

# Hand Tracking Using A Quadric Surface Model

R. Cipolla<sup>1</sup> B. Stenger<sup>1</sup> A. Thayananthan<sup>1</sup> P. H. S. Torr<sup>2</sup>

<sup>1</sup> University of Cambridge, Department of Engineering, Trumpington Street,  
Cambridge, CB2 1PZ, UK

<sup>2</sup> Microsoft Research Ltd., 7 J J Thomson Ave, Cambridge CB3 0FB, UK

## Abstract

*Within this paper a technique for model-based 3D hand tracking is presented. A hand model is built from a set of truncated quadrics, approximating the anatomy of a real hand with few parameters. Given that the projection of a quadric onto the image plane is a conic, the contours can be generated efficiently. These model contours are used as shape templates to evaluate possible matches in the current frame. The evaluation is done within a hierarchical Bayesian filtering framework, where the posterior distribution is computed efficiently using a tree of templates. We demonstrate the effectiveness of the technique by using it for tracking 3D articulated and non-rigid hand motion from monocular video sequences in front of a cluttered background.*

## A Introduction

Hand tracking has great potential as a tool for better human-computer interaction. Tracking hands, in particular articulated finger motion, is a challenging problem because the motion exhibits many degrees of freedom (DOF). Representing the hand pose by joint angles, the configuration space is 27 dimensional, 21 DOF for the joint angles and 6 for orientation and location. Given a kinematic hand model, one may attempt to use inverse kinematics to calculate the joint angles [19], however this problem is ill-posed when using a single view. It also requires exact feature localization, which is particularly difficult in the case of self-occlusion.

Most successful methods have followed the approach of using a geometric hand model, introduced by Rehg and Kanade [13] in the *DigitEyes* tracking system. Their hand model is constructed from truncated cylinders. The axes of these cylinders are projected into the image, and the distances to local edges are minimised using nonlinear optimisation. Heap and Hogg [9] use a deformable surface mesh model, which is constructed via principal component analysis (PCA) from example shapes obtained with an MRI scanner. This is essentially a 3D version of active shape models, and shape variation is captured by only a few principal components. The motion is not based on a physical deformation model and thus implausible finger motions can result. Wu *et al.* [20] model the articulated hand motion from data captured using a data glove. The tracker is based on

importance sampling, and hypotheses are generated by projecting a 'cardboard model' into the image. This model is constructed from planar patches, and thus the system is view-dependent.

It is clear that the performance of a model-based tracker depends on the type of the used model. However, there is a trade-off between accurate modelling, and efficient rendering and comparison with the image data. In fact this is generally true when modelling articulated objects for tracking, which is commonly done in the context of human body tracking (see [11] for a survey). A number of different models have been suggested in this context, using various primitives such as boxes, cylinders, ellipsoids or super-quadrics.

The next section describes the geometric hand model used in this paper. Section C reviews work on tree-based detection. A short introduction to Bayesian filtering is given in D, and in section E we introduce filtering using a tree-based estimator. Tracking results on video sequences of hand motion are shown in section F.

## B Modelling Hand Geometry

This section describes the construction of a hand model from truncated quadrics. The advantage of this method is that the object surface can be approximated with low complexity and that contours can be generated using tools from projective geometry. This hand model has previously been described in [15] but here it is used in a different tracking framework.

### B.1 Projective Geometry of Quadrics and Conics

A quadric is a second degree implicit surface in 3D space, and it can be represented in homogeneous coordinates by a symmetric  $4 \times 4$  matrix  $\mathbf{Q}$  [8]. The surface is defined by all points  $\mathbf{X} = [x, y, z, 1]^T$  satisfying the equation

$$\mathbf{X}^T \mathbf{Q} \mathbf{X} = 0. \quad (1)$$

Different families of quadrics are obtained from matrices  $\mathbf{Q}$  of different ranks. Particular cases of interest are:

- ellipsoids**, represented by matrices  $\mathbf{Q}$  with full rank;
- cones and cylinders**, represented by matrices  $\mathbf{Q}$  with  $\text{rank}(\mathbf{Q}) = 3$ ;
- a pair of planes**  $\pi_0$  and  $\pi_1$ , represented as  $\mathbf{Q} = \pi_0 \pi_1^T + \pi_1 \pi_0^T$  with  $\text{rank}(\mathbf{Q}) = 2$ .

Note that there are several other projective types of quadrics, such as hyperboloids or paraboloids, which like ellipsoids have a matrix of full rank. Under a Euclidean transformation  $\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix}$  the shape of a quadric is preserved, but in the new coordinate system  $\mathbf{Q}$  is represented by  $\hat{\mathbf{Q}} = \mathbf{T}^{-T} \mathbf{Q} \mathbf{T}^{-1}$ .

A quadric has nine degrees of freedom, corresponding to the independent elements of  $\mathbf{Q}$  up to a scale factor. Given a number of point correspondences

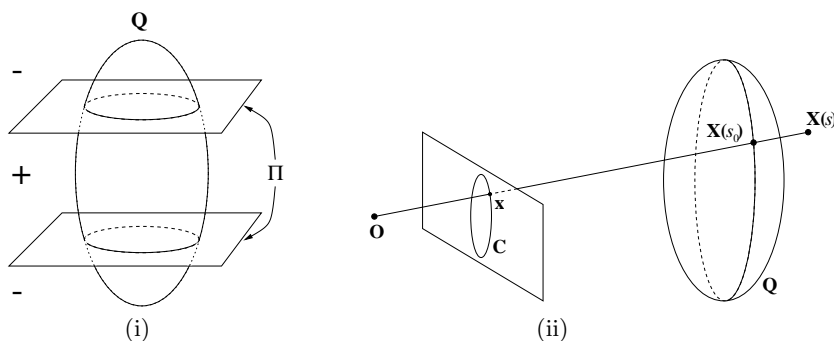
or outlines in multiple views, quadric surfaces can be reconstructed, as shown by Cross and Zisserman [6]. It is also suggested that for many objects using a piecewise quadric representation gives an accurate and compact surface representation. In order to employ quadrics for modelling such general shapes, it is necessary to truncate them. For any quadric  $\mathbf{Q}$  the truncated quadric  $\mathbf{Q}_{\Pi}$  can be obtained by finding points  $\mathbf{X}$  satisfying:

$$\mathbf{X}^T \mathbf{Q} \mathbf{X} = 0 \quad \text{and} \quad \mathbf{X}^T \mathbf{\Pi} \mathbf{X} \geq 0, \quad (2)$$

where  $\mathbf{\Pi}$  is a matrix representing a pair of clipping planes (see figure 1a). The image of a quadric  $\mathbf{Q} = \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{b}^T & c \end{bmatrix}$  seen from a normalised projective camera  $\tilde{\mathbf{P}} = [\mathbf{\Pi} \mid \mathbf{0}]$  is a conic  $\mathbf{C}$  given by

$$\mathbf{C} = c\mathbf{A} - \mathbf{b}\mathbf{b}^T, \quad (3)$$

as shown in figure 1b. In order to obtain the image of a quadric  $\mathbf{Q}$  in an arbitrary projective camera  $\mathbf{P}$  it is necessary to compute the transformation  $\mathbf{H}$  such that  $\mathbf{P}\mathbf{H} = [\mathbf{\Pi} \mid \mathbf{0}]$ . This normalising transformation is given by the matrix  $\mathbf{H} = [\mathbf{P}^\dagger \mid \mathbf{p}^\perp]$ , where  $\mathbf{P}^\dagger$  is the pseudo inverse of  $\mathbf{P}$  and  $\mathbf{p}^\perp$  is the camera centre or the null vector of  $\mathbf{P}$  (see [5]).

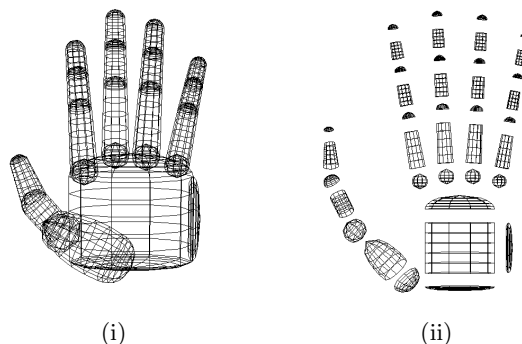


**Fig. 1. Projection of a quadric.** (a) A truncated quadric  $\mathbf{Q}_{\Pi}$ , here a truncated ellipsoid, can be obtained by finding points on quadric  $\mathbf{Q}$  which satisfy  $\mathbf{X}^T \mathbf{\Pi} \mathbf{X} \geq 0$ . (b) The projection of a quadric  $\mathbf{Q}$  into the image plane is a conic  $\mathbf{C}$ .

## B.2 Description of the Hand Model

The hand model is built using a set of quadrics  $\{\mathbf{Q}_i\}_{i=1}^q$ , representing the anatomy of a human hand as shown in figure 2. We use a hierarchical model with 27 degrees of freedom (DOF): 6 for the global hand position, 4 for the pose of each finger and 5 for the pose of the thumb. Starting from the palm and ending at the tips, the coordinate system of each quadric is defined relative to

the previous one in the hierarchy. The palm is modelled using a truncated cylinder, its top closed by a half-ellipsoid. Each finger consists of three segments of a cone, one for each phalanx. They are connected by hemispheres, representing the joints. A default shape is first obtained by taking measurements from a real hand. Given the image data, shape matching can be used to estimate a set of shape parameters, including finger lengths and a width parameter [17].



**Fig. 2. Geometric hand model.** *The hand model has 27 degrees of freedom is constructed using truncated quadrics as building blocks. Depicted is (a) a front view and (b) an exploded view.*

### B.3 Generation of the Contours

Each clipped quadric of the hand model is projected individually as described in section B.1, generating a list of clipped conics. For each conic matrix  $\mathbf{C}$  we use eigen-decomposition to obtain a factorisation given by

$$\mathbf{C} = \mathbf{T}^{-\mathbf{T}} \mathbf{D} \mathbf{T}^{-1}. \quad (4)$$

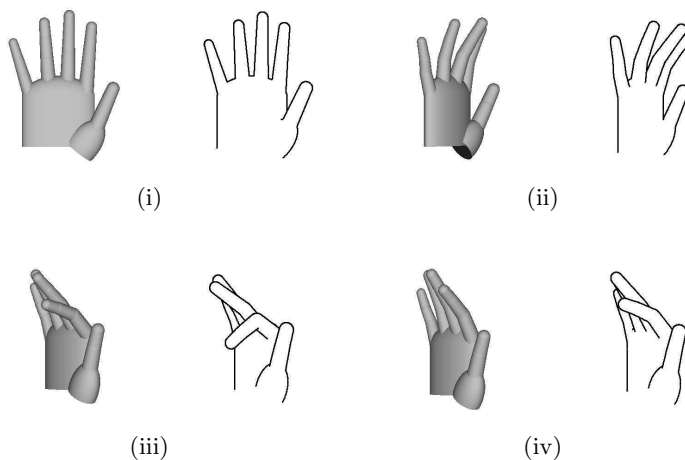
The diagonal matrix  $\mathbf{D}$  represents a conic aligned with the  $x$ - and  $y$ -axis and centred at the origin. The matrix  $\mathbf{T}$  is the Euclidean transformation that maps this conic onto  $\mathbf{C}$ . We can therefore draw  $\mathbf{C}$  by drawing  $\mathbf{D}$  and transforming the points according to  $\mathbf{T}$ . The drawing of  $\mathbf{D}$  is carried out by different methods, depending on its rank. For  $\text{rank}(\mathbf{D}) = 3$  we draw an ellipse, for  $\text{rank}(\mathbf{D}) = 2$  we draw a pair of lines.

The next step is occlusion handling. Consider a point  $\mathbf{x}$  on the conic  $\mathbf{C}$ , obtained by projecting the quadric  $\mathbf{Q}$ , as shown in figure 1. The camera centre and  $\mathbf{x}$  define a 3D ray  $L$ . Each point  $\mathbf{X} \in L$  is given by  $\mathbf{X}(s) = \begin{bmatrix} \mathbf{x} \\ s \end{bmatrix}$ , where  $s$  is a free parameter determining the depth of the point in space, such that the point  $\mathbf{X}(0)$  is at infinity and  $\mathbf{X}(\infty)$  is at the camera centre. The point of intersection of the

ray with the quadric  $\mathbf{Q}$  is found by solving the equation

$$\mathbf{X}(s)^T \mathbf{Q} \mathbf{X}(s) = 0 \quad (5)$$

for  $s$ . Writing  $\mathbf{Q} = \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{b}^T & c \end{bmatrix}$ , the unique solution of (5) is given by  $s_0 = -\mathbf{b}^T \mathbf{x} / c$ . In order to check if  $\mathbf{X}(s_0)$  is visible, (5) is solved for each of the other quadrics  $\mathbf{Q}_i$  of the hand-model. In the general case there are two solutions  $s_1^i$  and  $s_2^i$ , yielding the points where the ray intersects with quadric  $\mathbf{Q}_i$ . The point  $\mathbf{X}(s_0)$  is visible if  $s_0 \geq s_j^i \quad \forall i, j$ , in which case the point  $\mathbf{x}$  is drawn. Figure 3 shows examples of hand model projections.

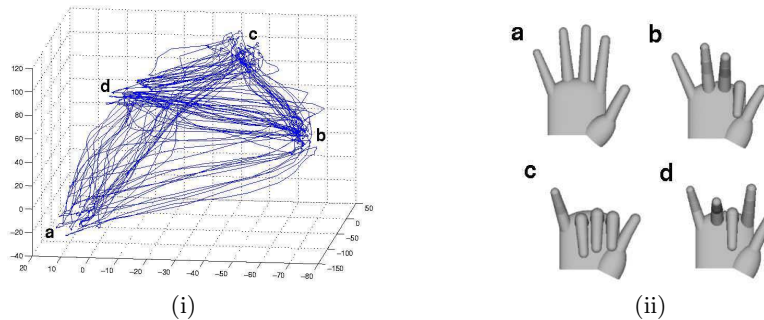


**Fig. 3. Examples of model projections.** (a)-(d) show different hand poses. For each example the 3D hand model is shown on the left and its projection into the image plane on the right. Note that self-occlusion is handled when generating the contours.

#### B.4 Learning Natural Hand Articulation

Model-based trackers commonly use a 3D geometric model with an underlying biomechanical deformation model [1, 2, 13]. Each finger can be modelled as a kinematic chain with 4 DOF, and the thumb with 5 DOF. Thus articulated hand motion lies in a 21 dimensional joint angle space. However, hand motion is highly constrained as each joint can only move within certain limits. Furthermore the motion of different joints is correlated, for example, most people find it difficult to bend the little finger while keeping the ring finger fully extended at the same time. Thus hand articulation is expected to lie in a compact region within the 21 dimensional angle space. We used a data glove to collect a large

number of joint angles in order to capture natural hand articulation. Experiments with 15 sets of joint angles captured from three different subjects, show that in all cases 95 percent of the variance is captured by the first eight principal components, in 10 cases within the first seven, which confirms the results reported by Wu *et al.* in [20]. Figure 4 shows trajectories projected onto the first three eigenvectors between a set of hand poses. As described in the next



**Fig. 4. Paths in the configuration space found by PCA.** (a) The figure shows a trajectory of projected hand state vectors onto the first three principal components. (b) The hand configurations corresponding to the four end points in (a).

section, this lower dimensional eigen-space will be quantised into a set of discrete states. Hand motion is then modelled by a first order Markov process between these states. Given a large amount of training data, higher order models can be learned.

### C Tree-Based Detection

For real applications the problem of tracker initialisation, as well as the handling of self-occlusion and cluttered backgrounds remain obstacles. Current state-of-the-art systems often employ a version of particle filtering, allowing for multiple hypotheses. The use of particle filters is primarily motivated by the need to overcome ambiguous frames in a video sequence so that the tracker is able to recover. Another way to overcome the problem of losing lock is to treat tracking as object detection at each frame. Thus if the target is lost in one frame, this does not affect any subsequent frame. Template based methods have yielded good results for locating deformable objects in a scene with no prior knowledge, e.g. for hands or pedestrians [2, 7, 14, 17]. These methods are made robust and efficient by the use of distance transforms such as the chamfer or Hausdorff distance between template and image [3, 10], which were originally developed for matching a single template. A key suggestion was that multiple templates

could be dealt with efficiently by building a template hierarchy and a coarse-to-fine search [7, 12]. The idea is to group similar templates and represent them with a single prototype template together with an estimate of the variance of the error within the cluster, which is used to define a matching threshold. The prototype is first compared to the image; only if the error is below the threshold are the templates within the cluster compared to the image. This clustering is done at various levels, resulting in a hierarchy, with the templates at the leaf level covering the space of all possible templates.

If a parametric object model is available, another option to build the tree is by partitioning the state space. Each level of the tree defines a partition with increasing resolution, the leaves defining the finest partition. Such a tree is depicted schematically in figure 5(a), for a single rotation parameter. This tree representation has the advantage that prior information is encoded efficiently, as templates with large distance in parameter space are likely to be in different sub-trees.

It may be argued that there is no need for a parametric model and that an exemplar-based approach could be followed, as by Toyama and Blake in [18]. However, for models with many degrees of freedom the storage space for templates becomes excessive. The use of a parametric model allows the combination of an on-line and off-line approach in the tree-based algorithm. Once the leaf level is reached more child templates can be generated for further optimisation. Hierarchical detection works well for locating a hand in images, and yet often there are ambiguous situations that could be resolved by using temporal information. The next section describes the Bayesian framework for filtering.

## D Bayesian Filtering

Filtering is the problem of estimating the state (hidden variables) of a system given a history of observations. Define, at time  $t$ , the state parameter vector as  $\boldsymbol{\theta}_t$ , and the data (observations) as  $\mathbf{D}_t$ , with  $\mathbf{D}_{1:t-1}$ , being the set of data from time 1 to  $t-1$ ; and the data  $\mathbf{D}_t$  are conditionally independent at each time step given the  $\boldsymbol{\theta}_t$ . In our specific application  $\boldsymbol{\theta}_t$  is the state of the hand (set of joint angles, location and orientation) and  $\mathbf{D}_t$  is the image at time  $t$  (or some set of features extracted from that image). Thus at time  $t$  the posterior distribution of the state vector is given by the following recursive relation

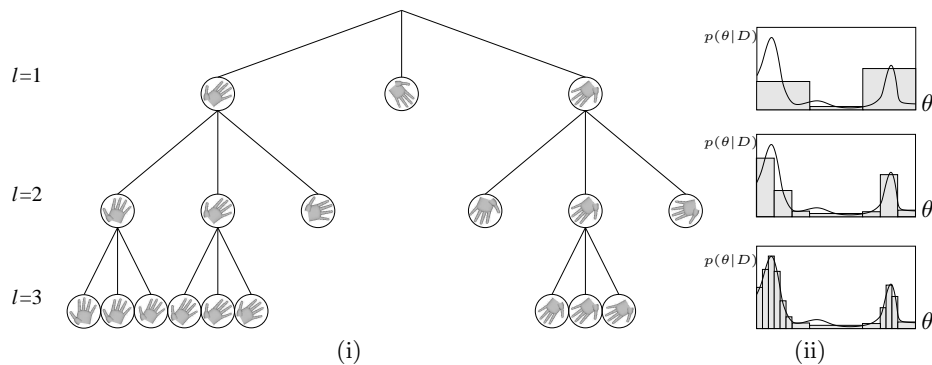
$$p(\boldsymbol{\theta}_t | \mathbf{D}_{1:t}) = \frac{p(\mathbf{D}_t | \boldsymbol{\theta}_t) p(\boldsymbol{\theta}_t | \mathbf{D}_{1:t-1})}{p(\mathbf{D}_t | \mathbf{D}_{1:t-1})}, \quad (6)$$

where the normalising constant is

$$p(\mathbf{D}_t | \mathbf{D}_{1:t-1}) = \int p(\mathbf{D}_t | \boldsymbol{\theta}_t) p(\boldsymbol{\theta}_t | \mathbf{D}_{1:t-1}) d\boldsymbol{\theta}_t. \quad (7)$$

The term  $p(\boldsymbol{\theta}_t | \mathbf{D}_{1:t-1})$  in (6) is obtained from the Chapman-Kolmogorov equation:

$$p(\boldsymbol{\theta}_t | \mathbf{D}_{1:t-1}) = \int p(\boldsymbol{\theta}_t | \boldsymbol{\theta}_{t-1}) p(\boldsymbol{\theta}_{t-1} | \mathbf{D}_{1:t-1}) d\boldsymbol{\theta}_{t-1} \quad (8)$$



**Fig. 5. Tree-based estimation of the posterior density.** (a) Associated with the nodes at each level is a non-overlapping set in the state space, defining a partition of the state space (here rotation angle). The posterior distribution for each node is evaluated using the centre of each set, depicted by a hand rotated by a specific angle. Sub-trees of nodes with low posterior probability are not further evaluated. (b) Corresponding posterior density (continuous) and the piecewise constant approximation using tree-based estimation. The modes of the distribution are approximated with higher precision at each level.

with the initial prior pdf  $p(\boldsymbol{\theta}_0|\mathbf{D}_0)$  assumed known. It can be seen that (6) and (8) both involve integrals. Except for certain simple distributions these integrals are intractable and so approximation methods must be used. As has been mentioned, Monte Carlo methods represent one way of evaluating these integrals. Alternatively, hierarchical detection provides a very efficient way to evaluate the likelihood  $p(\mathbf{D}_t|\boldsymbol{\theta}_t)$  in a deterministic manner, even when the state space is high dimensional; as the number of templates in the tree increases exponentially with the number of levels in the tree. This leads us to consider dividing up the state space into non-overlapping sets, just as the templates in the tree cover the regions of parameter space. Typically this methodology has been applied using an evenly spaced grid and is thus exponentially expensive as the dimension of the state space increases. In this paper we combine the tracking process with the empirically successful process of tree-based detection as laid out in section C resulting in an efficient deterministic filter.

## E Filtering Using a Tree-Based Estimator

Our aim is to design an algorithm that can take advantage of the efficiency of the tree-based search whilst also yielding a good approximation to Bayesian filtering. We design a grid-based filter, in which a multi-resolution partition is provided by the tree as given in Section C. Thus we will consider a grid defined by the leaves of the tree. Because the distribution is characterised by being almost zero in large regions of the state space with some isolated peaks, many of



the grid regions can be discarded as possessing negligible probability mass. The tree-based search provides an efficient way to rapidly concentrate computation on significant regions. At the lowest level of the tree the posterior distribution will be assumed to be piecewise constant. This distribution will be mostly zero for many of the leaves. At each tree level the regions with high posterior are identified and explored in finer detail in the next level (Figure 5b). It is to be expected that the higher levels will not yield accurate approximations to the posterior. However, just as for the case of detection, the upper levels of the tree can be used to discard inadequate hypotheses, for which the negative log posterior of the set exceeds a threshold value. The thresholds at the higher levels of the tree are set conservatively so as to not discard good hypotheses too soon. The equations of Bayesian filtering (6)-(8), are recast to update these states. For more details see [16].

### E.1 Formulating the Likelihood

A key ingredient for any tracker is the likelihood function  $p(\mathbf{D}_t|\boldsymbol{\theta}_t)$ , which relates the observations  $\mathbf{D}_t$  to the unknown state  $\boldsymbol{\theta}_t$ . For hand tracking finding good features and a likelihood function is challenging, as there are few features which can be detected and tracked reliably. Colour values and edges contours appear to be suitable and have been used frequently in the past [2, 20]. Thus the data is taken to be composed of two sets of observations, those from edge data  $\mathbf{D}_t^{edge}$  and from colour data  $\mathbf{D}_t^{col}$ . The likelihood function is assumed to factor as

$$p(\mathbf{D}_t|\boldsymbol{\theta}_t) = p(\mathbf{D}_t^{edge}|\boldsymbol{\theta}_t) p(\mathbf{D}_t^{col}|\boldsymbol{\theta}_t). \quad (9)$$

The likelihood term for edge contours  $p(\mathbf{D}_t^{edge}|\boldsymbol{\theta}_t)$  is based on the chamfer distance function [3, 4]. Given the set of projected model contour points  $\mathcal{U} = \{\mathbf{u}_i\}_{i=1}^n$  and the set of Canny edge points  $\mathcal{V} = \{\mathbf{v}_j\}_{j=1}^m$ , a quadratic chamfer distance function is given by

$$d_{cham}^2(\mathcal{U}, \mathcal{V}) = \frac{1}{n} \sum_{i=1}^n d^2(i, \mathcal{V}), \quad (10)$$

where  $d(i, \mathcal{V}) = \max(\min_{v_j \in \mathcal{V}} \|u_i - v_j\|, \tau)$  is the thresholded distance between the point,  $u_i \in \mathcal{U}$ , and its closest point in  $\mathcal{V}$ . Using a threshold value  $\tau$  makes the matching more robust to outliers and missing edges. The chamfer distance between two shapes can be computed efficiently using a distance transform, where the template edge points are correlated with the distance transform of the image edge map. Edge orientation is included by computing the distance only for edges with similar orientation, in order to make the distance function more robust [12]. We also exploit the fact that part of an edge normal on the interior of the contour should be skin-coloured.

In constructing the colour likelihood function  $p(\mathbf{D}_t^{col}|\boldsymbol{\theta}_t)$ , we seek to explain all the image pixel data given the proposed state. Given a state, the pixels in the image  $\mathcal{I}$  are partitioned into a set of object pixels  $\mathcal{O}$ , and a set of background

pixels  $\mathcal{B}$ . Assuming pixel-wise independence, the likelihood can be factored as

$$p(\mathbf{D}_t^{col}|\boldsymbol{\theta}_t) = \prod_{k \in \mathcal{I}} p(I_t(k)|\boldsymbol{\theta}_t) = \prod_{o \in \mathcal{O}} p(I_t(o)|\boldsymbol{\theta}_t) \prod_{b \in \mathcal{B}} p(I_t(b)|\boldsymbol{\theta}_t), \quad (11)$$

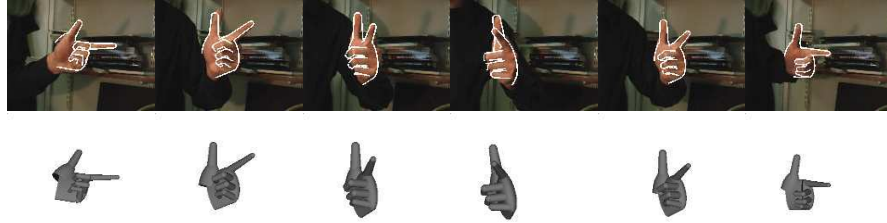
where  $I_t(k)$  is the intensity normalised rg-colour vector at pixel location  $k$  at time  $t$ . The object colour distribution is modeled as a Gaussian distribution in the normalised colour space, and a uniform distribution is assumed for the background. For efficiency, we evaluate only the edge likelihood term while traversing the tree, and incorporate the colour likelihood only at the leaf level.

## F Results

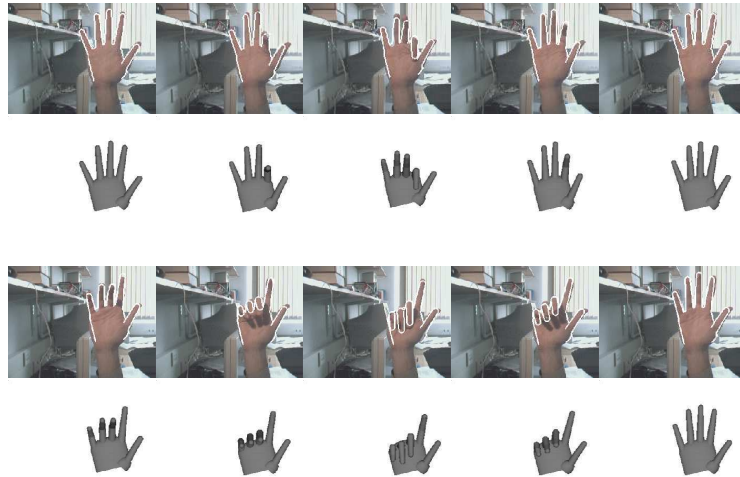
We demonstrate the effectiveness of our technique by tracking both hand motion and finger articulation in cluttered scenes using a single camera. The results reveal the ability of the tree-structure to handle ambiguity arising from self-occlusion and 3D motion. In the first sequence (figure 6) we track the global 3D motion of a pointing hand. The 3D rotations are limited to a hemisphere. At the leaf level, the tree has the following resolutions: 15 degrees in two 3D rotations, 10 degrees in image rotation and 5 different scales. These 12,960 templates are then combined with a search at 2-pixel resolution in the image translation space. In the second example (figure 7) finger articulation is tracked while the hand is making transitions between different types of gestures. The tree is built by partitioning a lower dimensional eigen-space. Applying PCA to the data set shows that more than 96 percent of the variance is captured within the first four principal components, thus we partition the four dimensional eigen-space. The number of nodes at the leaf level in this case is 9,163. In the third sequence (figure 8) tracking is demonstrated for global hand motion together with finger articulation. The manifolds in section B.4 are used to model the articulation. The articulation parameters for the thumb and fingers are approximated by the first 2 eigenvectors of the joint angle data set obtained from opening and closing of the hand. For this sequence the range of global hand motion is restricted to a smaller region, but it still has 6 DOF. In total 35,000 templates are used at the leaf level. The tree evaluation takes approximately 2 to 5 seconds per frame on a 1GHz Pentium IV machine. Note that in all cases the hand model was automatically initialised by searching the complete tree in the first frame of the sequence.

## G Summary and Conclusion

Within this paper we have described a model-based hand tracking system which overcomes some of the major obstacles which have limited the use of hand trackers in practical applications. These are the handling of self-occlusion, tracking in cluttered backgrounds, and tracker initialisation.



**Fig. 6. Tracking a pointing hand in front of clutter.** *The images are shown with projected contours superimposed (top) and corresponding 3D avatar models (bottom), which are estimated using our tree-based algorithm. The hand is translating and rotating. A 2D deformable template would have problems coping with topological shape changes caused by self-occlusion.*



**Fig. 7. Tracking finger articulation.** *In this sequence a number of different finger motions are tracked. The images are shown with projected contours superimposed (top) and corresponding 3D avatar models (bottom), which are estimated using our tree-based filter. The nodes in the tree are found by hierarchical clustering of training data in the parameter space, and dynamic information is encoded as transition probabilities between the clusters.*

Our algorithm uses a tree of templates, generated from a 3D geometric hand model. The model is constructed from a set of truncated quadrics, and its contours can be projected into the image plane while handling self-occlusion. Articulated hand motion is learned from training data collected using a data glove. The likelihood cost function is based on the chamfer distance between projected contours and edges in the image. Additionally, edge orientation and skin colour



**Fig. 8. Tracking a hand opening and closing with rigid body motion.** *This sequence is challenging because the hand undergoes translation and rotation while opening and closing the fingers. 6 DOF for rigid body motion plus 2 DOF using manifolds for finger flexion and extension are tracked successfully with our tree-based algorithm.*

information is used, making the matching more robust in cluttered backgrounds. The problem of tracker initialisation is solved by searching the tree in the first frame without the use of any prior information. We have tested the tracking method on a number of sequences including hand articulation and cluttered backgrounds. Furthermore within these sequences the hand undergoes rotations leading to significant topological changes in the projected contours. The tracker performs well even in these circumstances.

*Acknowledgements* The authors would like to thank the Gottlieb Daimler–and Karl Benz–Foundation, the EPSRC, the Gates Cambridge Trust, and the Overseas Research Scholarship Programme for their support.

## References

1. K. N. An, E. Y. Chao, W. P. Cooney, and R. L. Linscheid. Normative model of human hand for biomechanical analysis. *J. Biomechanics*, 12:775–788, 1979.
2. V. Athitsos and S. Sclaroff. Estimating 3D hand pose from a cluttered image. In *Proc. Conf. Computer Vision and Pattern Recognition*, Madison, USA, June 2003. to appear.
3. H. G. Barrow, J. M. Tenenbaum, R. C. Bolles, and H. C. Wolf. Parametric correspondence and chamfer matching: Two new techniques for image matching. In *Proc. 5th Int. Joint Conf. Artificial Intelligence*, pages 659–663, 1977.

4. G. Borgefors. Hierarchical chamfer matching: A parametric edge matching algorithm. *IEEE Trans. Pattern Analysis and Machine Intell.*, 10(6):849–865, November 1988.
5. R. Cipolla and P. J. Giblin. *Visual Motion of Curves and Surfaces*. Cambridge University Press, Cambridge, UK, 1999.
6. G. Cross and A. Zisserman. Quadric reconstruction from dual-space geometry. In *Proc. 6th Int. Conf. on Computer Vision*, pages 25–31, Bombay, India, January 1998.
7. D. M. Gavrilu. Pedestrian detection from a moving vehicle. In *Proc. 6th European Conf. on Computer Vision*, volume II, pages 37–49, Dublin, Ireland, June/July 2000.
8. R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, Cambridge, UK, 2000.
9. A. J. Heap and D. C. Hogg. Towards 3-D hand tracking using a deformable model. In *2nd International Face and Gesture Recognition Conference*, pages 140–145, Killington, USA, October 1996.
10. D. P. Huttenlocher, J. J. Noh, and W. J. Rucklidge. Tracking non-rigid objects in complex scenes. In *Proc. 4th Int. Conf. on Computer Vision*, pages 93–101, Berlin, May 1993.
11. T. B. Moeslund and E. Granum. A survey of computer vision-based human motion capture. *Computer Vision and Image Understanding*, 81(3):231–268, 2001.
12. C. F. Olson and D. P. Huttenlocher. Automatic target recognition by matching oriented edge pixels. *Transactions on Image Processing*, 6(1):103–113, January 1997.
13. J. M. Rehg. *Visual Analysis of High DOF Articulated Objects with Application to Hand Tracking*. PhD thesis, Carnegie Mellon University, Dept. of Electrical and Computer Engineering, 1995. TR CMU-CS-95-138.
14. N. Shimada, K. Kimura, and Y. Shirai. Real-time 3-D hand posture estimation based on 2-D appearance retrieval using monocular camera. In *Proc. Int. WS. RATFG-RTS*, pages 23–30, Vancouver, Canada, July 2001.
15. B. Stenger, P. R. S. Mendonça, and R. Cipolla. Model based 3D tracking of an articulated hand. In *Proc. Conf. Computer Vision and Pattern Recognition*, volume II, pages 310–315, Kauai, USA, December 2001.
16. B. Stenger, A. Thayananthan, P. H. S. Torr, and R. Cipolla. Hand tracking using a tree-based estimator. Technical Report CUED/F-INFENG/TR 456, University of Cambridge, Department of Engineering, 2003.
17. A. Thayananthan, B. Stenger, P. H. S. Torr, and R. Cipolla. Shape context and chamfer matching in cluttered scenes. In *Proc. Conf. Computer Vision and Pattern Recognition*, Madison, USA, June 2003. to appear.
18. K. Toyama and A. Blake. Probabilistic tracking with exemplars in a metric space. *Int. Journal of Computer Vision*, pages 9–19, June 2002.
19. Y. Wu and T. S. Huang. Capturing articulated human hand motion: A divide-and-conquer approach. In *Proc. 7th Int. Conf. on Computer Vision*, volume I, pages 606–611, Corfu, Greece, September 1999.
20. Y. Wu, J. Y. Lin, and T. S. Huang. Capturing natural hand articulation. In *Proc. 8th Int. Conf. on Computer Vision*, volume II, pages 426–432, Vancouver, Canada, July 2001.