 343–356, April 1996.
- [8] Charnwood Dynamics Ltd. The CodaMotion system. <http://www.codamotion.com>.
- [9] J. MacCormick and M. Isard. Partitioned sampling, articulated objects, and interface quality hand tracking. In *Proc. 6th European Conf. Computer Vision - Part II*, pages 3–19, 2000.
- [10] I. Mikić, M. Trivedi, E. Hunter, and P. Cosman. Articulated body posture estimation from multi-camera voxel data. In *Proc. CVPR2001*, 2001.
- [11] J.R. Mitchelson and A. Hilton. From visual tracking to animation using hierarchical sampling. In *Proc. Mirage 2003*, INRIA Roquencourt, France, March 2003.
- [12] T. Moeslund and E. Granum. A survey of computer vision based human motion capture. *Computer Vision and Image Understanding*, 81(3), 2001.
- [13] H. Sidenbladh, M.J. Black, and L. Sigal. Implicit probabilistic models of human motion for synthesis and tracking. In *Proc. European Conf. Computer Vision*, volume 1, pages 784–800, 2002.
- [14] C. Welman. Inverse kinematics and geometric constraints for articulated figure manipulation. Master's thesis, Simon Fraser University, 1993.
- [15] C.R. Wren, A. Azarbayehani, T. Darrell, and A.P. Pentland. Pfinder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7), 1997.

Acknowledgements

This work is part of the PROMETHEUS project, supported by the UK DTI and EPSRC under the Link Broadcast Technology Programme. Special thanks to Jon Starck and Eng-Jon Ong at University of Surrey for many productive discussions, and to Graham Thomas and Marc Price at BBC R&D for their willing and helpful advice on studio techniques.