Learning Dynamical Models Using Expectation-Maximisation

Ben North and Andrew Blake

University of Oxford, Oxford OX1 3PJ, UK.

north@robots.ox.ac.uk, ab@robots.ox.ac.uk

Abstract

Tracking with deformable contours in a filtering framework requires a dynamical model for prediction. For any given application, tracking is improved by having an accurate model, learned from training data. We develop a method for learning dynamical models from training sequences, explicitly taking account of the fact that training data are noisy measurements and not true states. By introducing an 'augmented-state smoothing filter', we show how the technique of Expectation-Maximisation can be applied to this problem, and show that the resulting algorithm produces more robust and accurate tracking.

1 Introduction

The dynamic contour tracker of Blake *et al.* [2], developing earlier work by Kass *et al.* [8], provides a general-purpose framework for estimating over time the configuration of an image contour, for example the outline of a 3D object. A Kalman filter is used to make the tracker robust in the presence of measurement noise, and one of the key factors producing stable tracking is the dynamical model, used to predict the contour's configuration based on its recent history. While this model can be hand-specified, better results can be obtained, with less operator involvement, by using a 'training sequence' to *learn* it; a Maximum Likelihood estimate is produced based on a time-series of filtered states [2]. This approach — 'filtered learning' — works well when there is low noise, since, in such circumstances, the filtered state estimates are good compared with the true states. However, when noise is higher (for example, in the presence of image clutter or motion blur), the accuracy of the filtered estimates drops, and a dynamical model learnt from these estimates can be incorrect, as in figure 1. Here we address the problem of learning the dynamical model directly from a sequence of measurements, using Expectation-Maximisation (EM).

The EM algorithm, as introduced by Dempster *et al.* [3], is a general iterative technique for performing maximum likelihood estimation of parameters in

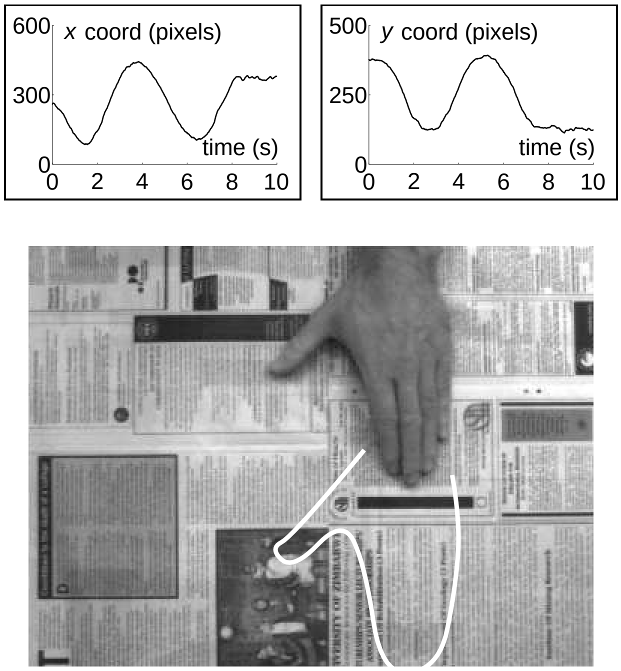

Figure 1: **A tracker trained using filtered learning is distracted by a clutter feature.** *The time-series of the estimated x- and y-coordinates are shown (top). A representative frame (bottom), taken towards the end of the sequence, shows the contour distracted by a clutter feature.*

problems which can be regarded as having 'missing data'. It has been applied in various situations, for example, learning the parameters describing a neural network for pattern recognition [1], and learning Hidden Markov Models for use in speech recognition problems [4]. It has been used for computer vision research in areas including optic flow computation [7] and the processing of noisy images [12].

The technique of EM is a natural approach to apply to our problem; in our case, the missing data are the true states describing the configuration of the contour, and the data we have are the image-based measurements. This paper develops the theory to solve our problem using EM, and shows the improvement gained by using models learnt in this way as the prediction component of a real-time dynamic contour tracker.

2 Expectation-Maximisation

Suppose a problem can be represented in terms of two measure spaces: X, a space of observable data, and Y, one of unobservable data. Suppose also that there is a parameter vector φ on which the distributions of X and Y depend. The aim is to find that φ which maximises $L(x, \varphi) = \log[p(x \mid \varphi)]$ for a given set x of observed data. For example, x might be a set of samples from a Gaussian mixture; here φ would consist of the means and covariances together with the *a priori* mixture probabilities. In general, finding this φ is not possible analytically, and so some approximating algorithm must be used instead. The Expectation-Maximisation algorithm produces a series of estimates for φ, each one producing a greater value for L. The procedure can then be run to convergence of φ. Dempster [3] shows that, given an estimate φ^0 for φ, a better estimate φ^1 can be produced by choosing it to maximise

$$Q(\varphi^0, \varphi^1) = \mathcal{E}[\log[p(x, y \mid \varphi^1)] \mid x, \varphi^0],$$

where $\mathcal{E}$ is the expectation operator over the distribution for y. The two parts of the algorithm are therefore the *Expectation* step, in which the expectation of L is found, with respect to a distribution based on the old value for the parameters, and the *Maximisation* step, in which $\mathcal{E}[L]$ is maximised with respect to a new value for the parameters.

3 System Identification from a Measurement Sequence

The problem of learning a dynamical model given a time-sequence of measurements of the system is now addressed. Compared to the approach in [2] (of treating the filtered estimates as true state values), this is a more realistic formulation of the underlying problem of learning a dynamical system for the tracker and, more generally, the problem of system identification for a Kalman filter. In terms of the 'observed' and 'hidden' variables of the EM algorithm, the measurements are the observed data, and the actual states at each time-step are the hidden data.

Ljung [9] mentions one approach to the problem of system identification from a measurement sequence. However, while the method suggested is perfectly general, it is also computationally very expensive in this case, involving the inversion of large (order of the length of the training sequence multiplied by the state-vector size, typically a few thousand) matrices, and so a more specific solution is developed.

The approach used will be as follows. Each step in the EM algorithm involves maximising $\mathcal{E}[L]$ for the log-likelihood L. Therefore, the problem of maximising L is first solved, and then the problem of finding $\mathcal{E}[L]$.

3.1 Maximum Likelihood Estimation of Parameters

For the purposes of prediction within a tracking framework, a second-order system is a suitable choice — the model allows the resulting system to display oscillatory behaviour, which is often an appropriate representation of the real motion. Here, the model is of the form $x_{n+2} - \bar{x} = A_0(x_n - \bar{x}) + A_1(x_{n+1} - \bar{x}) + B_0 w_n$. The system parameters are therefore the mean $\bar{x}$, the deterministic parameters A_0 and A_1, and the noise component B_0.

For a sequence of states $x_1, x_2, \ldots, x_{N_0}$, the problem of finding the maximum likelihood second-order model in the case $\bar{x} = 0$ is solved by Blake *et al.* in [2] as follows. The log-likelihood is given by

$$L = -\frac{1}{2}\mathrm{tr}(ZC^{-1}) + \frac{1}{2}(N_0 - 2)\log|C^{-1}|,$$

where

$$\begin{aligned} C &= B_0 B_0^t; \\ Z &= S_{22} + A_1 S_{11} A_1^t + A_0 S_{00} A_0^t - S_{21} A_1^t - S_{20} A_0^t \\ &\quad + A_1 S_{10} A_0^t - A_1 S_{12} - A_0 S_{02} + A_0 S_{01} A_1^t, \end{aligned}$$

and the second-order moments S_{ij} are given by

$$S_{ij} = \sum_{n=1}^{N_0-2} x_{n+i} x_{n+j}^t.$$

The deterministic parameters are then the solutions of the simultaneous equations

$$S_{20} - A_0 S_{00} - A_1 S_{10} = 0; \quad S_{21} - A_0 S_{01} - A_1 S_{11} = 0,$$

and $C = B_0 B_0^t$ is given by

$$C = \frac{1}{N_0 - 2}\Big(S_{22} + A_1 S_{11} A_1^t + A_0 S_{00} A_0^t - S_{21} A_1^t - S_{20} A_0^t + A_1 S_{10} A_0^t - A_1 S_{12} - A_0 S_{02} + A_0 S_{01} A_1^t\Big).$$

This result is extended to allow the learning of $\bar{x}$ by Reynard *et al.* [10]; the details are not repeated here. From the point of view of applying EM, the important point is that the log-likelihood L is linear in the moments S_{ij} and $S_i = \sum_{n=1}^{N-2} x_{n+i}$.

This is the Maximisation step; to perform the Expectation step, note that the log-likelihood L depends on the hidden variables $x_1, x_2, \ldots, x_{N_0}$ only through its linear dependence on the moments S_i and S_{ij}. Therefore, finding the individual $\mathcal{E}[S_i]$ and $\mathcal{E}[S_{ij}]$ will provide the expected value of L.

3.2 Augmented-state Smoothing Filter

The procedure adopted for finding the expected values of the auto-correlation coefficients S_i and S_{ij} is an adaptation of 'fixed-interval smoothing' [11] — the problem of estimating states x_n given a complete set $\{z_1, z_2, \ldots, z_{N_0}\}$ of measurements, where $n \leq N_0$. Consider an augmented state variable $X_n = (x_n \quad x_{n+1} \quad x_{n+2})^t$; then $S = \sum_n X_n X_n^t$ consists of all the moments matrices of the original state sequence x_n:

$$S = \begin{pmatrix} S_{00} & S_{01} & S_{02} \\ S_{10} & S_{11} & S_{12} \\ S_{20} & S_{21} & S_{22} \end{pmatrix}.$$

Also, $\mathcal{E}[S] = \sum_n \mathcal{E}[X_n]\mathcal{E}[X_n]^t + \sum_n \mathrm{Var}[X_n]$. Therefore, finding the expected values and the variances of the X_n allows us to find $\mathcal{E}[S_{ij}]$, and finding $\mathcal{E}[S_i]$ is simpler: $\sum_n \mathcal{E}[X_n]$ comprises the three $\mathcal{E}[S_i]$.

3.2.1 Dynamic and Measurement Models

Consider, then, the dynamic and measurement models for the new state vectors X_n. The second-order dynamical model, $x_{n+2} - \bar{x} = A_0(x_n - \bar{x}) + A_1(x_{n+1} - \bar{x}) + B_0 w_n$, becomes $X_{n+1} - \bar{X} = A(X_n - \bar{X}) + BW_n$, where

$$A = \begin{pmatrix} 0 & I & 0 \\ 0 & 0 & I \\ 0 & A_0 & A_1 \end{pmatrix}; \quad B = \begin{pmatrix} 0 \\ 0 \\ B_0 \end{pmatrix}; \quad W_n = w_{n+1}$$

for $n = 1, 2, \ldots, N$, where $N = N_0 - 2$, and $\bar{X} = (\bar{x} \quad \bar{x} \quad \bar{x})^t$.

The measurement model, $z_n = H_0 x_n + v_n$, where v_n are zero-mean uncorrelated Gaussian noise terms, with variance Σ_0, similarly becomes — on setting $Z_n = z_{n+2}$, $H = (0 \quad 0 \quad H_0)$ and $V_n = v_{n+2}$ — a measurement model of the form $Z_n = HX_n + V_n$ with the noise terms V_n uncorrelated and with variance $\Sigma = \Sigma_0$.

3.2.2 Smoothing Algorithm

The next problem is how to find the values of $\mathcal{E}[X_n]$ and $\mathrm{Var}[X_n]$. Assuming that the *a priori* distribution for the initial state of the system is Gaussian, all distributions remain Gaussian as time proceeds, and the solution for the mean and variances of the system states $X_1, X_2, \ldots, X_N$ given the complete set $\{Z_1, Z_2, \ldots, Z_N\}$ of measurements is developed by, for example, Sage and Melsa [11] or Jazwinski [6]. Their notation is adapted slightly as follows. Let

$$\begin{aligned} \hat{X}_k^l &= \mathcal{E}[X_k \mid Z_1, Z_2, \ldots, Z_l]; \\ P_k^l &= \mathrm{Var}[X_k \mid Z_1, Z_2, \ldots, Z_l]. \end{aligned}$$

The solution to the smoothing problem is then given by the recurrence relations

$$\begin{aligned} \hat{X}_k^N &= \hat{X}_k^k + F_k(\hat{X}_{k+1}^N - A\hat{X}_k^k); \\ P_k^N &= P_k^k + F_k(P_{k+1}^N - P_{k+1}^k)F_k^t, \end{aligned}$$

where $F_k = P_k^k A^t (P_{k+1}^k)^{-1}$.

Here the terms $\hat{X}_k^k$, P_{k+1}^k and P_k^k are known from the solution of the filtering problem; the recurrence relations are solved using $\hat{X}_N^N$ and P_N^N (also known from the filtered estimates) as boundary conditions.

The smoothing algorithm here relies on the availability of the solution to the filtering problem — the estimates $\hat{X}_k^k$ and the variances P_k^k and P_{k+1}^k; these are obtained by running the Kalman filter on the measurement sequence, given the *a priori* distribution of the first state in the sequence.

3.3 Summary of Algorithm

The steps of the EM algorithm for this case are therefore as follows.

(1) Initialise with the starting distribution of the state X_1 (mean and variance), and an initial estimate for the dynamics $(\bar{x}, A_0, A_1, B_0)$;

(2) Run the Kalman filter on the measurement set to produce the filtered estimates $\hat{X}_k^k$;

(3) Run the smoothing algorithm on the measurement set (using the filtered estimates) to produce the smoothed estimates $\hat{X}_k^N$;

(4) Using the results of the smoothing, find the expected values $\mathcal{E}[S_i]$ and $\mathcal{E}[S_{ij}]$ of the moments;

(5) From the expected values of the moments, estimate the system parameters $(\bar{x}, A_0, A_1, B_0)$;

(6) Using the system derived in the previous step, go back to step (2).

The algorithm should be run, as is usual for EM applications, until convergence of the parameters of interest, namely $(\bar{x}, A_0, A_1, B_0)$.

4 Dynamic Contour Tracking

We wish to apply the algorithm derived in section 3 to the problem of training a contour tracker. One area of application is to the problem of tracking an object moving across a cluttered background. This problem should be one in which the EM learning algorithm enjoys two (related) advantages: firstly, in a situation with poor measurements (as will be the case for tracking over a cluttered background), having a good dynamical model is important, more so than when the measurements are reliable, and secondly, experiments with synthetic data (not described here owing to lack of space) show that the differences between EM learning and filtered learning are greater when measurement noise is higher.

4.1 Example Application: Hand Tracking

As an example of tracking an object in clutter, we consider the problem of tracking a hand moving across news-sheet. This background was chosen as being representative of the type of clutter which would be expected in the general problem of hand-tracking across a noisy background, say, a cluttered desktop. A tracker was therefore set up with a contour matching the outline of the hand. The space of allowing deformations of the hand contour was restricted to the two-dimensional space of translations in the x- and y-directions.

4.1.1 Estimating the Measurement Noise

The first issue to deal with is that of determining the variance Σ of the measurement process. The measurement process in the tracker consists of casting normals at points around the predicted position of the contour and searching for features along these normals. One feature is chosen (on the grounds of being the strongest, for example), and the distance u_i along the normal between this chosen feature and the contour is used as one component of the innovation. The complete innovation vector is built up of the u_i from all measurement points. The noise of each measurement is assumed to be Gaussian, with all errors independent. The variance of the measurement vector is therefore $\sigma^2 I$, and it is σ^2 we estimate, as follows.

Consider the situation where the contour is positioned exactly over the outline of the hand. In this case, if there were no noise, every u_i would be zero. In the presence of noise, the sample variance of the u_i can then be used as an estimate of the measurement process variance σ^2. The process of calculating the sample variance of u_i was performed on a representative still, the contour having 27 measurement normals, with the result that the measurement process variance was estimated as $62\,\text{pels}^2$. This value was used for the experiments which follow.

4.1.2 Gathering a Training Sequence

To investigate the behaviour of the EM-based training algorithm in the case of hand-tracking, a training sequence was gathered. To gather a training sequence of reasonable length (ten seconds — 500 fields — in this example), some initial 'bootstrap' training was required, since the untrained dynamics were incapable of following the hand's motion for more than two or three seconds. ('Bootstrapping' consists of iteratively learning a dynamical model, using an estimated model to filter the measurements, and then learning the next estimate of the dynamics from the filtered state estimates.)

For this sequence, the hand was moved in a rough ellipse filling most of the field of view of the camera, in an anti-clockwise direction. The motion was tracked using dynamics learnt from a shorter sequence, as described above. The sequence thus gathered was used in both learning algorithms.

4.1.3 Learning the Dynamical Model

Both learning algorithms were run using the data gathered — we now summarise the models learnt using each method. The mean position of the hand as learnt by each method is as follows. For filtered learning, $\bar{x}_F = (307, 282)$, and for EM learning, $\bar{x}_{EM} = (305, 286)$, which is similar.

The deterministic parts of the models are as follows. Both systems exhibit one dominant oscillatory mode, with the others having large damping constants (order $20\,\text{s}^{-1}$). This dominant mode had damping constant β and frequency ω as follows for the two systems:

$$\beta_F = 0.01\,\text{s}^{-1}; \quad \omega_F = 0.75\,\text{rad}\,\text{s}^{-1};$$
$$\beta_{EM} = 0.02\,\text{s}^{-1}; \quad \omega_{EM} = 0.68\,\text{rad}\,\text{s}^{-1}.$$

These are similar systems, with the one learnt by EM learning being slightly more damped.

The stochastic parts of the two models, in terms of the eigenvalues σ_1^2 and σ_2^2 of the noise covariance $B_0 B_0^t$, are as follows (all in pels^2). For the system learnt with filtered learning, $\sigma_1^2 = 1.30$ and $\sigma_2^2 = 0.20$, whereas for the model learnt using EM learning, $\sigma_1^2 = 0.33$ and $\sigma_2^2 = 0.22$.

We observe that filtered learning has produced a model with significantly higher process noise. The deterministic parts are very similar. With greater process noise, we expect the tracker to rely more on the measurements, with two (related) results: firstly, we expect the estimates produced to be noisier, and secondly, we expect that the tracker may be more easily distracted by clutter, possibly to the extent of losing lock. Below we describe the performance of the two trackers on test sequences.

4.1.4 Tracking Results

To test the first prediction, that the tracked estimate will be noisier for the system produced by filtered learning, each tracker was run on a sequence in which the hand was held entirely still. The tracker was initialised by hand, and then run for five seconds. Ideally, the resulting estimate of the hand's position would therefore be constant over the sequence, but image noise will cause some jitter — this is unavoidable.

The variances of test sequences gathered in this way are as follows. (σ_x^2 is the variance of the estimated x-coordinate and σ_y^2 that of the estimated y-coordinate.) For the dynamics learnt with filtered learning, $\sigma_x^2 = 0.44$ and $\sigma_y^2 = 1.35$, and for the EM-learnt dynamics, $\sigma_x^2 = 0.11$ and $\sigma_y^2 = 0.45$ (all in pels^2). In both cases, the s.d. of the noise in the y-coordinate direction is larger by a factor of approximately 2. This is almost certainly caused by interlace — the camera produces images with double the resolution in the x-coordinate direction. The variances for the EM-trained tracker are 3–4 times lower than those for the filtered-trained tracker, although all are small (s.d. of order a pel). The difference may be significant in, for example, a 'mousing' application, where precise control of a pointer is required.

We now wish to test the trackers on longer test sequences in which the hand executes motion of broadly the same type (elliptical motion in an anti-clockwise direction) as in the training sequence. Two test sequences were recorded, and both trackers used on them. The first ten-second sequence was recorded of the hand performing two complete anti-clockwise rough ellipses — this test sequence therefore contained motion of a more agile nature than that in the training sequence, which consisted of just over one cycle. A tracker using each dynamical model was then run on this sequence. The only difference between the parameters controlling the behaviour of the trackers was the dynamical model. Each tracker was initialised by hand, then run on the sequence.

The performance of the tracker trained using filtered learning is indicated in figure 1. For all of the first ellipse and half of the second, it tracks satisfactorily, but then the presence of clutter features causes it to lose lock — it then remains positioned over this feature for the rest of the sequence. The corresponding results for the tracker trained using EM learning are presented in figure 2. Although there is some noise in the estimates, as is to be expected for a sequence with noisy measurements, lock is maintained throughout — tracking is successful.

A second ten-second test sequence was recorded, this time of the hand describing most of an ellipse, more slowly than in the training sequence. (Neither of the test sequences, therefore, contained motion exactly the same as the training sequence.) The results are not shown here, but are similar to those for the previous test sequence — as before, the tracker trained using filtered learning is distracted by a clutter feature, whereas that trained with EM learning tracks the whole sequence successfully.

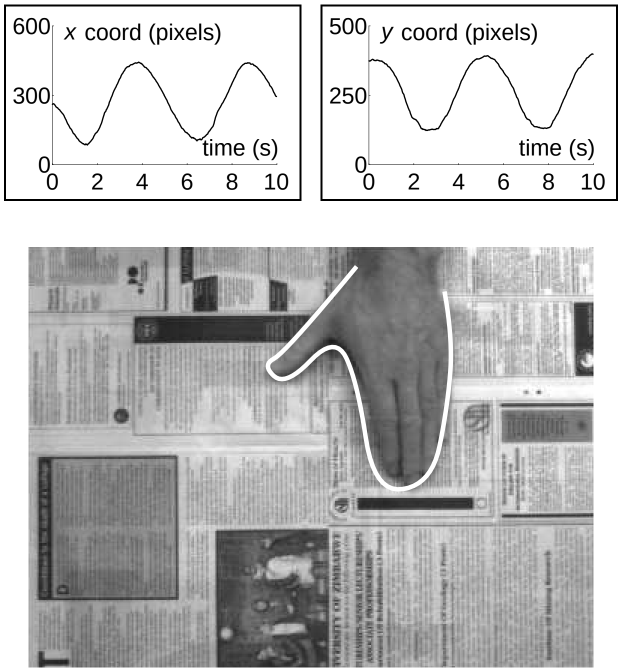

Figure 2: **Successful tracking using EM learning.** *The same sequence as used to produce figure 1 is tracked using dynamics learnt with EM learning. The estimated coordinates are shown (top), and a frame from the sequence, taken at the same time as the corresponding still in figure 1 (bottom), shows that the tracked estimate is good.*

5 Modelling a Beating Heart

Consider the problem of learning a dynamical model for the data shown in figure 3 — a time sequence of the expansion component of a beating heart tracked from an ultrasound image sequence [5]. (In this case, visual inspection of the tracking reveals that the estimates are reasonable, and therefore the approximation $x_n = \hat{x}_n$ will be taken.) Learning a standard second-order model of the form $x_{n+2} - \bar{x} = a_0(x_n - \bar{x}) + a_1(x_{n+1} - \bar{x}) + bw_n$ gives a system whose predictive behaviour is shown in figure 4(b). The learnt motion is far too damped when compared with real data (a sample of which is given in figure 4(a)), with a damping constant of $\beta = 4.6\,\mathrm{s}^{-1}$; the predicted frequency is, however, close to that of the training data.

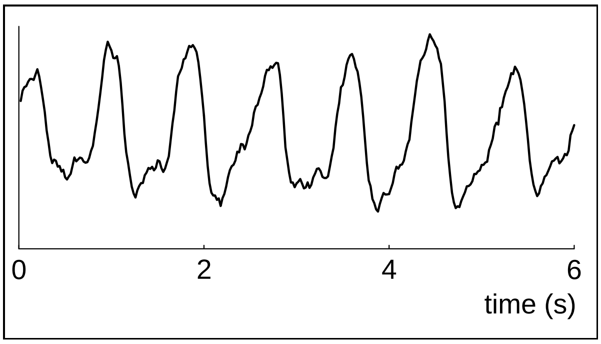

Figure 3: **Time-series of tracked heart data.** *A beating heart is tracked, and the component of the PCA basis representing expansion of the heart is plotted here against time. (Data courtesy of Gary Jacob.)*

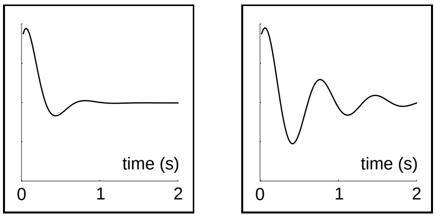

Figure 4: **Predictive behaviour of learnt models.** *Without any driving noise, the predictive behaviour of the learnt models can be seen. The effect of the larger damping constant of the filtered-learnt model can clearly be seen (left). Although the extended model learnt by EM (right) is not perfect, it captures the longer-scale behaviour of the data better.*

Therefore, an extended model of the form

$$\begin{aligned} x_{n+2} - \bar{x} &= a_0(x_n - \bar{x}) + a_1(x_{n+1} - \bar{x}) + b w_n; \\ y_n &= x_n + b_1 u_n, \end{aligned}$$

where the state is now given by the y_n, is considered. The additional stationary noise (the u_n) allows greater flexibility in learning a more appropriate model. This model is equivalent to a measurement process of the type considered in section 3 with $h = 1$ and $\sigma = b_1$. The same solution applies, therefore, and the predictive behaviour of the model learnt is shown in figure 4(c). The damping constant learnt is $\beta = 1.7\,\mathrm{s}^{-1}$, giving a much better predictive model.

Comparing the two models shows that the one including additive stationary noise, and learnt using the EM result presented here, gives much better predictive performance — this will result in a greater ability to track over temporary measurement failure.

6 Conclusions

We have successfully applied EM to the estimation problem of system identification, from a sequence of measurements, in a Kalman filter framework. Specifically, we have applied the method to learning a dynamical model for a dynamic contour tracker. Experiments show that a system learnt using this EM-based method is superior to one learnt using existing methods.

This work suggests other applications of EM to problems related to learning models for contour-tracking. For example, the space of allowable deformations of the contour is presently learnt using PCA independently of the dynamical model. A promising area for investigation is the possibility of learning this component of a contour tracker directly from the same sequence as that used to learn the dynamical model.

We thank Professor Brian Ripley for helpful discussion during the development of this work, and acknowledge financial assistance from the EPSRC.

References

[1] C.M. Bishop. *Neural networks for pattern recognition.* Oxford, 1995.

[2] A. Blake, M.A. Isard, and D. Reynard. Learning to track the visual motion of contours. *Artificial Intelligence*, 78:101–134, 1995.

[3] A. Dempster, M. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. R. Stat. Soc.*, B(39):1–38, 1977.

[4] X.D. Huang, Y. Arika, and M.A. Jack. *Hidden Markov Models for Speech Recognition.* Edinburgh University Press, 1990.

[5] Gary Jacob, J. Alison Noble, and Andrew Blake. Robust contour tracking in echocardiographic sequences. In *Proc. 6th Int. Conf. on Computer Vision*, 1998.

[6] A.H. Jazwinski. *Stochastic processes and filtering theory.* Academic Press, 1970.

[7] A. Jepson and M.J. Black. Mixture models for optical flow computation. *Proc. Conf. Computer Vision and Pattern Recognition*, pages 760–761, 1993.

[8] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. In *Proc. 1st Int. Conf. on Computer Vision*, pages 259–268, 1987.

[9] L. Ljung. *System identification: theory for the user.* Prentice-Hall, 1987.

[10] D. Reynard, A.P. Wildenberg, A. Blake, and J. Marchant. Learning dynamics of complex motions from image sequences. In *Proc. 4th European Conf. Computer Vision*, pages 357–368, Cambridge, England, Apr 1996.

[11] Sage and Melsa. *Estimation theory with applications to communications and control.* McGraw-Hill, 1971.

[12] M. Tanaka and T. Katayama. Edge detection and restoration of noisy images by the expectation-maximization algorithm. *Signal Processing*, 17:213–226, 1989.