Real-Time Markerless Human Body Tracking with Multi-View 3-D Voxel Reconstruction

Fabrice Caillette & Toby Howard
Advanced Interfaces Group, Department of Computer Science
University of Manchester, Manchester M13 9PL, UK
fabrice@cs.man.ac.uk

Abstract

We present an approach to full human-body tracking, using markerless multi-view images as input, performing acquisition, reconstruction and tracking in real-time on a single PC. Our approach employs a hierarchical visual-hull algorithm which segments only the most interesting regions of the images and includes colour information. The tracking step uses blobs attached to a kinematic model to recover joint angles in an expectation-maximization framework. We demonstrate the robustness of the approach on video sequences of various body configurations in an unaugmented office environment. We also show that tracking challenging poses with self-occlusions is possible without the processing cost of stochastic sampling schemes.

1 Introduction

The main challenge when recovering body poses from markerless images is the under-constrained nature of the problem. Not only does a typical kinematic model have at least 25 degrees of freedom, but camera images provide very poor evidence of limb positions, because of frequent problems such as self-occlusions, background segmentation errors, and loose clothing.

The work presented in this paper addresses the problem of recovering posture parameters from video images of a scene as observed by multiple cameras. The goal of our system is to operate in an unaugmented environment, in real-time on a single PC with cheap webcams. These requirements immediately give rise to a number of challenges such as robustness to noise and to a cluttered environment, camera distortions and limited processing power. The key goal of real-time performance (≥ 15 fps) on commodity hardware drives most of our design decisions.

Instead of performing the tracking step on each input frame, our system first reconstructs a 3-D voxel-based representation of the person, and then performs the matching of a kinematic model directly in this 3-D space. In addition to bringing more consistency, this approach simplifies the tracking step, especially in the case of ambiguous poses and self-occlusions. Novel aspects of our approach include a statistical reconstruction method which incorporates colour information, and the use of blobs for robust real-time tracking.

We begin by summarising some previous approaches to recovering human body parameters from video images. Section 3 focuses on our 3-D reconstruction method, and the tracking process is described in Section 4. We present results and performance analysis in Section 5, and conclude by discussing future research directions.

BMVC 2004 doi:10.5244/C.18.62

2 Previous Work

Several kinds of visual cues have been used to track human body movements from video images. Gavrilla and Davis [8] project a kinematic model associated with superquadrics on the image planes and perform a best-fit search using contours as a similarity measure. Delamarre and Faugeras [6] obtain good results using multi-view silhouettes to drive a kinematic model with virtual springs, while Wren *et al.* [15] introduce a statistical description of the whole scene making use of colour, and achieving convincing results, although their system works only in 2-D. Other image-based attempts range from the use of appearance models [5] to probabilistic filtering [11]. A common trend has been to use 3-D models, which has led to more robust results. For example, Bregler [2] presents a novel description of human models with twists and exponential maps to solve the underconstrained problem of tracking.

Recently there has been interest in multi-view systems which use a prior 3-D reconstruction as a basis for tracking. The most common technique is the shape-from-silhouette or visual-hull technique [12]. Using an efficient sparse occupancy reconstruction technique, Cheung *et al.* [4] fit ellipsoids in real-time (using a cluster of PCs) on the reconstructed voxels. Luck *et al.* [9] and Mikic *et al.* [10] extend this approach by incorporating a kinematic model, at the cost of losing the real-time aspect. Theobalt *et al.* [13] use a combination of voxel-based reconstruction and 2-D feature tracking to drive a bi-layered model. Borovikov and Davis [1] adopt a full hierarchical approach by using a formal density description to fit a model on the 3-D data, where the visual-hull is built efficiently as an octree. Other relevant 3-D methods include the work on temporal visual-hull alignment and refinement by Cheung [3].

The work described in this paper extends some of the ideas from the Pfinder System [15] into 3-D space, with the addition of a kinematic model replacing the Kalman filter for blobs dynamics. Figure 1 shows a general overview of the system.

3 Statistical 3-D Reconstruction

3.1 Overview and Design Choices

The visual-hull of an object is defined as the maximal portion of space which, projected into the camera image planes, lies totally inside all the silhouettes of the object. This definition holds for an infinite number of views, but since current hardware can support only a few cameras, an approximation will be used instead (Figure 1a). The standard algorithm for computing a visual-hull is to start from voxels in 3-D space, project them into the available image planes and keep only those which lie inside all the silhouettes of the object [12]. Our approach is very similar to this, except that it does not use full binary silhouettes but per-sample statistical background segmentation within a coarse to fine recursive algorithm (Section 3.3).

Using the visual-hull as a basis for tracking is mainly justified by the consistency of the approach: the tracking algorithms will then be working directly in 3-D space, independent of the camera parameters. This visual-hull extraction acts as a filter where only the valuable information is kept (volume and colour) whereas noise is dismissed due to its non-coherence across different views. The main drawbacks of this approach are

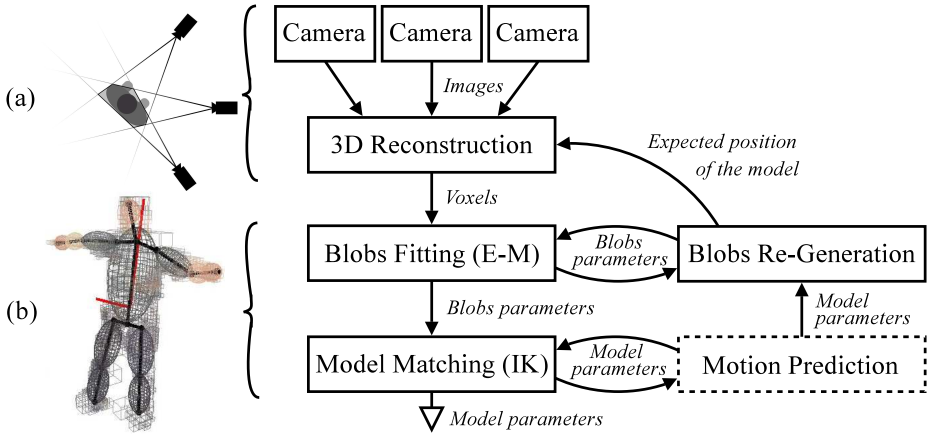

Figure 1: Overview of the system with (a) the prior 3-D reconstruction and (b) the model based tracking (The motion prediction module is the subject of future work).

processing costs and storage requirements, but we shall describe methods for minimizing both, in order to attain real-time performance.

The cameras are calibrated using Bouguet's Matlab toolbox, which is based on Zhang's method [16]. Synchronization is either done in hardware via the FireWire bus when all the cameras are plugged on the same PC, or over a network using a frame-server approach. Both types of input are supported by our system.

3.2 Voxel Projection and Background Segmentation

The projection of a voxel onto an image plane is an area that should theoretically be exhaustively inspected. However, for efficiency, we sample this area using pre-computed patterns of 8 points (Figure 2e). For each sample, a measure of likelihood that the corresponding pixel belongs to either the foreground or the background is computed. Statistics over the sampling are then used to classify the whole voxel (Section 3.3). As in most other systems [15, 4, 10], we assume that the background is static during the tracking. For each pixel, the background model M comprises the mean value μ_M and the covariance matrix K_M of the corresponding $[YUV]$ pixel components over around 50 frames captured beforehand to the tracking. The Mahalanobis distance is then computed for a pixel P as follows:

$$dst(P,M) = (P-\mu_M)\cdot K_M^{-1}\cdot (P-\mu_M)^T \tag{1}$$

This formulation benefits from hardware acceleration, since up to 4 matrix-vector multiplications may be performed in parallel with SIMD instructions like SSE, leading to an effective speedup factor of 2 to 3 times.

Shadows are not specifically handled in this representation, but our tests showed that relaxing the variance on the luminance component Y of the covariance matrix allowed most of the illumination artifacts to be removed without seriously affecting the quality of the segmentation. Note that no binary decision is taken at this stage about the classification of individual pixel samples as belonging to foreground or background. This decision

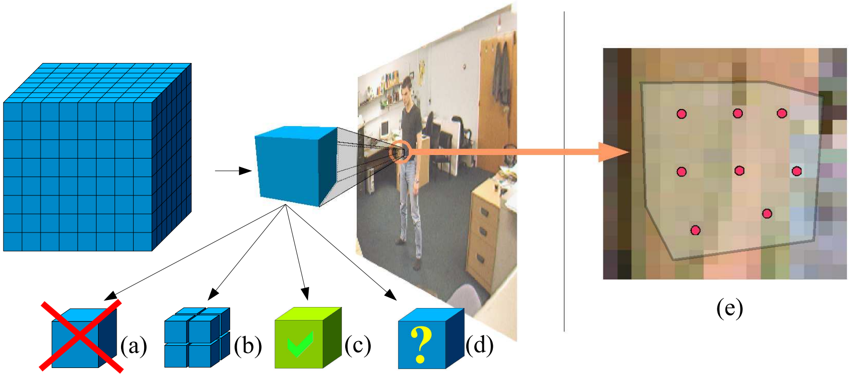

Figure 2: Projection of a voxel on an image plane, sampling of the projected area (e) and subsequent classification as background (a), edge (b), foreground (c) or unknown (d).

is only taken for the whole voxel, considering the statistics of the samplings in all the views.

3.3 Recursive 3-D Reconstruction

An important architectural choice is to compute the visual-hull volume hierarchically. The Octree representation [12, 1] was rejected because of its high memory usage. Instead, in our scheme each voxel is created dynamically during the recursive reconstruction process. An arbitrary level of accuracy can then be reached in the regions where it is needed, without the memory footprint of a rigid data structure.

Initially, a coarse $16 \times 16 \times 16$ grid of voxels is projected onto the image plane of each camera. For each voxel projection, the mean μ and the variance S of the distances (Equation 1) to the background model over the sampling are computed. Using a low and a high threshold (T_μ^L, T_μ^H) on the mean distance of the samples, as well as an adaptive threshold T_S on their variance, the classification of a voxel now takes place as follows:

- **if**$(\mu < T_\mu^L)$ the voxel is immediately classified as background (Figure 2a) and then discarded without reference to any other camera views.
- **if**$(S > T_S)$ the current voxel is temporarily marked as belonging to an edge (because of its high variance), but we still project it on the other views in case it can be seen as background from another viewpoint. If after projection on all the views, the voxel has not been discarded, it is subdivided into 8 sub-voxels (Figure 2b), and the classification process begins again recursively for each sub-voxel.
- **if**$(\mu > T_\mu^H)$ then in the current view, we may be reasonably confident that this voxel is foreground. Of course it still needs to be projected on the other views to make sure it is not eventually part of the background or an edge. If it is foreground from all the views, it is classified as such and retained as part of the reconstruction (Figure 2c).
- **Otherwise**, it means that the variance is quite low and the mean is somewhere between the two thresholds of certainty. Because we cannot infer anything definite

> about the current voxel, it is therefore temporarily classified as unknown (Figure 2d) and the other views are inspected to disambiguate the situation. If the voxel is still marked as unknown after all the views have been consulted, we look at the classification of the neighbour voxels, and give it the classification of the majority.

The recursive nature of the process leads to flexibility. For example, there is no need to maintain a static data-structure: each of the voxels gets its position from its parent, and is only created physically when it has been classified as foreground. The base position and the size of the zone of interest are then modified dynamically at no extra cost using the predicted model position from the previous frame.

One of the novelties of the reconstruction is that all foreground voxels include colour information. The mean colours of all samplings as seen by the available camera views are kept in each voxel. We do not perform any kind of visibility test at this stage to determine the real colour of the voxel since the colour that fits best the model will automatically be used, as described in the next section.

4 Model-Based Tracking

4.1 Blobs Matching with EM

Having extracted the visual-hull, we proceed by identifying the individual parts of the body, and following their movement from one frame to the next. This is a complex problem which requires knowledge about both the appearance and the dynamics of the objects being tracked. In our system, the appearance is modeled by entities called *blobs*, which track sets of voxels sharing coherent attributes. The attributes of a given voxel v are its position P_v and colours from every view $\{C_{v,i}, i \in [0..N-1]\}$ with N being the number of views. A given blob B is a statistical entity defined by a mean position and colour (μ, μ') and the corresponding covariance matrices (K, K'). The process used to match blobs to groups of voxels is a variant of the well-known Expectation-Maximization (EM) algorithm [7] which proceeds in 2 steps:

Step 1: Expectation (Figure 3a)

For each voxel v compute the distance to every blob B_j using the Mahalanobis distance dst, which is fundamentally the same as in Equation 1. Then, assign the voxel to the nearest blob:

$$dst(v,B_j) = (P_v - \mu_j) \cdot K^{-1} \cdot (P_v - \mu_j)^T + \min_{i \in [0..N-1]} (C_{v,i} - \mu'_j) \cdot K'^{-1} \cdot (C_{v,i} - \mu'_j) \tag{2}$$

$$B_{\text{argmin}_j(dst(v,B_j))} \leftarrow v \tag{3}$$

Note that the colour minimizing the distance to the nearest blob is used, reducing the effect of occlusions, without the cost of a visibility test.

Step 2: Maximization (Figure 3b)

Estimate the new means and covariances $(\hat{\mu}_j, \hat{\mu}'_j, \hat{K}_j, \hat{K}'_j)$ of each blob B_j using the set of voxels assigned to it. We shall note s_v the size of a voxel v, C_v its colour minimizing the

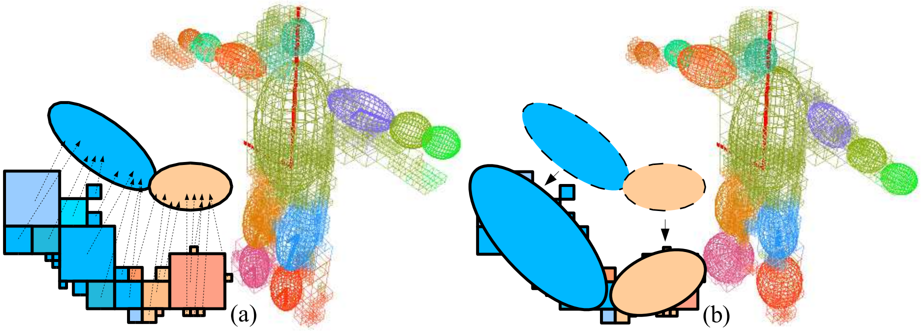

Figure 3: (a) Expectation: the voxels are assigned to the nearest blob. (b) Maximization: The blobs parameters are re-estimated.

distance to the blob B_j in Equation 2, and finally $k_v = \frac{s_v^5}{12}.I_3$ its internal covariance matrix.

$$
\begin{array}{lll}
\hat{\mu}_i = E[P_v] = \frac{\Sigma_{v\in B_i} {s_v}^3.P_v}{\Sigma_{v\in B_i} {s_v}^3} & & \hat{\mu}'_i = E[C_v] = \frac{\Sigma_{v\in B_i} {s_v}^3.C_v}{\Sigma_{v\in B_i} {s_v}^3} \\
\hat{K}_i = E[(P_v - \hat{\mu}_i)(P_v - \hat{\mu}_i)^T] & & \hat{K}'_i = E[(C_v - \hat{\mu}'_i)(C_v - \hat{\mu}'_i)^T] \\
\quad = \frac{\Sigma_{v\in B_i} k_v + {s_v}^3.P_v \cdot P_v^T}{\Sigma_{v\in B_i} {s_v}^3} - \hat{\mu}_i \cdot \hat{\mu}_i^T & & \quad = \frac{\Sigma_{v\in B_i} {s_v}^3.C_v \cdot C_v^T}{\Sigma_{v\in B_i} {s_v}^3} - \hat{\mu}'_i \cdot (\hat{\mu}'_i)^T
\end{array}
\tag{4}
$$

Constraints are placed on the maximization step to ensure that the appearance of the blob does not change too radically from one frame to the next. The use of *a priori* knowledge for colour mean and variance, and position variance (shape) gives strong hints about the kind of attributes a blob is supposed to track. For example, a blob supposed to track a hand will initially have a very strong prior of skin colour as well as a determined shape, whereas a blob created dynamically to track a forearm will have poor priors.

The formulation of the distance used in the expectation step (Equation 2) assumes that distances on position and colour are coherent. This is usually the case since the covariance matrices act as scaling factors. It can however happen that the voxels assigned to a blob are too similar in colour, leading to a predominance of the colour information over the position. To prevent this effect, we rescale $\hat{K}'_i$ when its determinant is too small.

The two steps of the EM algorithms are iterated until the error becomes sufficiently small. A simple but efficient measure of the error is the level of filling of the blobs: all the blobs should be assigned a volume of voxels in relation to their size. In practice, a single iteration is usually sufficient.

4.2 Model Pose Estimation

We use a kinematic model to constrain the blobs' movements and to recover the body parameters. Our model is a simple hierarchy of joints and bones, with constraints placed on the rotation angles of each joint. Each blob is attached to an arbitrary position along a bone segment. The positions and orientations of the blobs after the maximization step are used to drive the model with an inverse kinematics algorithm.

The first step is to compute the goal position of each joint from the blobs attached to the adjacent bones, regardless of the model constraints. It is an optimization process that uses both the attributes of the blobs and the predicted placement of the bone segments from the previous frame. The main axis of a blob gives a very valuable estimation of the orientation of the corresponding bone segment, especially for oblong body parts such as the limbs. Extracting the main eigenvector from a covariance matrix is usually done via a principal component analysis, but we found this approach inappropriate for real-time purposes. We perform instead a local iterative maximization of the main eigenvalue around the predicted eigenvector, which proved very fast and robust in practice. For rounder body parts like the head or the hands, no orientation is computed.

In a second step, we use the goal positions of the joints to optimize the model parameters with inverse kinematics. Our implementation is based on the Coordinate Cyclic Descent method [14] which sequentially performs a local optimization of joint angles from the end-effectors (hands, feet and head) down to the root of the kinematic tree. This local optimization consists in minimizing the distances between the following joints in the kinematic tree and their goal position by acting only on the current joint's angle. An attenuation factor is used to smooth the movements and a few iterations of the process are then necessary. We found this 2-step approach very robust, especially in the case of missing data where the inverse kinematics algorithm accommodates easily with the absence of some goals.

Finding reliably the position and orientation of the root of the tree (pelvis) is crucial for the above inverse kinematics scheme, and must be done beforehand. We determine those as the parameters minimizing the Euclidean distances between the goals and the actual joints for the hips, the torso and the shoulders. The new root's rotation is constrained by the previous orientation to prevent sudden back/front flippings. We also found the root-position to be more stable by constraining at least one part of the body to touch the floor. For initialization, the subject is asked to adopt a starfish position during which the model is positioned using the eigenvectors of the voxels' distribution. A few frames are typically sufficient for the blobs to acquire their colour and shape models before starting tracking.

4.3 Re-Generation of the Blobs

The kinematic model is not only used to recover the body parameters, but to correct the blobs' attributes as well. Without model-based adjustments, the blobs would degenerate, especially in the presence of noise or occlusions. Blobs are re-created at each frame before the expectation step, using their attached position from the kinematic model and their estimated attributes from the previous maximization. Depending on the strength of the priors in shape and colour, the covariance matrices are either kept or blended with the prior's.

One of the most challenging aspects of body tracking is to deal with occlusions. A blob is considered as occluded when its level of filling after the EM steps falls below a given percentage of its prior volume. When this happens, the blob is simply de-activated in the sense that it does not take part in the Maximization step anymore, and it is not used to drive the kinematic model. However, the blob's position is still updated by the movements of the underlying kinematic model, and it still receives voxels in the Expectation step. Thus, if and when its level of filling returns to a significant level, it is re-activated.

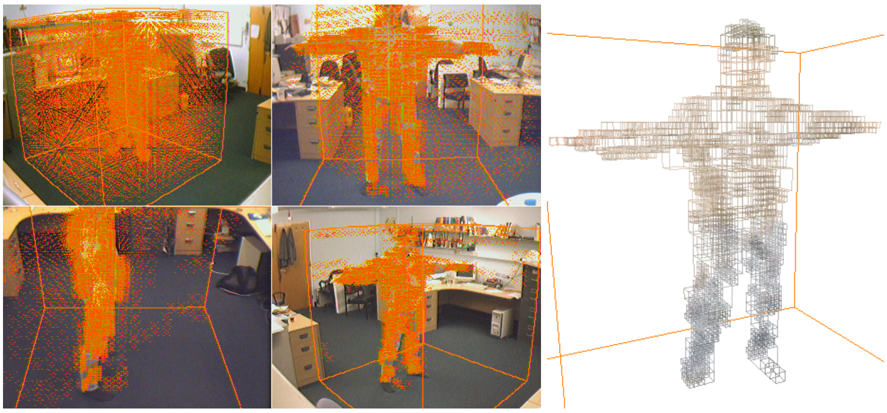

Figure 4: 3-D reconstruction with (left) dots representing the pixel samples used for reconstruction on the input camera-views and (right) the resulting visual-hull.

5 Results

The configuration used for the tests is a single 2 GHz PC with a FireWire interface and four web-cams capable of 320 × 240 at 30 Hz or 640 × 480 at 15 Hz. The resolution of the input images does not affect the speed of the reconstruction as the segmentation is done per voxel. The voxel-space has a maximal resolution of 128 × 128 × 128, which corresponds to a precision of 1.5 centimetres.

The 3-D reconstruction process is demonstrated in Figure 4. We can notice that less and less samples (red dots on the input images) are necessary on consecutive camera views, concentrating only on the foreground regions. The consequence of this is that additional camera views can be added at a negligible extra-cost. The reconstruction ran on average at 62 fps with 2 cameras, 57 fps with 3 cameras, and 61 fps with 4 cameras. Our explanation for these results is that the voxels quickly discarded by adding camera views roughly compensate the cost of the extra samples.

Tracking results are presented in Figure 5. We found the model-based pose estimation very robust to noise and self occlusions that are not always solved by the visual-hull algorithm. The processing costs of EM blob fitting and model matching are relatively small, thanks to the hierarchical voxel reconstruction, allowing the image capture, reconstruction and tracking to run at 15 fps. Figure 6 highlights the advantage of using colour information for cases where the standard visual-hull is insufficient.

6 Conclusion and Future Work

We have presented a system performing robust and accurate full human-body tracking in real-time on commodity hardware. Novelties include per-sample statistical 3-D reconstruction and the use of coloured voxels for tracking. The presented results confirm the validity of the approach for real-time applications, despite the additional reconstruction step. Future work will include motion prediction and filtering to limit the small jitters currently affecting the model's movements. We also intend to improve the model's constraints, possibly using learned Bayesian priors.

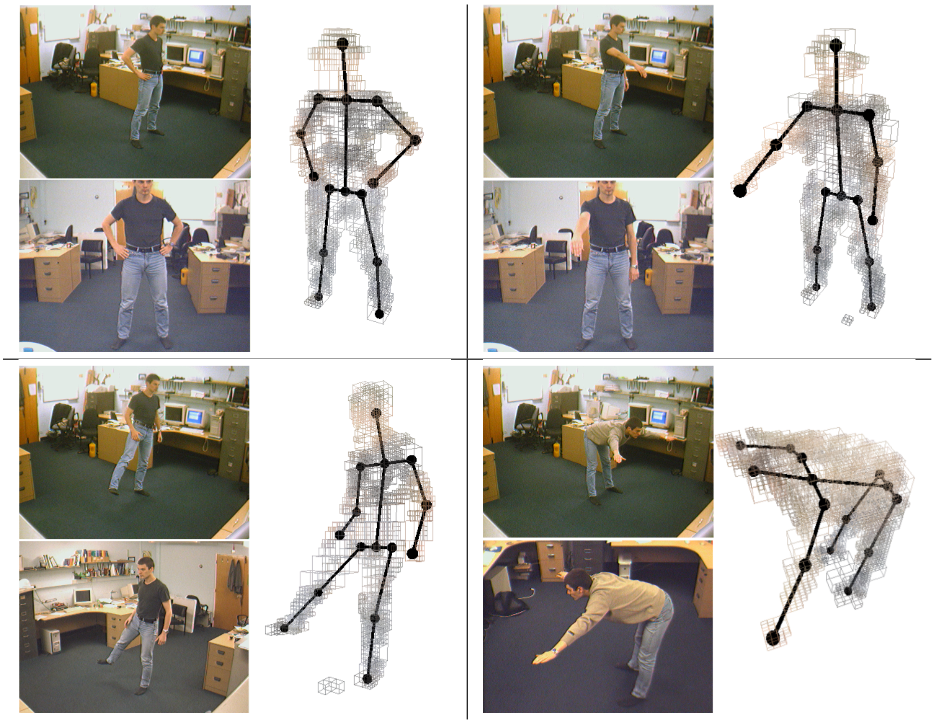

Figure 5: Examples of tracking for various body configurations. The reconstruction is performed from 4 camera views of which two are shown on the left. The displayed voxels' colours are the mean of the colours in all views. It can be noticed that shadows, cluttered environment and the self-occlusions lead to segmentation and reconstruction errors, which are mostly dismissed by the model matching.

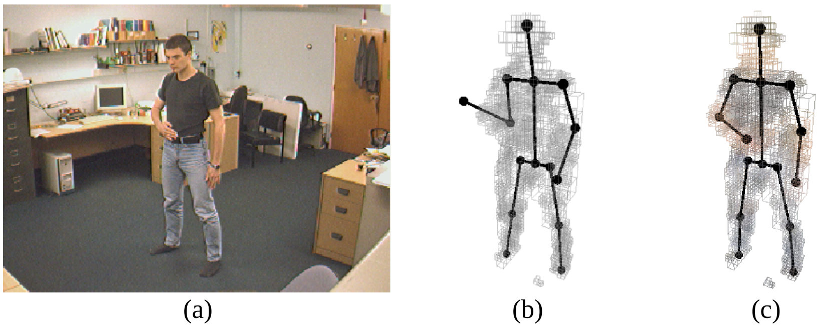

Figure 6: In cases where limbs are close to the body (a), or more generally when self-occlusions occur, tracking using only the voxel volume gives poor results. However, using colour information (c) increases the likelihood of correct pose estimation.

Acknowledgments

It is a pleasure to acknowledge the helpful contributions of our colleagues in the Advanced Interface Group.

References

[1] E. Borovikov and L. Davis. 3D shape estimation on density driven model fitting. In *3DPVT'02*, page 116, 2002.

[2] C. Bregler. Tracking people with twists and exponential maps. In *CVPR*, 1998.

[3] K. M. Cheung. *Visual Hull Construction, Alignment and Refinment for Human Kinematic Modeling, Motion Tracking and Rendering*. PhD thesis, CMU, 2003.

[4] K. M. Cheung, T. Kanade, J.-Y. Bouguet, and M. Holler. A real time system for robust 3D voxel reconstruction of human motions. In *CVPR00, V2, P714–720*.

[5] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. In *ECCV*, volume 2, pages 484–498. H. Burkhardt & B. Neumann, 1998.

[6] Q. Delamarre and O. D. Faugeras. 3D articulated models and multi-view tracking with silhouettes. In *ICCV*, pages 716–721, 1999.

[7] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society Series B*, 39(1):1–38, November 1977.

[8] D. M. Gavrila and L. S. Davis. Towards 3-D model-based tracking and recognition of human movement: a multi-view approach. In *FG'95*, 1995.

[9] J. P. Luck, D. E. Small, and C. Q. Little. Real-time tracking of articulated human models using a 3D shape-from-silhouette method. In *CVPRIP'2002*, 2002.

[10] I. Mikic, M. Trivedi, E. Hunter, and P. Cosman. Human body model acquisition and tracking using voxel data. *IJCV*, 53(3), July/August 2003.

[11] J. Mitchelson and A. Hilton. Simultaneous pose estimation of multiple people using multiple-view cues with hierarchical sampling. In *BMVC*, September 2003.

[12] R. Szeliski. Real-time octree generation from rotating objects. Technical Report 90/12, Digital Equipment Corporation, Cambridge Research Lab, December 1990.

[13] C. Theobalt, M. Magnor, P. Schueler, and H. P. Seidel. Combining 2D feature tracking and volume reconstruction for online video-based human motion capture. In *Proceedings of Pacific Graphics, Beijing, China*, pages 96–103, 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. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):780–785, 1997.

[16] Z.Y. Zhang. Flexible camera calibration by viewing a plane from unknown orientations. In *ICCV99*, pages 666–673, 1999.