Learning to track curves in motion

Proc. IEEE Int. Conf. Decision Theory and Control, 1994, 3788–3793.

Andrew Blake, Michael Isard and David Reynard
Department of Engineering Science,
University of Oxford,
Oxford OX1 3PJ, UK.

ABSTRACT

Recent developments in video-tracking allow the outlines of moving, natural objects in a video-camera input stream to be tracked live, at full video-rate. The system used here is based on Kalman Filtering with a B-spline representation of curves to track the silhouettes of moving non-polyhedral objects. For example hands, lips, legs, vehicles, fruit can be tracked at video-rate without any special hardware beyond a desktop workstation and a video-camera and framestore.

The novel contribution of this paper is a tracking algorithm that uses a bootstrapping technique to learn *a stochastic, dynamic model for given motions from example video-streams. Incorporating such a model into the tracking algorithm greatly enhances maximum tracking speed and robustness to distraction from background objects. Experiments with learning both rigid and non-rigid motions, using moving hands and lips, clearly show the increased tracking power resulting from the learned dynamics.*

1 Introduction

Real-time video tracking has been previously achieved for line-drawings [11] and polyhedral structures [14] and for simple, natural features such as road-edges [8]. Commercial systems (Watsmart, Oxford Metrics) are available which track artificial markers on live video. Natural point features (e.g. on a face) but not curves, have been tracked at 10Hz using a workstation assisted by image-processing hardware [3]. Curve trackers [13] have been demonstrated on modest workstations but slower than frame-rate. The approach presented in this paper allows agile tracking of curves in live video data, at 50Hz, on a modest workstation (SUN IPX plus Cohu video-camera and Datacell S2200 framestore) without additional hardware. The possibility of tracking curves within a Kalman filtering framework was raised by Szeliski and Terzopoulos [16]. A Kalman Filter framework [10] with a B-spline state-space is used here, with preprogrammed dynamics, as described in [6], to form a "default" tracker.

The new development reported in this paper is the use of a training algorithm to replace the *default* dynamics embedded in the tracking filter by *learned* dynamics. The learned dynamical models are based on stochastic differential equations [1]. and the learning algorithm is a simple application of stochastic system identification [2] using maximum likelihood estimation.

Training proceeds as follows. Example motions are tracked by a general-purpose tracker based on the assumption of programmed, default object dynamics. The tracked motion is then used as a training set for the new algorithm which estimates the underlying dynamics of the training motion. The learned dynamics are then incorporated into a new version of the tracker which then enjoys enhanced tracking capability for motions similar to those in the training set. The learning process can be repeated to bootstrap trackers of successively increasing performance. The effectiveness of the learning algorithm in generating agile trackers which are resistant to distraction from background clutter has been demonstrated.

2 Tracking framework

The tracker consists of an estimator for a piecewise smooth image-plane curve in motion

$$\mathbf{r}(s,t) = (x(s,t), y(s,t)).$$

Following the tracking formulations of others [15, 7], the curve representation is in terms of B-splines. Quadratic splines are used, with the possibility of multiple knots for vertices. The tracking framework described here follows an earlier framework [6] but is generalised to allow any learned motion model to be used in the prediction phase of tracking.

2.1 Curve representation

Curves are represented as parametric B-splines with N spans and N_c control points.

$$x(s,t) = \mathbf{B}(s)\mathbf{X}(t) \text{ and } y(s,t) = \mathbf{B}(s)\mathbf{Y}(t), \ 0 \le s \le N$$

where $\mathbf{X} = (X_1, .., X_{N_c})^T$ and similarly for $\mathbf{Y}$. The elements of $\mathbf{X}$ and $\mathbf{Y}$ are simply the x, y coordinates of the set of control points (X_m, Y_m) for the B-spline. The number of control points is equal to the number of spans — $N_c = N$ — for closed curves and $N_c = N + d$ for open ones (with, in each case, appropriate variations where multiple knots are used to vary curve continuity). The vector $B(s)$ consists of blending coefficients defined by

$$\mathbf{B}(s) = (B_1(s), ..., B_M(s))$$

where B_m is a B-spline basis function [9, 5] appropriate to the order of the curve and its set of knots.

A state vector $\mathbf{Q}$ is defined:

$$\begin{pmatrix} \mathbf{X} \\ \mathbf{Y} \end{pmatrix} = W\mathbf{Q} + \begin{pmatrix} \overline{\mathbf{X}} \\ \overline{\mathbf{Y}} \end{pmatrix}$$

to generate the space of allowed deformations of the curve, relative to a "hand-drawn" template $(\overline{\mathbf{X}}, \overline{\mathbf{Y}})$, and assumes rigidity, with the possible addition of a few degrees of freedom for limited non-rigid motion — details are given in [6]. The dimension of the state-space N_Q is typically between 4 and 10. In the case that the object is planar or nearly so and the field of view is small, the $\mathbf{Q}$-space is an "affine" subspace generated by the template itself, and defined by a W-matrix of the following form:

$$W = \begin{pmatrix} \mathbf{1} & \mathbf{0} & \overline{\mathbf{X}} & \mathbf{0} & \mathbf{0} & \overline{\mathbf{Y}} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} & \overline{\mathbf{Y}} & \overline{\mathbf{X}} & \mathbf{0} \end{pmatrix}, \qquad (1)$$

so that $N_Q = 6$, and where the N_c-vectors $\mathbf{0}$ and $\mathbf{1}$ are defined by:

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

2.2 Discrete-time motion model

It is natural to model the motion of an object in continuous time; discrete time enters only when the measurement process is considered. In the case of video imagery, measurements are made synchronously at a sample interval Δ. In between sampling epochs, continuous equations of motion can be integrated [10], to give $\mathcal{X}_{n+1} \equiv \mathcal{X}((n+1)\Delta)$ in terms of $\mathcal{X}_n \equiv \mathcal{X}(n\Delta)$. The resulting discrete model has the form

$$\mathcal{X}_{n+1} - \overline{\mathcal{X}} = A(\mathcal{X}_n - \overline{\mathcal{X}}) + \begin{pmatrix} \mathbf{0} \\ B\mathbf{w}_n \end{pmatrix}. \qquad (2)$$

The matrix coefficient A is a $2N_Q \times 2N_Q$ matrix, defining the deterministic part of the dynamics.

Without loss of generality, A and $\overline{\mathcal{X}}$ can be standardised to the forms

$$A = \begin{pmatrix} \mathbf{0} & \mathbf{I} \\ A_0 & A_1 \end{pmatrix} \quad \text{and} \quad \overline{\mathcal{X}} = \begin{pmatrix} \overline{\mathbf{Q}} \\ \overline{\mathbf{Q}} \end{pmatrix}. \qquad (3)$$

Then the equations of motion are simplified to the standard form:

$$\mathbf{Q}_{n+2} = A_0\mathbf{Q}_n + A_1\mathbf{Q}_{n+1} + (I - A_0 - A_1)\overline{\mathbf{Q}} + B\mathbf{w}_n. \quad (4)$$

As for the stochastic component of the discrete dynamics, at each time n, the $\mathbf{w}_n$ are independent vectors of independent unit normal random variables. This noise process is then transformed by the matrix B which couples the driving noise into the various natural modes of the deterministic dynamics. In fact B is not completely observable, only the covariance $C = BB^T$ can be computed; this is because the probability distribution for $\mathbf{w}$, being a vector of i.i.d. normal variables, is (it is easily shown) invariant to orthogonal transformations of $\mathbf{w}$.

2.3 Measurement model

In an earlier framework [6], measurement was represented as a continuous time process, not because this is a realistic model — after all, video is a synchronous sampled data stream — but because it is tractable in the sense of facilitating analysis of the tracker's performance as a control system. Here however, the learned motion models are too varied to allow the kind of performance analysis that was possible for constant velocity models. Therefore, in this paper, a synchronous, discrete-time measurement model is used throughout.

Given an estimator (see next section) $\hat{\mathbf{r}}(s,t)$ for the contour $\mathbf{r}(s,t)$, the visual measurement process at time t consists of casting rays along normals $\hat{\mathbf{n}}(s,t)$ to the estimated curve and, simultaneously at certain points s along the curve, measuring the innovation $\nu(s,t)$ of a feature (typically a high contrast edge) along the ray, so that

$$\nu(s,t) = [\mathbf{r}(s,t) - \hat{\mathbf{r}}(s,t)].\hat{\mathbf{n}} + v(s,t) \qquad (5)$$

where $v(s,t)$ is a scalar noise variable, assumed gaussian, with a variance σ^2 that will be taken to be constant, both spatially and temporally. Defining measurement to be along the normal *only* is essential as displacement tangential to the curve is unobservable, the well-known "aperture problem" of visual motion [12].

For incorporation into a tracking filter, we need to express ν in terms of the state vector $\mathcal{X}$, by means of a measurement matrix $H(s)$:

$$\nu(s,t) = H(s)(\hat{\mathcal{X}} - \mathcal{X}) + v(s,t) \qquad (6)$$

where the measurement matrix $H(s)$ is:

$$H(s) = \begin{pmatrix} \hat{\mathbf{n}}(s)^T \otimes B(s) & \mathbf{0} \end{pmatrix} W, \qquad (7)$$

in which $\otimes$ denotes the "Kronecker product" [4] of two matrices.

2.4 Tracker

The system model and measurement model are now combined in a standard way [10] to form a Kalman filter for visual curve tracking. In order to collect a training set, some form of un-trained tracker is required. Since the dynamics of the moving object are unknown at this stage it is necessary to use a tracker with reasonable default dynamics, based on constant velocity with homogeneous and isotropic plant noise [6].

3 System identification

The learning task is now to estimate the coefficients A_0, A_1, B from a training sequence $\mathbf{Q}_1, ..\mathbf{Q}_m$, gathered at the image sampling frequency of $1/\Delta = 50$Hz. As mentioned earlier, it is impossible in principle to estimate B uniquely, but the covariance coefficient $C = BB^T$ can be determined, from which a standard form for B can be computed as $B = \sqrt{C}$, applying the square root operation for a square matrix [4].

For simplicity of notation we assume that the mean $\overline{\mathbf{Q}}$ has been subtracted off:

$$\mathbf{Q} \rightarrow \mathbf{Q} - \overline{\mathbf{Q}}.$$

The log-likelihood function for the multivariate normal distribution is then, up to a constant:

$$\begin{aligned} L(\mathbf{Q}_1, ..\mathbf{Q}_n | A_0, A_1, B) = & \qquad (8) \\ -\frac{1}{2}\sum_{n=1}^{m-2} \left| B^{-1}\left(\mathbf{Q}_{n+2} - A_0\mathbf{Q}_n - A_1\mathbf{Q}_{n+1}\right)\right|^2 \\ -(m-2)\log\det B. \end{aligned}$$

Now the problem is to estimate A_0, A_1 and $C = BB^T$ by maximising the log-likelihood L. Maximising first with respect to A_0, A_1, it will be shown that separability holds — maxima with respect to A_0, A_1 turn out to be independent of the value of C. Equivalently to maximising L we minimise

$$\sum_{n=1}^{m-2} \left| B^{-1}\left(\mathbf{Q}_{n+2} - A_0\mathbf{Q}_n - A_1\mathbf{Q}_{n+1}\right)\right|^2$$

with respect to A_0, A_1. Now this function can be expanded as

$$f(A_0, A_1) = \mathrm{tr}(ZC^{-1})$$

where

$$\begin{aligned} Z = & S_{22} + A_1 S_{11} A_1^T + A_0 S_{00} A_0^T \\ & -2S_{21}A_1^T - 2S_{20}A_0^T + 2A_1 S_{10} A_0^T \end{aligned}$$

and

$$S_{ij} = \sum_{n=1}^{m-2} \mathbf{Q}_{n+i}\mathbf{Q}_{n+j}^T, \quad i,j = 0,1,2, \qquad (9)$$

the second-order moment matrices for the multivariate time-sequence $\mathbf{Q}_n$, $n = 1, .., m$. Now, completing the square in Z, with respect to both A_0 and A_1, we can rewrite Z as

$$Z(A_0, A_1) = Z' + Z_0$$

where Z_0 is a constant matrix and

$$\begin{aligned} Z' = & (A_1 - \hat{A}_1) S_{11} (A_1 - \hat{A}_1)^T \\ & +(A_0 - \hat{A}_0) S_{00} (A_0 - \hat{A}_0)^T \\ & +2(A_1 - \hat{A}_1) S_{10} (A_0 - \hat{A}_0)^T \end{aligned}$$

and where $\hat{A}_0, \hat{A}_1$ are the solutions of the simultaneous equations

$$S_{20} - \hat{A}_0 S_{00} - \hat{A}_1 S_{10} = 0 \qquad (10)$$

$$S_{21} - \hat{A}_0 S_{01} - \hat{A}_1 S_{11} = 0. \qquad (11)$$

Assuming the minimum of f exists (easily shown to be the case), f must be convex, so the variable term, the quadratic form $\mathrm{tr}(ZC^{-1})$, must be positive definite, achieving its minimum of 0 when

$$A_0 = \hat{A}_0 \text{ and } A_1 = \hat{A}_1.$$

These conditions are independent of the value of C — the separability condition as required.

Having obtained estimators for A_0 and A_1, now C can be estimated. Rewriting (8) as

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

fixing $A_0 = \hat{A}_0$, $A_1 = \hat{A}_1$, and extremising with respect to C^{-1} (using the identity $\partial(\det M)/\partial M \equiv (\det M)M^{-T}$) we obtain

$$\hat{C} = \frac{1}{m-2} Z(\hat{A}_0, \hat{A}_1), \qquad (12)$$

which can be computed efficiently using the moments S_{ij}.

4 Learning rigid motion

4.1 Learning in configuration space

Three training sets are used here, each involving one component of rigid body motion, namely, "zoom", "rotation" and "flap". After training, each of the three trackers follows motions similar to the ones in the respective training sets, but will not follow the other two rigid body motions. This is illustrated in figure 1.

4.2 Learning in phase-space

Configuration-space training, as demonstrated above only captures the static component of the object-model. The next experiment demonstrates the power of incorporating learned dynamics into a curve tracker. A training set of vertical, oscillatory, rigid motion has been generated and used to learn motion coefficients A, B. Testing of the trained tracker, incorporating the learned motion, is done against an un-trained, default tracker. The test sequences consist of rapid, vertical, oscillatory motions of a hand. The sequences are stored on video so that fair comparisons can be made, using the standard sequences, of the performance of different trackers. The trained tracker (figure 2) follows the motion successfully, right up to a rate of around 3 cycles per second. The untrained trackers cannot achieve this. Snapshots in figure 2 which show that when the measurement process fails due to excessive lag in the tracker, it is the learned dynamics that effectively bridge the hiatus and allow lock subsequently to be recovered. Such trackers can even perform against a significantly cluttered background, as figure 3 shows.

5 Conclusions

A new learning algorithm has been described for live tracking of moving objects from video. It supplies particular dynamics, modelled by a stochastic differential equation, to be used predictively in a contour tracker. The process is bootstrapped by a default tracker which assumes constant velocity rigid motion

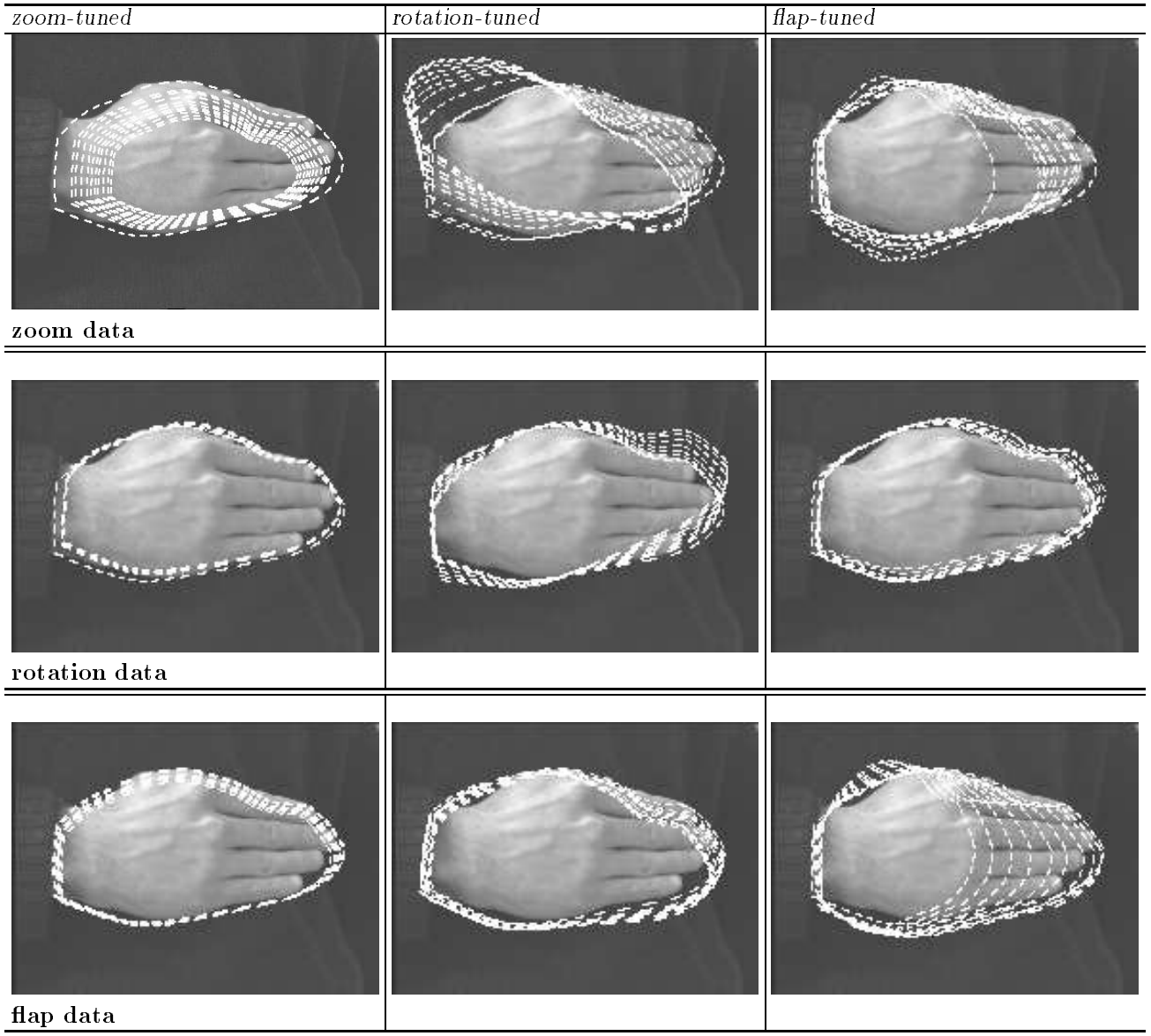

Figure 1: **Filtering single components of affine deformation** *Three training sets have been used here to generate filters sensitive specifically to zoom, rotation and flapping. Each filter is tested on the zoom, rotation and flapping training sequences to illustrate the specificity of training. Accurate tracking is shown in images along the diagonal in which each filter is run on its own training sequence. Off the diagonal, when a tuned tracker is applied to an inappropriate test motion sequence, tracking fails.*

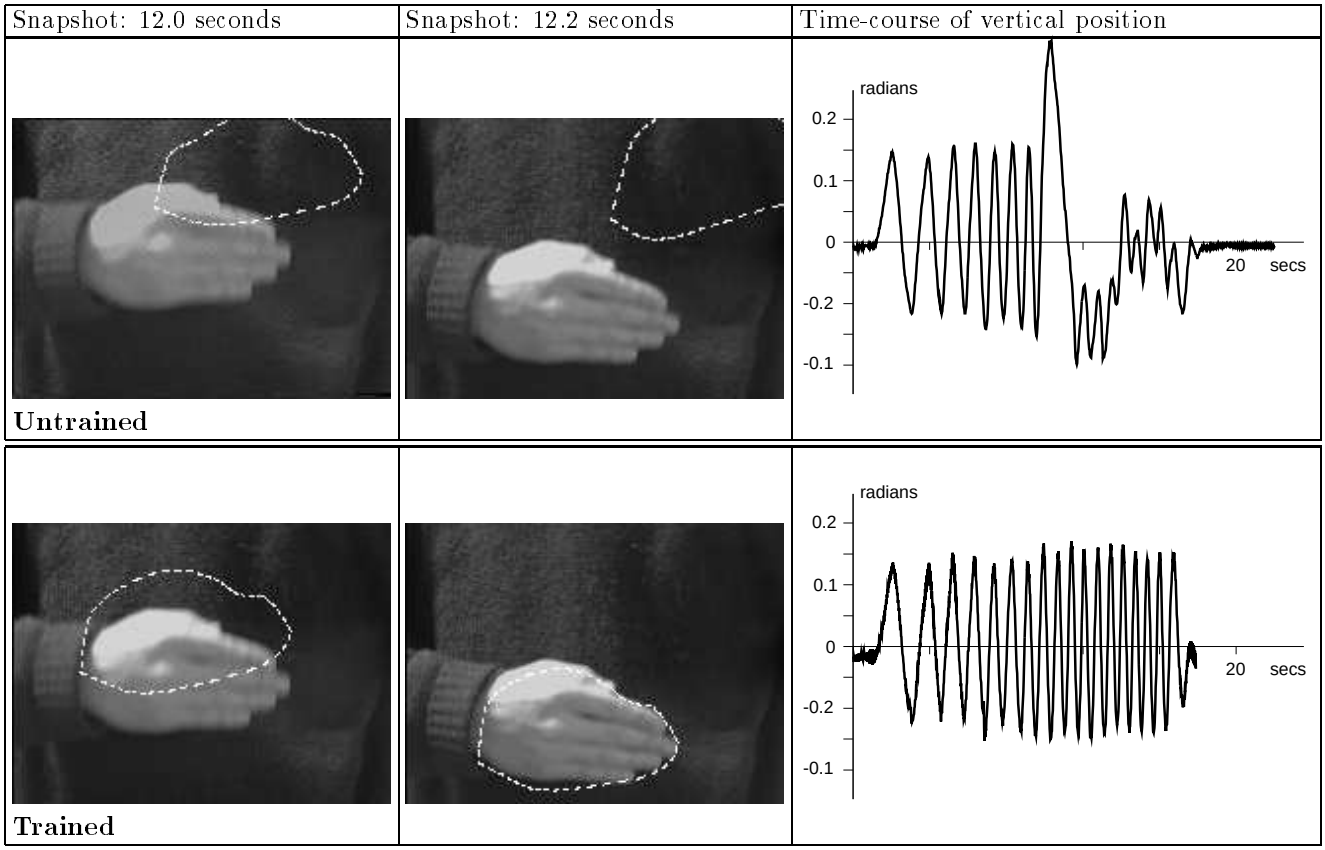

Figure 2: **Trained tracker for rigid motion, tested with rapid oscillations**. *A "chirp" test motion, consisting of vertical oscillations of progressively increasing frequency. For the untrained tracker, lock is lost after about 12 seconds and is unrecoverable another 0.2 second later. The trained tracker is still tracking after 12 seconds, though it is lagging sufficiently (more than 40 pixels, the size of the measurement window) to have lost lock. The learnt model takes over tracking temporarily, in the absence of measurements, and by 12.2 seconds lock has been recovered.*

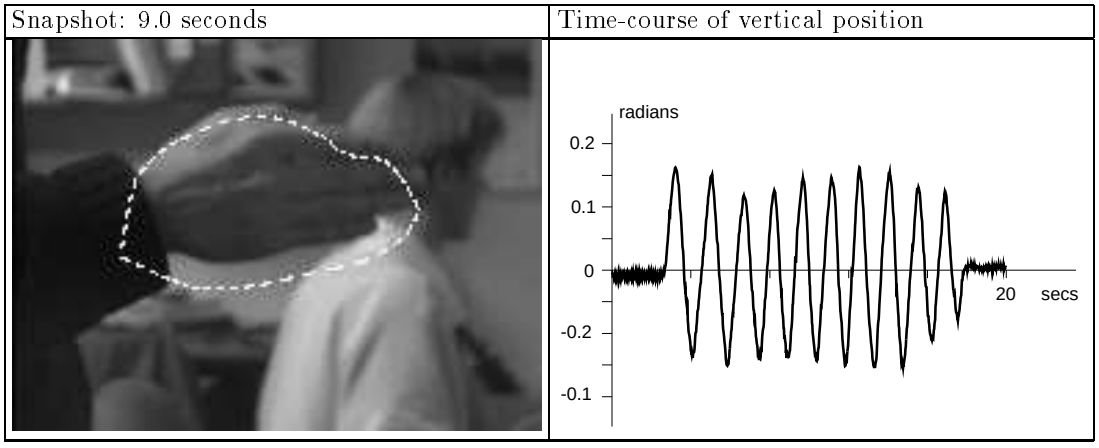

Figure 3: **Trained tracker for oscillatory rigid motion, tested against clutter**.

driven randomly. It is crucial that the constraints of rigid-body motion are incorporated — represented in our algorithm by the **Q**-space. This is what allows stable tracking, for which free parameters must be limited, to be combined with the apparently conflicting requirement that a large number of control points are needed for accurate shape representation. Rather than using arbitrarily chosen dynamics in the tracker they are acquired by the learning algorithm that allows dynamical models to be built from examples. When such a model is incorporated into a tracker, agility and robustness to clutter are considerably increased. In tests with non-rigid motion (figure 4) the learning algorithm has proved, so far, to be essential to obtaining any satisfactory tracking performance at all.

Figure 4: **Trained trackers for nonrigid motion.** *Learning dynamics has proved essential for obtaining any acceptable performance with lip tracking.*

Acknowledgements

We are grateful for the use of elegant software constructed by Rupert Curwen, Nicola Ferrier, Simon Rowe, Henrik Klagges and for discussions with Roger Brockett, Yael Moses, Brian Ripley, Richard Szeliski, Andrew Zisserman. We acknowledge the support of the SERC, the EC Esprit programme and the Newton Institute, Cambridge.

[1] K. J. Astrom. *Introduction to stochastic control theory.* Academic Press, 1970.

[2] K. J. Astrom and B. Wittenmark. *Adaptive control.* Addison Wesley, 1989.

[3] A. Azarbayejani, T. Starner, B. Horowitz, and A. Pentland. Visually controlled graphics. *IEEE Trans. Pattern Analysis and Machine Intell.*, 15(6):602–604, 1993.

[4] Stephen Barnett. *Matrices: Methods and Applications.* Oxford University Press, 1990.

[5] R.H. Bartels, J.C. Beatty, and B.A. Barsky. *An Introduction to Splines for use in Computer Graphics and Geometric Modeling.* Morgan Kaufmann, 1987.

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

[7] R. Cipolla and A. Blake. The dynamic analysis of apparent contours. In *Proc. 3rd Int. Conf. on Computer Vision*, pages 616–625, 1990.

[8] E.D. Dickmanns and V. Graefe. Applications of dynamic monocular machine vision. *Machine Vision and Applications*, 1:241–261, 1988.

[9] I.D. Faux and M.J. Pratt. *Computational Geometry for Design and Manufacture.* Ellis-Horwood, 1979.

[10] Arthur Gelb, editor. *Applied Optimal Estimation.* MIT Press, Cambridge, MA, 1974.

[11] C. Harris. Tracking with rigid models. In A. Blake and A. Yuille, editors, *Active Vision*, pages 59–74. MIT, 1992.

[12] B.K.P. Horn. *Robot Vision.* McGraw-Hill, NY, 1986.

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

[14] D.G. Lowe. Robust model-based motion tracking through the integration of search and estimation. *Int. Journal of Computer Vision*, 8(2):113–122, 1992.

[15] S. Menet, P. Saint-Marc, and G. Medioni. B-snakes: implementation and application to stereo. In *Proceedings DARPA*, pages 720–726, 1990.

[16] R. Szeliski and D. Terzopoulos. Physically-based and probabilistic modeling for computer vision. In B. C. Vemuri, editor, *Proc. SPIE 1570, Geometric Methods in Computer Vision*, pages 140–152, San Diego, CA, July 1991. Society of Photo-Optical Instrumentation Engineers.