Optimal and Suboptimal Shape Tracking Based on Multiple Switched Dynamic Models [*]

Jorge S. Marques [†]
IST / ISR
Av. Rovisco Pais
1049-001 Lisboa, Portugal

João M. Lemos
IST / INESC
R. Alves Redol
1000 Lisboa, Portugal

Abstract

Object tracking based on multiple models has been recently advocated as a way to tackle sudden changes of shape or motion parameters. This paper addresses the estimation of time-varying parameters described by a bank of shared state stochastic models, switched according to a probabilistic mechanism (hidden Markov chain). A state estimation algorithm is proposed, based on the propagation of Gaussian mixtures in a multi-model framework. For preventing mode explosion a pruning strategy combining mode elimination and merging is used. This is shown to be better than employing either just elimination or merging. Examples dealing with image processing of moving objects are provided.

[*]this work was partially supported by PRAXIS XXI in the scope of project EEI/12050/1998.

[†]Please send all the correspondence to Jorge S. Marques, Torre Norte, IST, Av. Rovisco Pais, 1049-001 Lisbon, Portugal; Email: jsm@isr.ist.utl.pt

1 Introduction

Tracking objects with time-varying shapes is a challenging problem. Three points make this problem difficult: the dependence of the object shape on the camera point of view; the shape changes which occur during the experiment; and the existence of complex motion trajectories and dynamics. The first difficulty arises even in the case of static shape analysis while the remaining difficulties are caused by object or camera motion. In the static case, the use of multiple models has been advocated by a number of researchers, *e.g.* [5, 6, 17, 22]. A typical example is the representation of the object as a linear combination of multiple views leading to the concept of shape subspace. Three views are enough to represent a rigid object under orthographic projection, if no occlusion occurs [22]. More views are needed to avoid the effect of occlusions and to deal with non-rigid objects [5]. The model views are either specified by the user, by selecting a set of typical images of the object, or they are obtained by training procedures (*e.g.*, using Principal Component Analysis [6, 17]).

These principles can be extended to deal with moving objects in video sequences. When an object undergoes motion or shape deformation during the observation interval, it is often represented as a sequence of views or pre-defined image features such as correlation type features [8], corners, strokes [10] or silhouettes [5]. Time dependence among consecutive views or features is exploited by DTW (Dynamic Time Warping)[7], hidden Markov models or by syntactic Pattern Recognition methods. These approaches have been extensively studied in the scope of Human-machine interface and activity recognition [7, 24, 23].

Previous works represent the object motion as a sequence of static models, *e.g.*, views or image features. A different approach is adopted here. The data is represented by a sequence of models as before, but dynamic models are used instead: the video sequence is approximated by a library of stochastic dynamic models switched according to a probabilistic rule. This type of systems is used in Control theory to represent abrupt changes in industrial plants [21, 18] as well as in other problems [4], including target tracking for surveillance [2, 9] but little work has been done to use them in the dynamic scene analysis. A recent step in this direction is the work [12] based on the use of non parametric methods to deal with multiple motion models. Discontinuous changes of the object shape during tracking operations have been recently addressed in [11] using the so-called wormholes. This work is again based on non parametric descriptions of the probability distributions.

This paper has the same goal. However a different approach is adopted since the *a posteriori* distribution of the unknown parameters is represented by a mixture of Gaussians. This leads to a parametric and optimal update of the *a posteriori* density. Furthermore, it is shown that switched models are a useful tool to track rapid shape transitions. The use of multiple models with shared state [16] allows to tackle complex shapes and nonlinear or changing dynamics by combining simple local models. The overall algorithm complexity is thus reduced. Further simplification may also be achieved by pruning least significant modes of the density according to suitable criteria. As shown below, both procedures can be combined with advantage. This results in suboptimal algorithms.

The paper is organised as follows. Section 2 provides a background on multi-model dynamic systems. The data sequence is described as the output of a bank of linear systems equipped with a switching mechanism. The estimation of the outputs and switching schedule is addressed in Section 3. Section 4 describes the application of switched shared state filters in the context of object tracking. Section 5 presents experimental results and section 6 concludes the paper.

2 Switched Dynamic Systems

Let $x_t \in R^n$ denote the variables to be estimated and $k_t \in \{1, \ldots, M\}$ the label of the dynamic/sensor model which describes the evolution of the process variables at the instant t (discrete variable). This means that the data is assumed to be generated by one (numbered k_t) of several systems differing either in dynamics or observation model. Two examples further addressed below are a moving hand making signs and moving lips. In both cases there is noticeable shape deformation.

The hybrid state is defined by the pair $z_t = (x_t, k_t)$. The first component, x_t, is a state vector shared by all possible local models. The second, k_t, denotes the index of the model currently generating data. The hybrid state summarises all the past information needed to generate future realisations of the stochastic processes x_t, k_t. Estimates are made on the basis of measured variables, y_t, which are instantaneous observations of the state variables. Figure 1 summarises the direct dependencies among these variables. At time t, it is assumed that the index k_t depends on k_{t-1}. The vector x_t depends on x_{t-1} and k_t. For reasons to discuss below a dependency on k_{t-1} is also assumed. The observation y_t depends on k_t and x_t.

Assuming that z_t is a first order Markov process, then z_t is characterised by the transition density $p(z_t/z_{t-1})$. The transition density can be split as follows

$$p(z_t/z_{t-1}) = p(x_t/k_t, z_{t-1})p(k_t/z_{t-1}). \tag{1}$$

It will be assumed that k_t is a Markov chain with transition matrix T such that $p(k_t = j/x_{t-1}, k_{t-1} = i) = T_{ij}$. This means that model switching is independent of previous x values, provided that k_{t-1} is known (Fig. 1). Instead of defining $p(x_t/k_t, z_{t-1})$ directly, it is assumed that x_t is the output of a stochastic difference equation

$$x_t = A_{k_{t-1},k_t} x_{t-1} + b_{k_{t-1},k_t} + w_t, \tag{2}$$

$$y_t = C_{k_t} x_t + d_{k_t} + v_t \tag{3}$$

where w_t, v_t are independent Gaussian processes: $w_t \sim N(0, Q_{k_{t-1},k_t})$, $v_t \sim N(0, R_{k_t})$. Expression (2) is a stochastic difference equation whose parameter matrices A, b, Q depend on k_t and k_{t-1}.

In image analysis special attention must be paid to transitions, since there is no guarantee that x_t should have the same meaning or even the same dimension for two different values of k_t (see the comments at the end of example 2). Two cases are considered: i) steady state operation ($k_t = k_{t-1}$)characterised by M linear systems and ii) transition epochs ($k_t \neq k_{t-1}$) which are described by M(M-1) equations in order to consider all the admissible transitions.

Two simple examples illustrate the need of probability densities associated to transitions.

Example 1

Consider two shape models (Fig. 2a): two rectangles of different length which have not been aligned. The object present in the image corresponds either to the first or to the second model. Let x_t be the rotation angle needed to align one of the shape models with the object observed in the image. Every time the model changes in response to a change in the object being observed, this must be compensated by a jump in x_t. This can be done by assigning

b_{k_{t-1},k_t} an appropriate value. Figure 2b shows a sequence of observed shapes and Fig. 2c the corresponding state trajectory. As expected, the state variable is discontinuous at the transition.

Example 2

Consider (Fig. 3) a moving object whose shape and position evolve according to two models: an affine model up to time t_0 and a translation model from time t_0 on. The state vector has a different number of components in both cases (six parameters are needed for defining an affine transform while only two have to be specified in the case of translation motion).

Assuming a Wiener model for the evolution of the coefficients, matrix A is a 6×6 identity matrix up to time $t_0 - 1$ and a 2×2 identity matrix from time $t_0 + 1$ on. At the transition instant, matrix A is 2×6 matrix. Choosing this matrix depends on both the indices k_t and k_{t-1}. This allows to use state vectors with different dimensions and to perform appropriate switching between the corresponding state spaces and probability density functions defined in those spaces. In this case, A is no longer a square matrix. It is remarked that the template must be updated at the beginning of each transition among models.

An alternative would be to freeze some parameters of the high dimension representation (e.g., affine) when a lower dimensional model (e.g., translation) is to be used. Fig. 4 shows an example in which a square, whose observations are corrupted by noise (Fig. 4a), follows a sequence of straight and curved paths. The output of the tracker segmenting both types of movements is shown in Fig. 4(b).

3 Density Propagation

Density propagation may be performed either by an optimal or a suboptimal algorithm. In the case of optimal algorithms all the mixture modes are propagated. This leads to an exponential increase of complexity. Pruning least significant modes of the density according to a suitable criterion results in suboptimal algorithms.

3.1 Optimal Algorithm

The main question which has to be addressed is the following: given a set of observations $Y_t = (y_1, \ldots, y_t)$, what is the posterior density $p(z_t/Y_t)$? From this, we can estimate z_t, *e.g.*, using the MAP criterion.

Using the law of total probabilities, the *a posteriori* density is given by

$$p(z_t/Y_t) = \sum_{K_{t-1}} p(x_t, K_t/Y_t) = \sum_{K_{t-1}} c_{K_t} p(x_t/Z_t) \tag{4}$$

where $K_t = (k_1, \ldots, k_t)$ is the model label sequence, $Z_t = (z_1, \ldots, z_t)$ is the hybrid state sequence and $c_{K_t} = P(K_t/Y_t)$. The density $p(x_t/Z_t)$, which corresponds to the case of known dynamic and sensor models is normal $N(\hat{x}_{K_t}, P_{K_t})$ with mean $\hat{x}_{K_t}$ and covariance matrix P_{K_t}, updated by Kalman filtering [2]. Therefore, $p(z_t/Y_t)$ is a mixture of Gaussians, each of them is associated to a specific model sequence K_t. Since $K_t \in \{1, \ldots, M\}^t$, there are M^t different model sequences and the mixture will have M^t components.

The computation of the mean and covariance of each component is organised in a tree structure where each branch corresponds to an iteration of a Kalman filter (see fig. 5). This optimal structure cannot be directly implemented and some complexity reduction schemes have to be devised to avoid the combinatorial explosion.

Consider now the computation of the mixing parameters $c_{K_t} = P(K_t/Y_t)$. Since the model sequence is a Markov chain with transition probabilities T_{ij}, the mixing parameters can be recursively updated (see Appendix 1) by

$$c_{K_t} = \alpha T_{k_{t-1},k_t} G(y_t) c_{K_{t-1}} \tag{5}$$

where G is a Gaussian density function (see Appendix 1) and α is a constant obtained from the normalisation condition $\sum c_{K_t} = 1$. As explained before, there is a mixture mode associated to each tree node. The algorithm used to compute the parameters associated to each node is summarised in Table I.

3.2 Sub-optimal Algorithms

In practice, the number of components of the Gaussian mixture cannot grow to infinity and must be limited. Several strategies have been proposed for this sake in Control literature [21]. In this paper, two methods are used to achieve this goal: component elimination and merging. The first method discards components with mixing parameters, c_{K_t}, smaller than a given threshold (*e.g.*, $T_e =^{-3}$). These components produce a negligible contribution to the mixture density. The second method tries to avoid multiple components with close densities by merging then into a single component. The divergence is used for deciding whether two components are similar (the divergence is computed for all pairs of components; the pair with smallest divergence is merged if the divergence is below a given threshold T_m; the process continues until there is no pair meeting the merging conditions). Here, divergence is defined as in [19]. Accordingly, the divergence between two normal distributions, $N(\mu, P), N(\mu', P')$ is given by [19]

$$D = \tfrac{1}{2}(\mu - \mu')^T (P^{-1} + P'^{-1})(\mu - \mu') + \tfrac{1}{2} tr\{P^{-1}P' + P'^{-1}P - 2I\} \tag{6}$$

Another possibility would be to employ Kullback-Lleibler divergence. A similar criterion is used in [14] to approximate a periodic function by a Gaussian mixture in the context of nonlinear phase estimation.

3.3 Example

An example illustrating density propagation with the suboptimal algorithm described is now given.

Example 3

Consider two point targets moving on a line with random velocity (Fig. 6). Suppose there is a sensor which provides the coordinates of one of the points (we do not know which) at each instant of time and the measurement is corrupted by noise. We wish to estimate the location of *both* points at *every* instant of time from the noisy observations.

This is a state estimation problem with interrupted observations which can be formulated using multiple models. The state vector $x = (x^1, x^2)$ contains the points' coordinates. It will be assumed that the points locations are the outputs of the stochastic equation

$$x_t = \begin{bmatrix} 0.995 & 0 \\ 0 & 0.85 \end{bmatrix} x_{t-1} + w_t \tag{7}$$

where $w_t \sim N(0, Q), Q = diag(0.5, 4)$ and the sensor equations are

$$\begin{aligned} Model\ 1: \quad y_t &= [1\ \ 0]x_t + v_t \\ Model\ 2: \quad y_t &= [0\ \ 1]x_t + v_t \end{aligned} \tag{8}$$

with $v_t \sim N(0, 0.3)$. The model transitions are described by a Markov chain with transition matrix T ($T_{11} = T_{22} = 0.9$). This state dynamic model is constant in time but observations are modeled either by model 1 or model 2. It is not *a priori* known which of these models is active at a given time.

Figure 7 shows the output of the state estimation algorithm described in table I for a single experiment. Figure 7a shows the observations available to locate the points (try to guess their motion from this graphic only). Figure 7b shows which target is measured at each time instant (solid line) and the most probable value of k_t computed from the mixture coefficients (dotted line). The evolution of the state components, x^1, x^2, (solid line) and their MAP estimates (dotted line) are displayed in Fig. 7c,d. This experiment shows that the algorithm manages to locate both targets and to guess which target is being sensed, most of the time. Finally, Figs. 7e-i show the density function (Gaussian mixture) $p(x_t, k_t/Y_t)$, $k_t = 1, 2$ for the first 5 instants as well as the true state value (vertical line; remark the line is sometimes hidden by the state density function). The best model sequence estimated from Fig. 7e-h is given by the sequence of indices 11222 which agrees with Fig. 7b.

To assess the performance of pruning, the two-point problem was solved considering four strategies:

i) no pruning

ii) mode elimination (threshold $T_e = 10^{-3}$),

iii) mode merging (threshold $T_m = 5 \times 10^{-2}$) and

iv) mode merging and elimination (thresholds $T_e = 10^{-3}, T_m = 5 \times 10^{-2}$).

In all these experiments the maximum number of modes allowed was 200. When the limit is exceeded the mixture modes with smallest coefficients are discarded.

The *a posteriori* density was propagated using the algorithm described in this paper during the first 100 instants. The mixture densities obtained with pruning (cases ii, iii, iv) are compared with the density obtained without pruning (case i) The difference was evaluated using the divergence [19]

$$d(p_1, p_2) = \frac{1}{2} \int (p_1(x) - p_2(x)) \log \frac{p_1(x)}{p_2(x)} dx \tag{9}$$

Since there is no closed form expression for the divergence between two mixtures of Gaussians, Monte Carlo techniques were used instead to evaluate the integral.

Equation (9) can be rewritten as a sum of the expected values of appropriate log functions (Kullback-Leibler divergences)

$$d(p_1, p_2) = \frac{1}{2} \int p_1(x) \log \frac{p_1(x)}{p_2(x)} dx + \frac{1}{2} \int p_2(x) \log \frac{p_2(x)}{p_1(x)} dx \tag{10}$$

It is remarked that in the special case of Gaussian distributions this reduces to (6).

Each integral was computed in two steps. First, 10^5 realisations of the variable x were sampled from the density p_1 (*viz.* p_2). Then the average value of $\log p_1/p_2$ (*viz.* $\log p_2/p_1$) was computed.

Fig. 8(a) shows the number of modes and (b) the divergence obtained by the different pruning strategies considered. As seen, the combination of merging and elimination always yields the smallest number of modes. Average values are seen in Table II.

The simultaneous use of mode elimination and merging is clearly better than each of them used in isolation. As seen from the table, elimination+merging is almost as close to pruning than just elimination and much closer than just merging. The combination of elimination and merging requires however less than four times the number of modes if either the elimination or merging is used in isolation. Indeed, in this example, very good results are achieved with approximately 10 modes (5 modes per model).

Fig. 9 provides a visual illustration of this fact. While both densities are seen as quite similar (a) was obtained with 100 Gaussians and (b) with just 8 Gaussians, selected by the merging+elimination method.

4 Application to Tracking

The methods considered in section 3 may be applied to a variety of problems. This section specialises these methods for tracking of moving objects whose shape and dynamics change in time.

4.1 Dynamic Shape Modeling

Consider an image sequence of a moving object. We wish to estimate the evolution of the shape and motion parameters. By shape it is meant either the outer boundary of the object or a set of strokes. There are several ways to represent shape and motion evolution as the output of stochastic difference equations. In this paper, it is assumed that the edges detected in the image are transformed versions of a set of reference (model) strokes. Each stroke is a curve in R^2, represented by a continuous function defined on an interval $I \subset R$ [3]. Let $c_t : I \to R^2$ be a parametric representation of the curve detected in the t-th image and $c^i : I \to R^2, i = 1, \ldots, M$, the parametric representation of the reference curves associated with M shape models. It is assumed that

$$c_t = \mathcal{T}_t c^{k_t} + v \tag{11}$$

where $\mathcal{T}_t$ is a geometric transform, $k_t \in \{1, \ldots, M\}$ is the model label and v is a noise curve. The geometric transform $\mathcal{T}_t$ conveys motion and distortion information. Several transforms are considered in this paper. The affine transform is a popular choice since it provides enough flexibility to represent the motion of planar and even tri-dimensional objects under orthographic projection [22]. For the sake of simplicity, it will be assumed that $\mathcal{T}_t$ is either a translation or an affine transform. Therefore, the image curve $c_t(s) = (c_{1t}, c_{2t})$ (where s is the parameter corresponding to the curve parameterisation considered) is given by

Translation:

$$\begin{aligned} c_{1t}(s) &= c_1^{k_t}(s) + x_{1t} + v_1(s) \\ c_{2t}(s) &= c_2^{k_t}(s) + x_{2t} + v_2(s) \end{aligned} \tag{12}$$

Affine Transform:

$$\begin{aligned} c_{1t}(s) &= c_1^{k_t}(s)x_{1t} + c_2^{k_t}(s)x_{2t} + x_{3t} + v_1(s) \\ c_{2t}(s) &= c_1^{k_t}(s)x_{4t} + c_2^{k_t}(s)x_{5t} + x_{6t} + v_2(s) \end{aligned} \tag{13}$$

where $s \in I$ is a parameter which defines the location of a point on the stroke, $c_t^k(s) = (c_{1t}^k, c_{2t}^k)$, $v(s) = (v_{1t}, v_{2t})$ and x_{1t}, x_{2t} (translation) or $x_{1t}, \ldots, x_{6t}$ (affine) are the motion parameters at instant t. It is not realistic to assume that the whole contour c_t is known. In many cases, only samples $c_t(s_1), \ldots, c_t(s_m)$ of the image strokes are extracted from the t-th image, using appropriate image analysis algorithms [3]. These samples are the observations available at time t which should be used to retrieve the object boundary.

Assuming that the object moves during the acquisition process, dynamic equations have to be devised for describing the evolution of the motion parameters. Let

$$x_t = [x_{1t}, x_{2t}, \dot{x}_{1t}, \dot{x}_{2t}]^T \qquad \text{(translation model) or} \tag{14}$$

$$x_t = [x_{1t}, \ldots, x_{6t}, \dot{x}_{1t}, \ldots, \dot{x}_{6t}]^T \qquad \text{(affine model).} \tag{15}$$

It is assumed that x_t is the output of a difference equation (2).

Let us now consider the sensor model. Image processing provides samples $c_t(s_1), ..., c_t(s_m)$. Let y_t, c_t^i, v_t be $2m \times 1$ vectors with the coordinates of all sampling points stacked in a single column (*e.g.*, $y_t = [c_{1t}(s_1) \ldots c_{1t}(s_m) c_{2t}(s_1) \ \ldots c_{2t}(s_m)]^T$). Then, (12,13) can be written as (3) where

Translation:

$$C_i = \begin{bmatrix} 1_{m\times 1} & O_{m\times 1} & O_{m\times 2} \\ O_{m\times 1} & 1_{m\times 1} & O_{m\times 2} \end{bmatrix} \tag{16}$$

$$d_i = [c_{1t}^i(s_1) \ldots c_{1t}^i(s_m) c_{2t}^i(s_1) \ldots c_{2t}^i(s_m)]^T \tag{17}$$

Affine Transform:

$$d_i = O_{2m\times 1} \tag{18}$$

$$M_i = \begin{bmatrix} c_1^i(s_1) & c_2^i(s_1) & 1 \\ \vdots & \vdots & \vdots \\ c_1^i(s_m) & c_2^i(s_m) & 1 \end{bmatrix} \tag{19}$$

$$C_i = \begin{bmatrix} M^i & O_{m\times 3} & O_{m\times 6} \\ O_{m\times 3} & M^i & O_{m\times 6} \end{bmatrix} \tag{20}$$

In the translation case, matrix C_i is shape independent. Shape information in conveyed in d_i while in the case of the affine transform shape information is contained in C_i.

Equations (2-3) with the above definitions define a switched dynamic system which is able to represent sudden changes of shape (changes appearing in matrices of (3)), or motion (changes appearing in the matrices of (2)), or both, provided that the object are approximated by one of the reference shapes.

4.2 Feature Detection

It is assumed that the shape model is attracted by feature points detected in the image. Several methods are available for detecting image features, *e.g.*, by using line searching along the normal directions at specific contour points [3, 20] or by computing the data centroids using competitive learning methods [1]. In all of these, feature detection requires an estimate of the object shape in the next frame (predicted shape). A mean square error estimate of the object shape $\hat{y}_t^- = E\{y_t/Y_{t-1}\}$ is used in this paper. It is remarked that the detected features depend on the predicted shape. Indeed, only the vicinity of the predicted contour is usually considered by feature detection methods. In order to render feature detection more independent with respect to estimation, some authors [12] use multiple predictors.

Since the density of the predicted state, $p(x_t/Y_{t-1})$, is a mixture of Gaussians, shape prediction can be written as follows (see details in Appendix 2).

$$\hat{y}_t^- = \sum_{i=1}^{M} C_i \hat{x}_t^{i-} + d_t^{i-} \tag{21}$$

where

$$\hat{x}_t^{i-} = \sum_{K_t : k_t = i} c_{K_t}^- \hat{x}_{K_t}^- \tag{22}$$

$$d_t^{i-} = d_i \sum_{K_t : k_t = i} c_{K_t}^- \tag{23}$$

$$c_{K_t}^- = T_{k_{t-1} k_t} c_{K_{t-1}} \tag{24}$$

In the last three equations, the sum is performed for all the label sequences (tree leaves), K_t, such that $k_t = i$.

Shape prediction performed by (21) is a weighted average of the predicted shapes and motions produced by each of the models, weighted by the model likelihood. This combined predictor based on different shape models is easily computed as shown.

5 Results

The proposed algorithm was used for tracking moving objects with significant shape changes in image sequences. A simple dynamic model was adopted to describe the evolution of the motion parameters: the derivative of each parameter is modeled by a Wiener process, leading to a dynamic equation

$$x_t = \begin{bmatrix} I & I \\ O & I \end{bmatrix} x_{t-1} + w_t \tag{25}$$

where x_t ($x_t \in R^4$ in the case of translation, example 4 below, $x_t \in R^{12}$ in the case of affine transform, example 5 below, see definitions in equations [14,15]) is a state vector containing the motion parameters and their derivatives, I is the identity matrix and O the null matrix. It is assumed that the object shape in the first image is known. For each new image the following operations are performed:

- shape prediction according to $\hat{y}^- = E\{y/Y_{t-1}\}$;
- feature detection by constrained clustering methods [1];
- update of mixture components according to table I;
- component reduction;
- parameter estimation using the MAP method.

Example 4

Figure 10 shows examples of lip motion estimated with the above multi-model tracking algorithm. Three models are considered (see Fig. 10a). At each instant of time the algorithm manages to select the most appropriate model to represent the lips as well as the motion parameters (Fig. 10b-h). Lips are represented by 4 parameters (2 translation coordinates and their derivatives). The proposed algorithm manages to propagate the *a posteriori* distribution by using mixtures of Gaussians. The image analysis operations needed to extract shape information from the observed images are based on the constrained clustering techniques developed in [1]

Example 5

Figure 11 shows tracking results obtained with a sequence of a moving hand, assuming that the reference shapes are transformed according to an affine model (a 12D state vector is used in this experiment). It is remarked that this example can not be tackled by the translation model since the image undergoes rotation from frame to frame. Three shape models are considered as shown in fig. 11a. Fig. 11b-f displays the selected shape model transformed using the estimates of the affine parameters. This example illustrates the ability of the proposed algorithm to cope with significant shape changes, keeping good tracking capability.

6 Conclusion

This paper addresses the image processing problem of tracking moving objects with significant changes of shape or motion dynamics. The propagation of the *a posteriori* density is the most difficult step in this procedure and it is accomplished by using Gaussian mixtures whose parameters are recursively computed at each instant of time. In order to avoid a combinatorial explosion of the number of modes, mode pruning is used, according to suitable criteria. It is found that the combination of merging and mode elimination provides an adequate description of the *a posteriori* density while keeping the number of modes and consequently the computational complexity small. The proposed algorithm is applied to the estimation of object motion and shape in image sequences, assuming that sudden changes of the object shape or motion may occur during the experiment. While nonparametric methods can be used, efficiently propagating a density of a 12D space is performed, as in example 5, with advantage by parametric methods such as the ones considered in this paper.

Acknowledgement

We than the anonymous reviewers for helpful comments which improved the manuscript.

Appendix 1 - Computation of $\mathbf{c_K}$

Appendix 1 derives a recursive algorithm for updating the mixing coefficients of a Gaussian mixture for the propagation of the state density function, associated with a switched multiple model system. Since $c_{K_t} = P(K_t/Y_t)$, then

$$c_{K_t} = \frac{P(Y_t, K_t)}{P(Y_t)} = \frac{1}{P(Y_t)} \int P(Y_t, K_t, x_t) dx_t \tag{26}$$

$$= \frac{1}{P(Y_t)} \int P(y_t/Y_{t-1}, K_t, x_t) P(Y_{t-1}, K_t, x_t) dx_t \tag{27}$$

$$= \frac{1}{P(Y_t)} \int P(y_t/k_t, x_t) P(Y_{t-1}, K_t, x_t) dx_t \tag{28}$$

$$= \frac{1}{P(Y_t)} \int P(y_t/k_t, x_t) P(k_t/Y_{t-1}, K_{t-1}, x_t) P(Y_{t-1}, K_{t-1}, x_t) dx_t \tag{29}$$

Since k_t is a Markov chain, $P(k_t/Y_{t-1}, K_{t-1}, x_t) = T_{k_{t-1},k_t}$ and

$$c_{K_t} = \frac{T_{k_{t-1},k_t}}{P(Y_t)} \int P(y_t/k_t, x_t) P(Y_{t-1}, K_{t-1}, x_t) dx_t \tag{30}$$

$$= \frac{T_{k_{t-1},k_t}}{P(Y_t)} \int P(y_t/k_t, x_t) P(x_t/Y_{t-1}, K_{t-1}) P(Y_{t-1}, K_{t-1}) dx_t \tag{31}$$

$$= \alpha T_{k_{t-1},k_t} c_{K_{t-1}} \int P(y_t/k_t, x_t) P(x_t/Y_{t-1}, K_{t-1}) dx_t \tag{32}$$

where $\alpha = P(Y_{t-1})/P(Y_t)$ is the same for all the components. The integral which remains to be computed is well known: it is the innovation density function $N(0, P_\varepsilon)$, $P_\varepsilon = C_{k_t} P^- C_{k_t}^T + R_{k_t}$, computed at $\varepsilon = y_t - C_{k_t} x^-$, which will be denoted as $G(y_t)$, for simplicity. Therefore,

$$c_{K_t} = \alpha T_{k_{t-1} k_t} G(y_t) c_{K_{t-1}} \tag{33}$$

as we wished.

Appendix 2 - Output Prediction

This appendix derives the mean square error prediction of the output of a switched multiple model system $\hat{y}_t^- = E\{y_t/Y_{t-1}\}$.

Since the label sequence K_t is unknown, the predicted output can be written as follows

$$\hat{y}_t^- = \sum_{K_t} c_{K_t}^- E\{y_t/K_t, Y_{t-1}\} \tag{34}$$

where $c_{K_t}^- = P(K_t/Y_{t-1})$. Since $y_t = C_{k_t}x_t + d_{k_t} + v_t$, then

$$\hat{y}_t^- = \sum_{K_t} c_{K_t}^- [C_{k_t}\hat{x}_t^- + d_{k_t}] \tag{35}$$

where $\hat{x}_t^- = E\{x_t/Y_{t-1}\}$ is updated as shown in Table I.

Since matrices C_{k_t}, d_{k_t} depend on the current label k_t only, the sum in (35) can be reorganised by associating the terms with the same C, d matrices. Therefore

$$\hat{y}_t^- = \sum_{i=1}^{M} \sum_{K_t:k_t=i} c_{K_t}^- [C_i\hat{x}_t^- + d_i] = \sum_{i=1}^{M} C_i\hat{x}_t^{i-} + d_t^{i-} \tag{36}$$

where

$$\hat{x}_t^{i-} = \sum_{K_t:k_t=i} c_{K_t}^- \hat{x}_{K_t}^- \tag{37}$$

$$d_t^{i-} = d_i \sum_{K_t:k_t=i} c_{K_t}^- \tag{38}$$

The predicted coefficients $c_{K_t}^-$ can be recursively computed as follows

$$c_{K_t}^- = P(K_t/Y_{t-1}) = P(k_t, K_{t-1}/T_{t-1}) = P(k_t/K_{t-1}, T_{t-1})P(K_{t-1}/T_{t-1}) \tag{39}$$

Since $P(k_t/K_{t-1}, T_{t-1}) = T_{k_{t-1}k_t}, c_{K_{t-1}} = P(K_{t-1}/T_{t-1})$

$$c_{K_t}^- = T_{k_{t-1}k_t} c_{K_{t-1}}, \tag{40}$$

References

[1] A. Abrantes, J. Marques, A Class of Constrained Clustering Algorithms for Object Boundary Detection, IEEE Trans. on Image Processing, 1507-1521, 1996.

[2] Y. Bar-Shalom, T. Fortmann, Tracking and Data Association, Academic Press, 1988

[3] A. Blake, M. Isard, Active Models, Springer, 1998.

[4] C. Chang, M. Athans, State Estimation for Discrete Systems with Switching Parameters, IEEE Trans on Aerospace and Electronic Systems, 14, 418-425, 1978.

[5] J. Chen, G. Stockman, K. Rao, Recovering and Tracking Pose of Curved 3D Objects from 2D images, Proc. CVPR, 233-239, 1993.

[6] T. Cootes, T. Hill, C. Taylor and J. Haslam, The use of Active Shape Models for Locating Structures in Medical Images, Image Vision and Computing, 355-366, 1994.

[7] T. Darrel, A. Pentland, Space-Time Gestures, *Computer Vision and Pattern Recognition*, 335-340, 1993.

[8] E. Dickmanns, V. Graefe, Dynamic Monocular Machine Vision, Machine Vision and Applications, 1, 223-240, 1988.

[9] J. Evans, R. Evans, Image-Enhanced Multiple Model Tracking, Automatica, 1769-1786, 1999.

[10] N. Friedland, A. Rosenfeld, An Integrated Approach to 2D Object Recognition, Pattern Recognition, vol. 30, No. 3, 525-535, 1997.

[11] T. Heap and D. Hogg, Wormholes in shape space: tracking through discontinuous changes in shape, Sixth International Conference on Computer Vision, 34 4-9, Bombay, India, 1998.

[12] M. Isard, A. Blake, A Mixed-State Condensation Tracker with Automatic Model Switching, Int. Conference on Computer Vision, 1998.

[13] H. Lee, J. Kim, An HMM Threshold Model Approach for Gesture Recognition, Transaction Pattern Analysis and Machine Intelligence, 21, 961-973, October 1999

[14] J. Leitão, J. Moura, Nonlinear Phase Estimators Based on the Kullback Distance, *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing*, pp. 521-524, Adelaide, 1994.

[15] C. Morimoto, Y. Yacoob, L. Davis, Recognition of head Gesture using Hidden Markov Models, Proc. ICPR, 461-465, 1996

[16] A. S. Morse, Supervisory control of families of linear set-point controllers – Part 1: Exact matching, *IEEE Trans. Automat. Contr.*, **41**, 1413-1431, 1996.

[17] H. Murase, S. Nayar, Visual Learning and Recognition of 3-D Objects from Appearance, Int. Jornal of Computer Vision, 14, pp. 5-24, 1995

[18] V. Petridis, A. Kehagias, A multi-model algorithm for parameter estimation of time-varying nonlinear systems, Automatica, 34(4),469-475, 1998.

[19] B. Ripley, Pattern Recognition and Neural Networks, Cambridge University Press, 1996.

[20] H. Tagare, Deformable 2-D Template Matching Using Orthogonal Curves, Trans. Medical Imaging, 108-117, 1997

[21] J. Tugnait, Detection and Estimation for Abruptly Changing Systems, Automatica, 18, 607-615, 1982.

[22] S. Ullman, R. Basri, Recognition by Linear Combination of Models, IEEE Trans. Pattern Analysis and Machine Intelligence, 992-1006,1991.

[23] C. Vogler, D. Metaxas, ASL Recognition Based on a Coupling Between HMMs and 3D Motion Analysis, International Conference Computer Vision, 363-369, 1998.

[24] Y. Yacoob, M. Black, Parametrized Modeling and Recognition of Activities, International Conference Computer Vision, 120-127, 1998.

For each node created at instant t, update the mean and covariance, according to the following steps:

$i) Prediction$

$$\hat{x}^- = A_{k_{t-1},k_t}\hat{x}' + d_{k_{t-1},k_t}$$

$$P^- = A_{k_{t-1},k_t} P' A^T_{k_{t-1},k_t} + Q_{k_{t-1},k_t}$$

$$c^- = T_{k_{t-1},k_t} c'$$

$ii) Filtering$

$$\hat{x} = \hat{x}^- + K(y_t - C_{k_t}\hat{x}^-)$$

$$K = P^- C^T_{k_t}(C_{k_t} P^- C^T_{k_t} + R_{k_t})^{-1}$$

$$P = (I - KC_{k_t})P^-$$

$$c = \alpha G(y_t) c^-$$

where

$$(\hat{x}, P, c) \triangleq (\hat{x}_{K_t}, P_{K_t}, c_{K_t}), \quad (\hat{x}', P', c') \triangleq (\hat{x}_{K_{t-1}}, P_{K_{t-1}}, c_{K_{t-1}}),$$

Table 1: Density propagation for a Switched Dynamic System (tree update)

	average number of modes	average divergence
elimination	39.5	0.0015
merging	44.3	0.0043
merging + elimination	9.8	0.0016

Table 2: Comparison of pruning results with the ones obtained without prunning, using 200 Gaussians.

$$\begin{tikzcd}
k_1 \arrow[r] \arrow[d] \arrow[dr] & k_2 \arrow[r] \arrow[d] \arrow[dr] & k_3 \arrow[d] & \cdots & k_{t-1} \arrow[r] \arrow[d] \arrow[dr] & k_t \arrow[d] \\
x_1 \arrow[r] \arrow[d] & x_2 \arrow[r] \arrow[d] & x_3 \arrow[d] & \cdots & x_{t-1} \arrow[r] \arrow[d] & x_t \arrow[d] \\
y_1 & y_2 & y_3 & \cdots & y_{t-1} & y_t
\end{tikzcd}$$

Figure 1: Hybrid state model

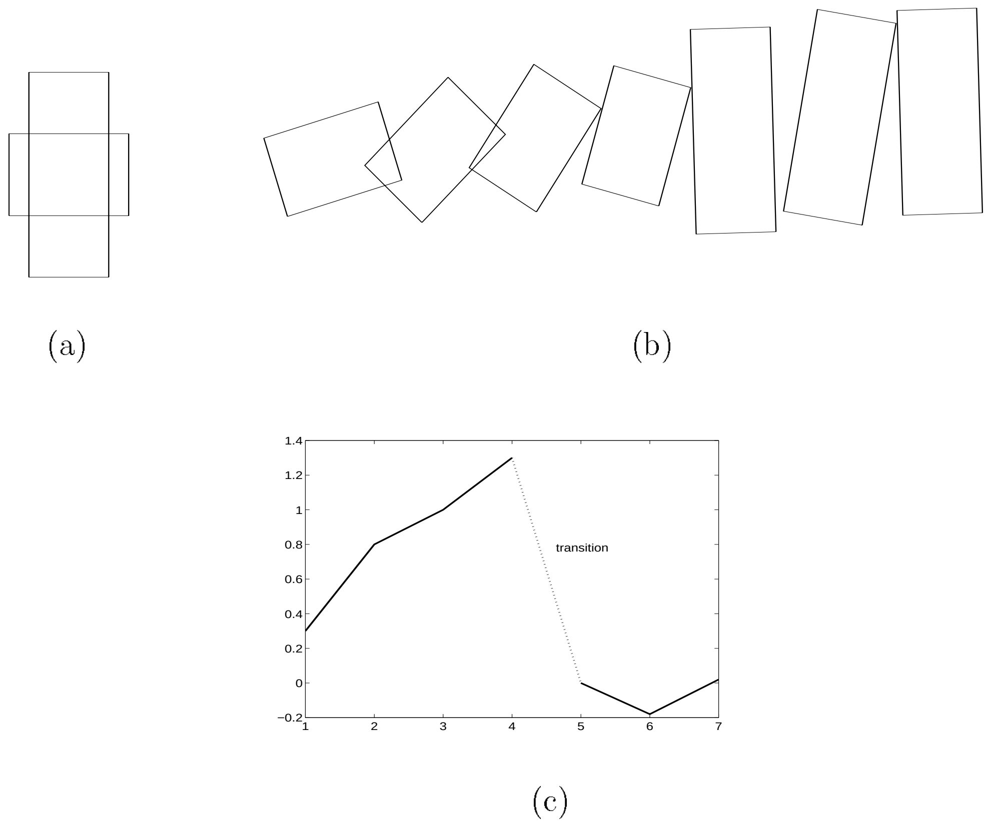


Figure 2: a) reference shapes; b) image sequence c) x_t trajectory

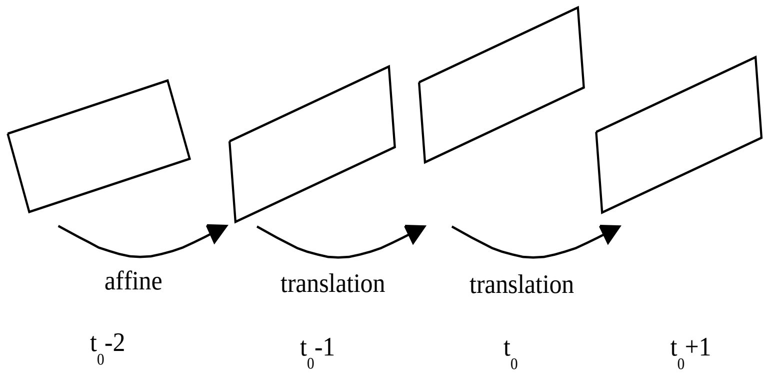


Figure 3: Computation of the mixture components

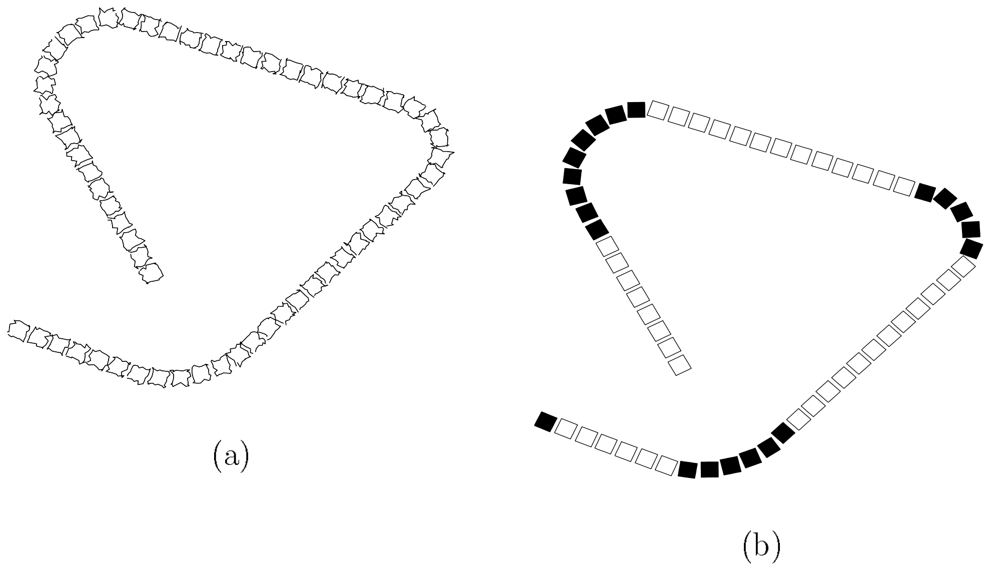


Figure 4: Tracking with two motion regimes: a) input data (noisy); b) tracking results with motion labeling

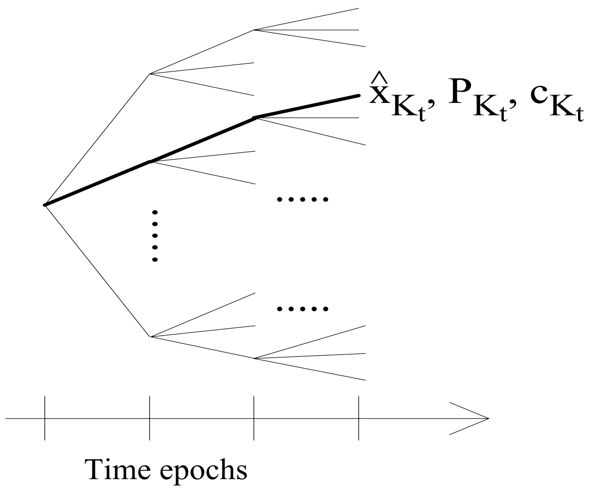
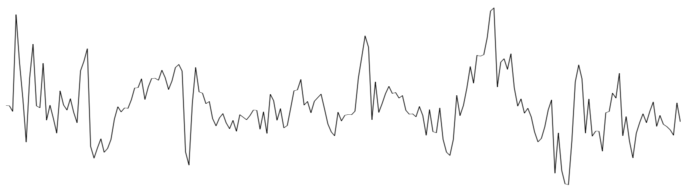
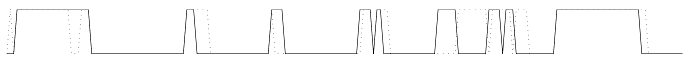


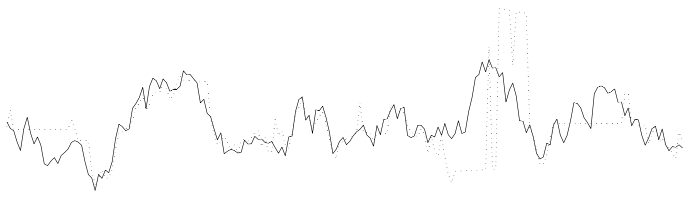
Figure 5: Computation of the mixture components

Figure 6: Moving targets

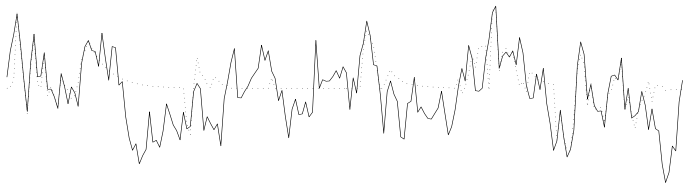


(a)





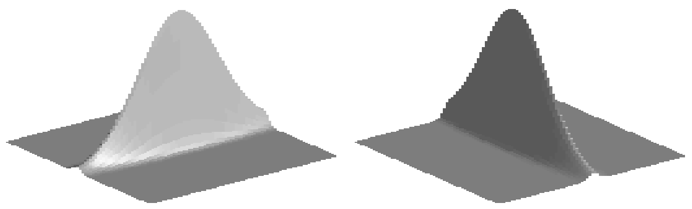
(b)



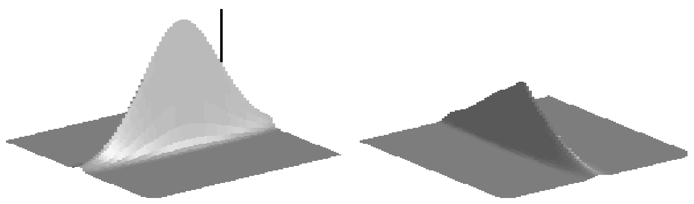
(c)

(d)

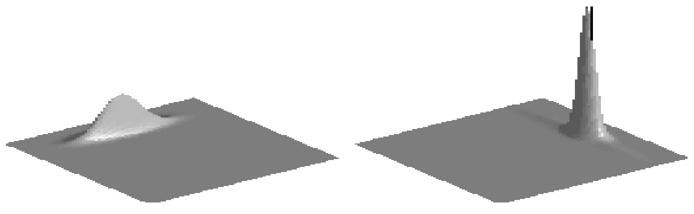
(e)



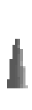
(f)

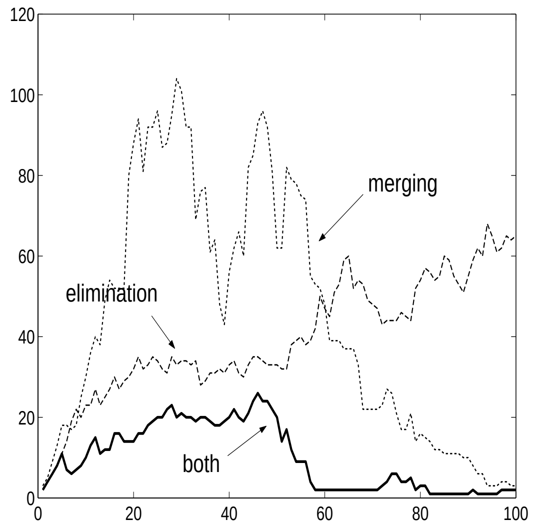


(g)

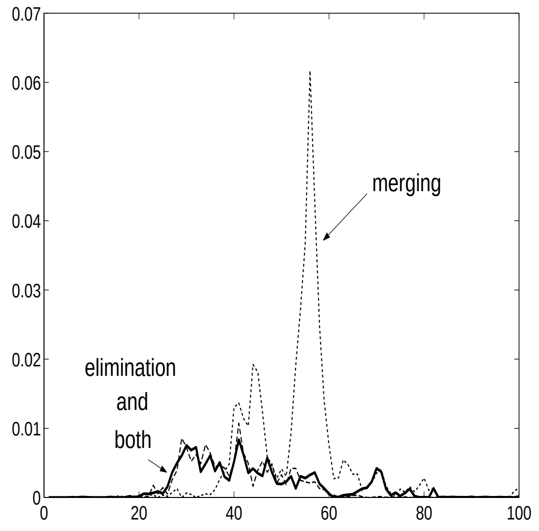


(h)



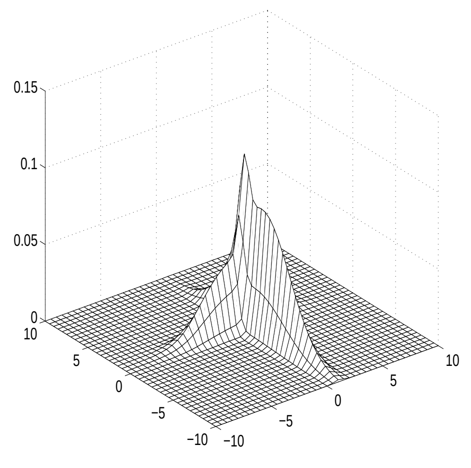


(a)

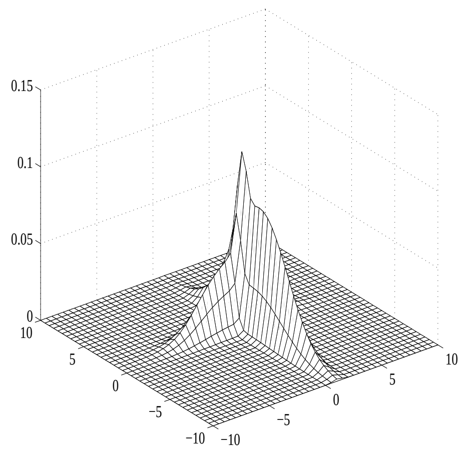


(b)

Figure 8: Pruning methods: a) number of modes; b) divergence between puned and unpruned mixtures



(a)



(b)

Figure 9: Mixtures: a) without pruning (100 Gaussians) and b) with pruning (8 Gaussians)

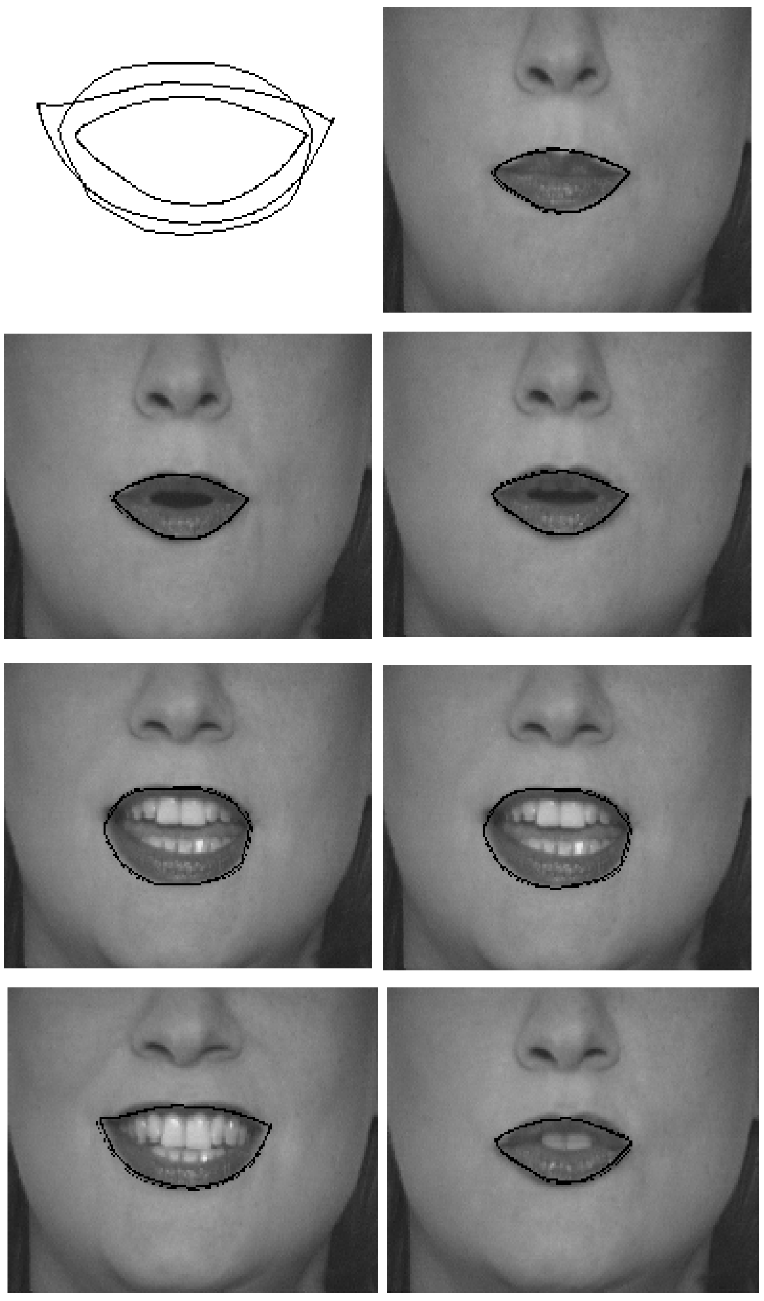


Figure 10: Lip tracking a) shape models b-h) original iamges and shape estimates

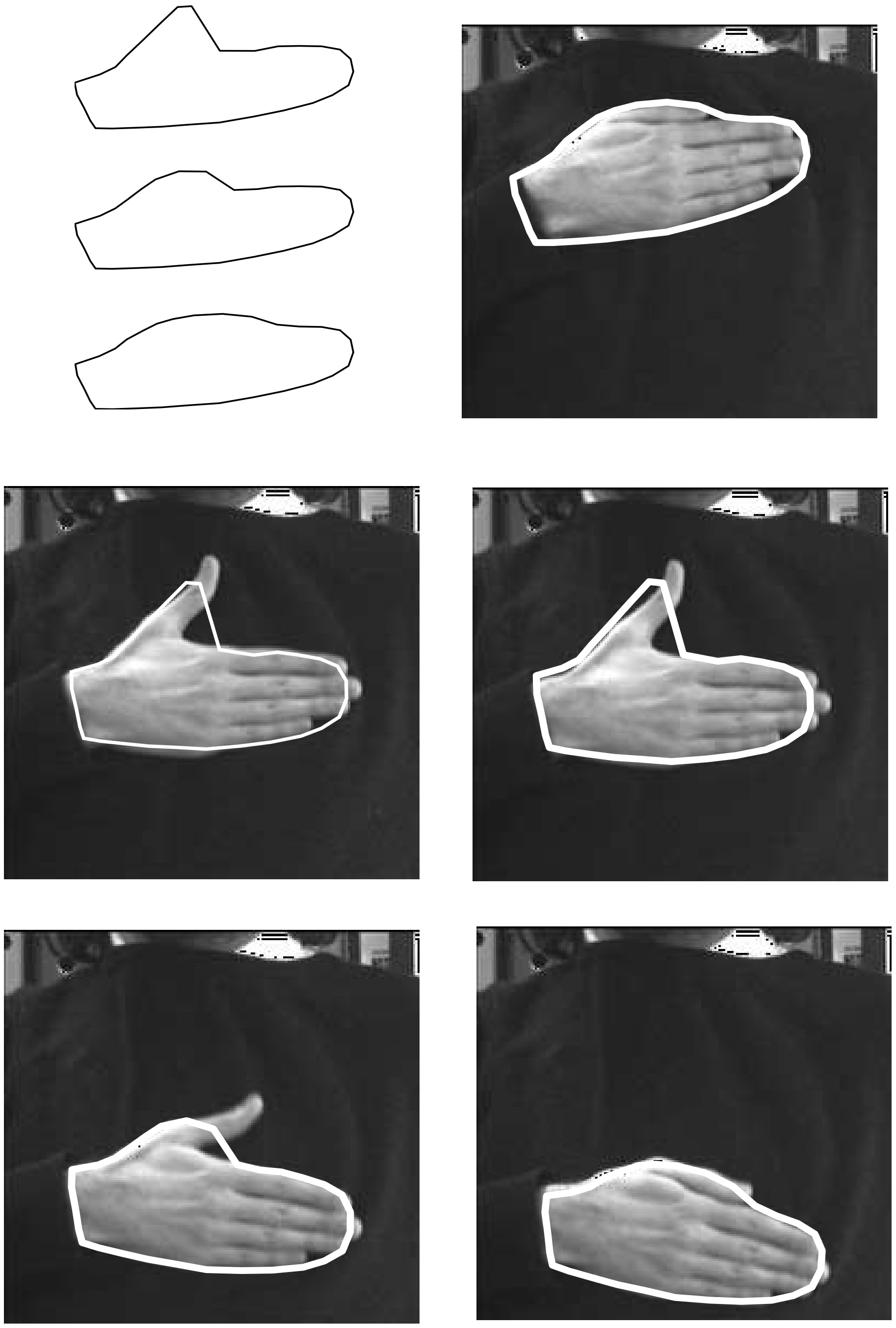


Figure 11: Tracking results with real data. a) shape models b-h) shape estimates