Wormholes in Shape Space: Tracking through Discontinuous Changes in Shape

Tony Heap and David Hogg
School of Computer Studies, University of Leeds, Leeds LS2 9JT, UK

Abstract

Existing object tracking algorithms generally use some form of local *optimisation, assuming that an object's position and shape change smoothly over time. In some situations this assumption is not valid: the trackable shape of an object may change discontinuously, for example if it is the 2D silhouette of a 3D object.*

In this paper we propose a novel method for modelling temporal shape discontinuities explicitly. Allowable shapes are represented as a union of (learned) bounded regions within a shape space. Discontinuous shape changes are described in terms of transitions between these regions. Transition probabilities are learned from training sequences and stored in a Markov model. In this way we can create 'wormholes' in shape space. Tracking with such models is via an adaptation of the CONDENSATION *algorithm.*

1 Introduction

Models of shape have been used widely as an aid to tracking deformable objects [11, 2, 4, 1]. Existing tracking algorithms based on such models generally rely on the assumption that objects move and deform smoothly over time; consequently object features and edges are located in an image via a local search from the object's previous position. There are cases where this behaviour is not adhered to, the main example being when tracking the *silhouette* of a 3D object.

Object silhouettes change shape smoothly for most of the time, but in certain cases there can be a discontinuous shape change. For example, in the case of human hands, this occurs when the fingers close together and the gaps between them disappear or when the hand turns sideways (see Figure 1). Similar effects are seen on the arms and legs of a walking person. Local optimisation-based approaches generally cannot track objects through such discontinuities, usually becoming trapped in a local maximum and sometimes failing more catastrophically.

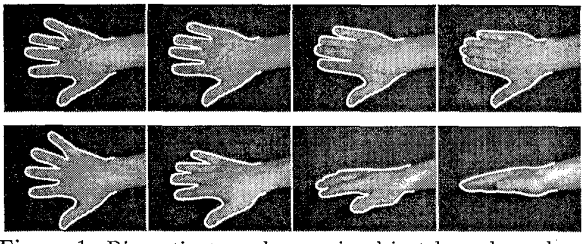

Figure 1: Discontinuous changes in object boundary shape due to deformation (top row) and rotation (bottom row).

In this paper we propose a novel method for modelling shape transitions which encompasses both continuous and discontinuous changes in a non-deterministic framework. We show how tracking with our new model is possible via an adaptation of Isard and Blake's CONDENSATION algorithm, and also describe an extension to CONDENSATION which gives an improvement in performance. We perform an evaluation of our new technique in comparison to previous approaches, and draw some conclusions.

2 Modelling Discontinuous Changes in Shape

The discontinuous shape changes described above are *specific* as opposed to *arbitrary*: they occur repeatedly between certain pairs of shapes. As such, they can be learned from training sequences containing examples of characteristic object movement.

Before we describe how this learning process proceeds, we must firstly discuss a suitable model of object shape.

2.1 Hierarchical PDMs

Point Distribution Models (PDMs) [5] are statistical models of shape, described in terms of the Cartesian coordinates of a number of *landmarks*, and constructed from training examples via principal component analysis (PCA). The main components of a PDM are a mean shape and a small number of orthonormal variation vectors (eigenshapes) which are added to the mean shape in a weighted combination to produce a particular shape. The ranges of scalar weights for the variation vectors form the axes for an n-dimensional model ***shape space***.

The shape space produced by a PDM is often unsatisfactory because it is both linear and continuous. Non-linear object deformations (such as bending or pivoting) must be modelled as a combination of linear deformations, and models of objects with two or more distinct classes of shape must also include all the (often nonsensical) shapes in between.

This results in large areas of model shape space which do not represent valid shapes; models are not *specific* enough, and tracking using such models is generally poor or unsuccessful. The effect is particularly noticeable when models are generated from automatically-collected training data (see Figure 2).

Hierarchical PDMs (HPDMs) [8] are an extension to PDMs, based on the work of Bregler and Omohundro [3]; after the initial PCA has been performed, the model shape space is further constrained in order to improve model specificity. A 'piecewise linear' approach is used: a number of evenly-spaced prototypes are found using k-means cluster analysis on the training data as projected into shape space. Each cluster then undergoes a separate PCA to give a set of locally-linear patches (hyperellipsoids), the union of which describes the improved, constrained shape space. To ensure this space is

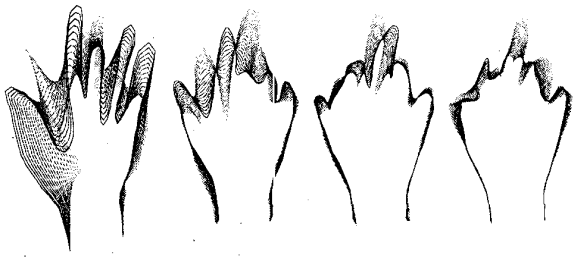

Figure 2: The first four modes of variation for an automatically trained hand PDM. Many invalid shapes appear.

continuous between adjacent patches, a degree of overlap is used in the clusters. Figure 3 demonstrates the process.

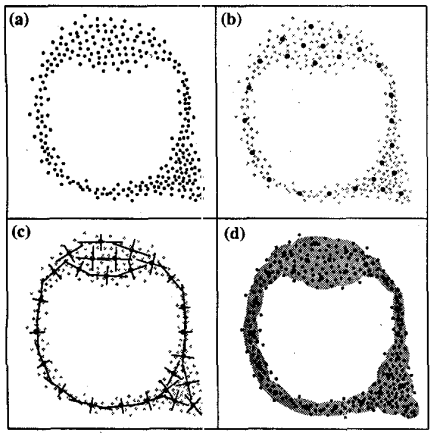

Figure 3: Building a Hierarchical PDM; (a) training data projected into a 2D global shape space, (b) k-means cluster centres, (c) principal axes for each linear patch and (d) the improved, constrained shape space.

For the HPDM to be of practical use it must be possible, given a general shape, to find the nearest *valid* shape. Full details of this are given in [8], but one simple algorithm involves finding the nearest linear patch to the shape's position in (unconstrained) shape space and forcing the shape to lie within the associated bounding hyperellipsoid.

Such an algorithm allows for the application of the model shape constraints to any given shape. The overall result is a model of shape which is much more *specific*; far fewer invalid shapes are included in the model. HPDMs can represent virtually any topology, including loop structures, variations in dimensionality and, most importantly, regions split into two or more separate parts. The only prerequisite is enough training data to build an accurate model. Figure 4 gives an example of the nearest equivalent to modes of variation for the HPDM: traversals through the constrained shape space in various directions. This space need not be continuous; for this reason large jumps appear in some of the traversals.

2.2 Building a Shape Transition Matrix

The aim is to construct a model of the shape changes which a particular object experiences, as learned from training sequences of typical object movement.

The HPDM produces a model of shape which is represented by a set of local patches that cover all the valid

Figure 4: Four different shape space traversals for an automatically trained hand HPDM.

regions within a shape space. The key to the approach is the observation that these patches are small enough to cover only minor variations in object shape, and could thus be looked at as being discrete states for the object shape. This allows for the use of a Markovian representation of object shape dynamics, with each state representing a different shape, and the state transition probabilities reflecting typical shape changes.

To this end, a Markov state transition matrix $\mathbf{T}$ is constructed as follows: E pairs of sequential training shapes, $\mathbf{x}_e$ and $\mathbf{y}_e$, are used. A preliminary matrix $\mathbf{T}'$ is first calculated using:

$$\mathbf{T}' = \sum_{e=1}^{E} \mathbf{p}(\mathbf{x}_e)\mathbf{p}(\mathbf{y}_e)^T \tag{1}$$

where $\mathbf{p}(\mathbf{x})$ is a vector of probabilities $p(\mathbf{x}, c)$, being the probability that shape $\mathbf{x}$ is a member of patch c. We have used the simple rule that $p(\mathbf{x}, c) = 1$ if c is the nearest patch and $p(\mathbf{x}, c) = 0$ otherwise, but a more complex function could be derived based on relative distances to patches.

The preliminary matrix $\mathbf{T}'$ is then normalised with respect to each row so that $\mathbf{T}_{a,b}$ gives the probability of a transition from patch a to patch b:

$$\mathbf{T}_{a,b} = \mathbf{T}'_{a,b} / \sum_i \mathbf{T}'_{a,i} \tag{2}$$

The matrix $\mathbf{T}$ provides a probabilistic model of object shape transitions. $\mathbf{T}$ would be expected to have large values along its diagonal, indicating that for the majority of the time a shape remains in the same patch. Large off-diagonal values represent learned shape transitions, many of which will be to adjacent patches. However, if a specific discontinuous change appears repeatedly in the training data, this too will give rise to a high transition probability.

For practical use, it is convenient to construct a *cumulative* version of the transition matrix; this makes for more efficient probability sampling:

$$\mathbf{C}_{r,c} = \sum_{i=1}^{c} \mathbf{T}_{r,i} \tag{3}$$

This model of shape transition can be used for generation or prediction of object deformation over time; we illustrate the latter case by applying it to model-based object tracking.

3 Tracking

Our model of shape transition is based on probabilities and, as such, lends itself to a non-deterministic

approach to tracking. For this reason we have chosen to make use of the stochastic CONDENSATION algorithm.

3.1 The CONDENSATION Algorithm

In Isard and Blake's CONDENSATION (Stochastic *Con*ditional *Den*sity Propag*ation*) algorithm [10], the location of an object is represented not by a single set of model parameters, but by a probability density function over the model parameter space. A model of conditional probability (learned from training sequences) is used to propagate the pdf over time. In other words, given the pdf at time t there is a mechanism to predict the pdf at time $t+1$, based on a simple model of object motion. The new pdf is consolidated and refined by referring to the current image. Specifically, a *fitness* function is used to determine the goodness-of-fit of model to image at any point in the pdf.

In practice, the pdf is represented by a population of samples drawn from the parameter space. Each one has its fitness calculated and *factored sampling* is used to choose seeds for propagation. The algorithm proceeds as follows:

1. To initialise, generate a random population of N candidate model shapes $\mathbf{s}_1 \ldots \mathbf{s}_N$. We use a Point Distribution Model [5], so a shape $\mathbf{s}$ is a vector containing centroid coordinates x and y, size s, orientation θ, and deformation parameters $b_1 \ldots b_T$.
2. Iterate:
   (a) Calculate the fitness $f_i = F(\mathbf{s}_i)$ of each shape.
   (b) To produce each shape in the *new* population:
      i. Select a member of the old population randomly, with probability of selecting shape j being equal to $f_j / \sum_i f_i$
      ii. Apply the propagation function $P(\mathbf{s})$ to produce the new shape. Isard and Blake use a 'Fokker-Planck' (drift-and-spread) equation: $P(\mathbf{s}) = \mathbf{A}\mathbf{s} + \mathbf{B}\omega$, where ω is a vector of independent standard normal random variables, and $\mathbf{A}$ and $\mathbf{B}$ are constant matrices learned from training sequences of characteristic movements; $\mathbf{A}$ defines the deterministic 'drift' in the model and $\mathbf{B}$ is used to scale and orientate ω.

Figure 5 illustrates the algorithm, and a more detailed explanation is given in [10]. A major issue is the question of what the 'correct' object shape is, given a pdf represented by a discrete population of candidate shapes. One possibility is to take the *modal* (highest fitness) shape, but this can sometimes produce a noisy output. Alternatively the statistical mean can be found, using:

$$\bar{\mathbf{s}} = \sum f_i \mathbf{s}_i / \sum f_i \qquad (4)$$

However, this has the undesirable feature of interpolating between peaks in the pdf. Neither solution is perfect but both are simple to calculate.

The fitness function $F(\mathbf{s})$ should be designed to produce a value which indicates the 'goodness of fit' of a shape to an object in the image. For a contour model, a simple such algorithm examines image pixels along a line normal to the contour at each control point and returns a fitness proportional to the number, proximity and strength of edges found in the image. More details can be found in [6].

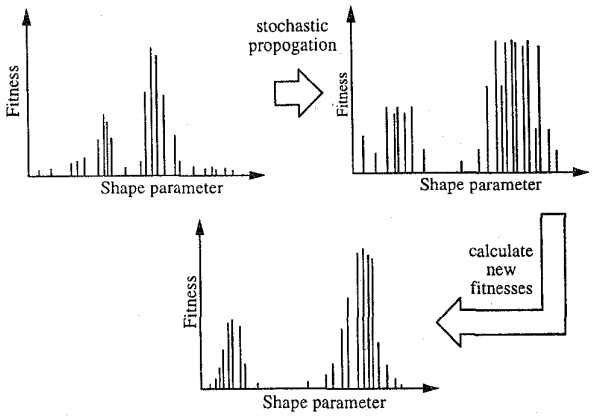

Figure 5: Propagation of a shape population with only one shape parameter using the CONDENSATION algorithm. Each vertical line represents a candidate shape in the population. The fitness peaks indicate promising solutions.

The benefits of the CONDENSATION are as follows. Firstly, it can support multiple hypotheses; this is represented by a pdf with multiple peaks. Secondly, it recovers well from failure; the stochastic nature of the algorithm allows it to escape from local maxima. Finally, it incorporates a level of prediction, which improves the speed of convergence and the quality of results over, for example, a Genetic Algorithm.

The prediction aspect is embedded in the propagation equations. Currently these have two elements; a deterministic term which allows for simple drifting of the pdf, and a stochastic term which encourages spreading of the pdf. Although the tracker *can* escape from local maxima (due to the stochastic term), the underlying dynamical model is still based on an assumption of smooth, continuous object movement.

3.2 Adapting CONDENSATION for Discontinuous Shape Changes

The tracking of sudden shape changes is possible via a modification to the *propagation* step of the CONDENSATION algorithm. The specification for propagation is quite general; the abstract notion is a conditional pdf $p(\text{shape at time } t+1 | \text{ shape at time } t)$ which propagates the shape pdf over time. The algorithmic requirement is a function which, given a vector $\mathbf{s}$, being the shape at time t, returns a new vector $\mathbf{s}'$, being a *possible* shape at time $t+1$, sampled from the conditional pdf.

Our learned model of object shape transition can be used to provide the control parameters for a Markov process-based propagation algorithm, using the cumulative transition matrix defined in (3). This proceeds as follows:

1. Determine the HPDM patch membership of the source shape $\mathbf{s}$. Currently we simply assume it is member of the nearest patch (but as for the transition matrix learning algorithm a more complex function could be derived based on relative distances to patches). Label this patch a.
2. Use row a of the cumulative transition matrix $\mathbf{C}$ to select probabilistically the destination patch b. To

do this, generate a random number z from a uniform distribution over the range $[0, 1]$ and choose the smallest b such that $C_{a,b} > z$.

3. Set $\mathbf{s}'$ (the destination shape) to be at a position within patch b. If $b = a$ then $\mathbf{s}'$ is set to $\mathbf{s}$ plus a random perturbation. If $b \neq a$ then $\mathbf{s}'$ is set to the centre of cluster b plus a random perturbation. In either case the perturbation is normally distributed, scaled and orientated with respect to the principal axes of linear patch b.

Under this algorithm, a small number of samples from the shape population 'leap' through wormholes in shape space at every iteration. In most cases these leaps will result in low fitness candidates, which are unlikely to survive into the following iteration, but if a sudden shape change *has* occurred then a high fitness candidate will be produced and other population members will quickly migrate to the new fitness peak. This has the desired result of tracking such discontinuous shape changes.

4 An Aside : Improving Performance and Speed of CONDENSATION

A drawback of the CONDENSATION algorithm over a deterministic local-optimisation (ie. snake-like) tracker is that of speed. This is because in every frame there is a population of up to several hundred candidate shapes to process, whereas deterministic trackers process only one shape.

We have investigated the construction of a hybrid tracker which makes use of both stochastic and deterministic tracking in order to produce a system which is robust but which is also faster and more accurate than a purely stochastic tracker.

The hybrid tracker is based on a stochastic tracker as described above, but after each new candidate shape is produced via the propagation function, one or more iterations of local optimisation are performed (using the snake-like Active Shape Model or ASM [4]), in order to refine the shape and hence improve its fitness. The number of iterations applied to each shape is proportional to its fitness: more cycles are allocated to promising shapes and fewer to poor shapes. A fixed number of cycles are shared out in order to keep the tracker running at a steady rate. The result is that the peaks in the pdf are better represented, and consequently a much smaller population is required for accurate tracking. A similar approach has been used to good effect by Hill *et al* in combining Genetic Algorithms with ASMs [9].

Figure 6 illustrates the propagation of a population of shapes under the hybrid algorithm. The local optimisation step results in better clustering of samples around the fitness peaks and higher overall fitness levels.

Figure 7 shows a comparison of three different tracking algorithms—purely stochastic (CONDENSATION), purely deterministic (ASM) and hybrid—on some typical sequences of hand movement. The left-hand graph shows comparative performance on moderate speed hand movement and the right-hand graph is for the same sequence at high speed (only every eighth frame is used). The model fitness in each case is determined using the fitness function outlined in Section 3.1, and the failure threshold shown is a (manually determined) level

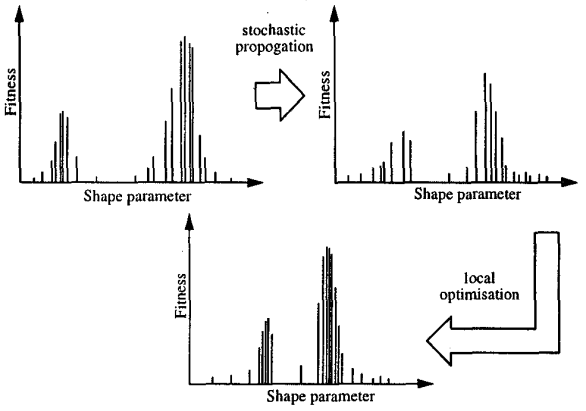

Figure 6: Propagation of a shape population using the Hybrid tracker.

below which a tracking failure is deemed to have occurred. The CONDENSATION tracker had a population of 150 shapes, this number being chosen to give reasonable performance. The hybrid tracker was allowed only 50 shapes, and 50 cycles of local optimisation were shared out per iteration; in total it was approximately 1.5 times as fast as the CONDENSATION tracker. The ASM tracker was appended with a (very slow) failure recovery mechanism which was activated if the fitness dropped below the failure threshold. This was for the purposes of illustration only.

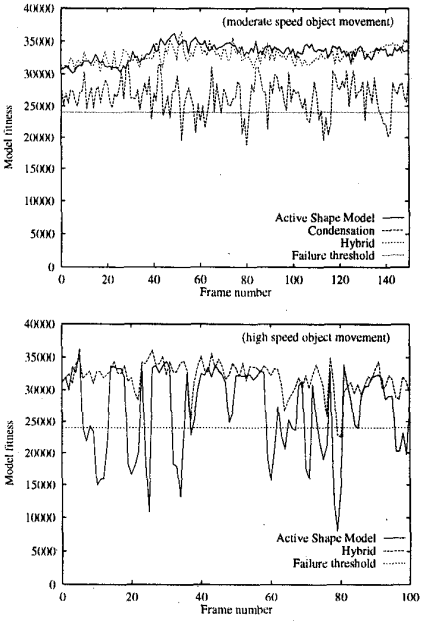

Figure 7: Graphs comparing the hybrid tracker to a CONDENSATION tracker and an Active Shape Model tracker, on moderate (left) and high (right) speed hand movements. The failure threshold was chosen manually.

The hybrid algorithm clearly out-performs the purely stochastic tracker, despite having a smaller population. At moderate speeds the ASM and hybrid trackers' performance is comparable, but at high speeds the ASM tracker is not robust, requiring re-initialisation every

few frames, whereas the hybrid tracker experiences only one perceived failure, and it recovers from this without any help. In summary the hybrid tracker is faster and more accurate than the purely stochastic tracker, and more robust but slower than the deterministic ASM tracker.

5 Evaluation

In order to demonstrate our new approach to tracking, we performed a comparative evaluation with two other trackers: a standard CONDENSATION tracker (using Fokker-Planck dynamics) and the snake-like Active Shape Model (ASM) tracker.

The underlying shape model used for all three trackers was the same; a HPDM of the human hand, similar to that illustrated in Figure 4, but with a gesture set chosen specifically to produce discontinuities in the shape space. Training data was collected by recording a sequence of gestures performed against a homogeneous background, and applying a simple boundary-finding algorithm to each frame in the sequence. Each training example consisted of 100 evenly spaced landmarks around the hand silhouette boundary.

The HPDM was then constructed as outlined in Section 2.1. Figure 8 shows the training data (as projected into the first two dimensions of shape space) and the 20 linear patches which were constructed from it.

Figure 8: The automatically captured training examples (projected into the first two dimensions of shape space) and the CPDM linear patch principal axes.

Because the model was built using training examples from a continuous sequence, the same data could also be used to build the Markov transition matrix which models the conditional probability density function. Figure 9 shows some example conditional pdfs for both Fokker-Planck (top row) and Markov model (bottom row) propagation algorithms, produced by choosing a single 'seed' position in shape space and generating a large number of destination positions.

The Fokker-Planck algorithm essentially produces a Gaussian distribution around the seed, truncated by the HPDM constraints. The Markov model algorithm uses the transition matrix to generate more general conditional pdfs. This model has captured valid jumps through shape space, indicated by the multiple dark areas. Note that the Fokker-Planck algorithm in this example is untrained; training would alter the shape of the pdf, but only in terms of orientation and eccentricity of the Gaussian; more complex pdfs are not possible under this model.

To compare the trackers' performance, an unseen sequence of hand movement was filmed. The sequence

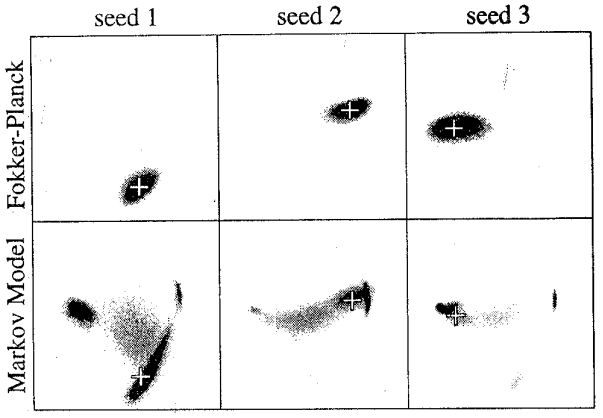

Figure 9: Some example conditional pdfs generated from single seeds via both the Fokker-Planck and Markov model algorithms. The crosses indicate the seed positions.

featured many gestures, including the types of sudden shape changes which have caused problems for our previous trackers. Figure 10 shows a graph of the fitness score (as outlined in Section 3.1) for each tracker over the video sequence (for the CONDENSATION-based algorithms the *modal* fitness value was used). The failure threshold gives the approximate fitness value above which a good fit is deemed to have occurred, but no failure recovery mechanisms were employed for this experiment.

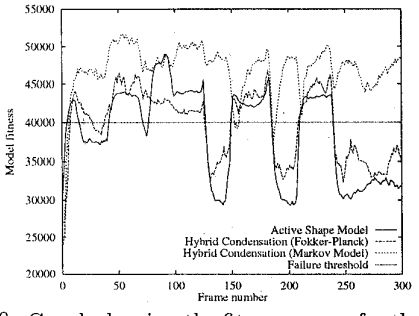

Figure 10: Graph showing the fitness scores for the different tracking algorithms on an image sequence containing sudden shape changes.

The positions of the sudden shape changes in the sequence (approximately every 30 frames) can easily be identified as the fitness for all three trackers changes suddenly. The ASM tracker is unable to respond at all to the shape changes; there are no image edges to follow in order to deform the model shape in the right way. This is indicated by the large fluctuations in the fitness score. The Fokker-Planck tracker has slightly improved performance: the same large fitness fluctuations are seen, but in some cases the tracker starts to recover (eg. frames 125 to 150) until another shape change occurs. The Markov model tracker, however, clearly out-performs the other two. Its fitness level does initially drop after a shape change, but it quickly manages to return to full strength, indicating that the underlying population has managed to 'migrate' to the correct shape.

Figure 11 provides a more detailed analysis of how the two different stochastic propagation algorithms han-

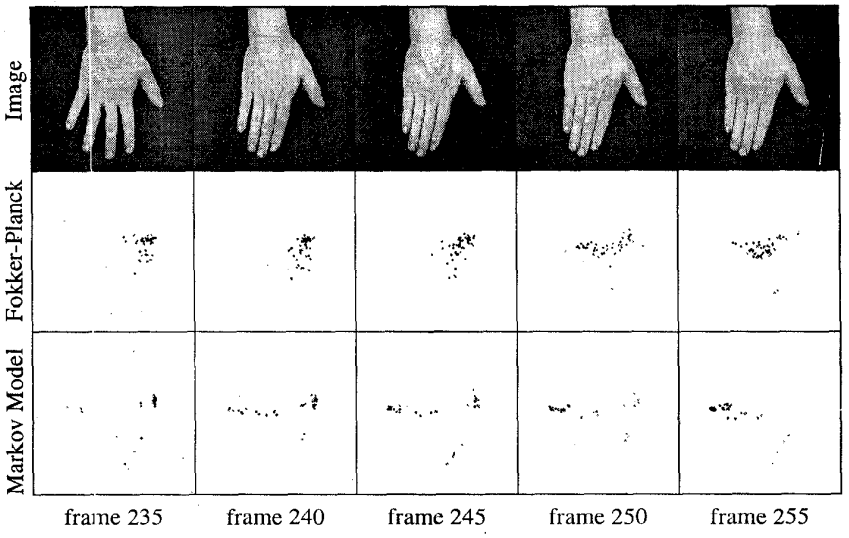

Figure 11: Coping with a sudden change: the image sequence, and the population distributions for the Fokker-Planck and Markov model tracking algorithms, as projected into the first two dimensions of shape space.

dle a sudden shape change. A section of the above experiment was isolated and snapshots were taken every five frames showing the input image and the positions in shape space (projected into 2D) of the populations for both the Fokker-Planck and Markov model propagation algorithms.

Initially, the populations for both trackers are clustered around the correct position in shape space (towards the right hand side). Note that the Markov model population is more focussed, but with a small number of members in distant areas; these areas are potential destinations of a sudden shape space jump. As soon as the shape change occurs, the Markov model population begins to migrate to the left, and by frame 245 a new cluster has formed on the left hand side. The Fokker-Planck population is much slower to move, and even after 20 frames it is only approximately half way there.

6 Conclusions

We have discussed the modelling of 'wormholes' in shape space using a union of regions representation of shape along with a learned Markov model containing transition probabilities. We have described an adaptation of the CONDENSATION algorithm to allow the tracking of temporal discontinuities using this model, and shown how this new method gives improved performance over previous approaches.

These results are interesting for two reasons. Firstly, the tracking of 3D objects using 2D shape models on their silhouettes is made possible in the general case. Secondly, tracking using models which have been trained *fully automatically* is made possible. Previously, manual annotation of training examples was necessary to ensure that the models produced were sufficiently *specific* and *continuous* for tracking purposes.

A substantial amount of training data is required to build the underlying HPDM shape model and the transition matrix; however, training is fully automatic and thus large amounts of training data can be collected in a short time. In the case of the hand models the training phase involved performing several gestures repeatedly under a rostrum camera, which took around five minutes.

References

[1] A. Baumberg and D. Hogg. An efficient method for contour tracking using active shape models. In IEEE Computer Society Press, editor, *IEEE Workshop on Motion of Non-rigid and Articulated Objects*, pages 194–199, November 1994. Also available as ftp://agora.leeds.ac.uk/scs/doc/reports/1994/94_11.ps.Z.

[2] A. Blake, R. Curwen, and A. Zisserman. A framework for spatiotemporal control in the tracking of visual contours. *International Journal of Computer Vision*, 11(2):127–145, 1993.

[3] C. Bregler and S. Omohundro. Surface learning with applications to lipreading. In J D Cowan, G Tesauro, and J Alspector, editors, *Advances in neural information processing systems 6*, 1994.

[4] T.F. Cootes and C.J. Taylor. Active shape models - 'Smart Snakes'. In *Proc. BMVC*, pages 266–275, Leeds, UK, 1992. Springer-Verlag.

[5] T.F. Cootes, C.J. Taylor, D.H. Cooper, and J. Graham. Training models of shape from sets of examples. In *Proc. BMVC*, pages 9–18, Leeds, UK, 1992. Springer-Verlag.

[6] A.J. Heap. Real-time hand tracking and gesture recognition using Smart Snakes. In *Proc. Interface to Human and Virtual Worlds*, Montpellier, France, June 1995. Also available as ftp://ftp.cam-orl.co.uk/pub/docs/ORL/1995/tr.95.1.ps.Z.

[7] A.J. Heap and D.C. Hogg. Towards 3D hand tracking using a deformable model. In *Proc. 2nd International Face and Gesture Recognition Conference*, Killington, Vermont, 1996.

[8] A.J. Heap and D.C. Hogg. Improving specificity in PDMs using a hierarchical approach. In *Proc. BMVC*, Colchester, UK, 1997. BMVA Press.

[9] A. Hill, T.F. Cootes, and C.J. Taylor. A generic system for image interpretation using flexible templates. In *Proc. BMVC*, pages 276–285, Leeds, UK, 1992. Springer-Verlag.

[10] M. Isard and A. Blake. Contour tracking by stochastic propogation of conditional density. In B Buxton and R Cipolla, editors, *Proc. ECCV '96*, volume I, pages 343–356, 1996.

[11] A. Pentland and B. Horowitz. Recovery of nonrigid motion and structure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(7):730–742, 1991.