Part II

Probabilistic Modelling

Chapter 8

Probabilistic models of shape

The purpose of this second part of the book is to put Active Contours into a probabilistic setting. As chapter 2 claimed, the probabilistic framework is essential for dealing with *classes* of shapes and motions. It is valuable even with deformable templates, in static problems, to describe classes of shapes. Then probabilistic modelling is extended to dynamic problems, to mesh with the powerful Kalman filtering formalism, in which cumulative temporal uncertainty about shape is counterbalanced by the inflow of measurements from an image sequence.

This chapter concentrates on the application of probabilistic models to static problems. The ideas discussed so far about fitting curves by regularisation are to be re-interpreted probabilistically. The deterministic approach of chapter 6 aimed to generate a unique estimate $\hat{\mathbf{X}}$ of curve shape from data $\mathbf{r}_f(s)$, moderated via regularisation towards a template $\overline{\mathbf{X}}$. Now, in a more general probabilistic setting, $\hat{\mathbf{X}}$ is merely one property, typically the mean or mode, of an entire probability distribution. The solution to the fitting problem is therefore no longer just a single value, but a whole family of possible curves. In that case, the variance of the distribution is a measure of how accurate the fitted curve is and can be used to generate a range of plausible fitted curves. One application for this is to sweep out the search region for image-processing operations. Another application, developed in later chapters on tracking, is to achieve a fusion of shape information accumulated over time and the latest visual measurements.

The distribution for curve shape $\mathbf{X}$ obtained from probabilistic fitting is expressed as a *posterior density* $p(\mathbf{X}|\mathbf{r}_f)$. This is the conditional probability density for the curve shape $\mathbf{X}$ *given* the observed data $\mathbf{r}_f(s)$. According to Bayes' formula for densities (see appendix A.3) the posterior density can be obtained as a product of a *prior density* $p_0(\mathbf{X})$ and an *observation density* $p(\mathbf{r}_f|\mathbf{X})$:

$$p(\mathbf{X}|\mathbf{r}_f) \propto p(\mathbf{r}_f|\mathbf{X})p_0(\mathbf{X}). \tag{8.1}$$

Note that although $p(\mathbf{r}_f|\mathbf{X})$ is a probability density over $\mathbf{r}_f$, in this formula $\mathbf{r}_f$ is considered fixed and it is the variation of $p(\mathbf{r}_f|\mathbf{X})$ with $\mathbf{X}$ — the "likelihood" of $\mathbf{X}$ — that is of interest. The prior density is the probabilistic mechanism for regularisation. For example, the norm-squared regulariser $\alpha\|\mathbf{X}-\overline{\mathbf{X}}\|^2$ used in chapter 6 becomes a Gaussian density function $p(\mathbf{X})$ whose mean is the curve $\overline{\mathbf{X}}$. The other term $\|\mathbf{r}-\mathbf{r}_f\|^2$ in the regularisation problem, conveying the influence of the data, also becomes a Gaussian density whose value is high when the hypothesised curve $\mathbf{r}$ fits the data closely. Then Bayes' rule simply combines the competing influences of the prior and the observations into a single density, also Gaussian.

Note that Bayesian principles of image interpretation reach far beyond Gaussian modelling. However, the Gaussian is the simplest case and has attractive properties that facilitate efficient computation. Consideration of more general distributions is left until much later in the book, in chapter 12.

8.1 Probability distributions over curves

The Gaussian prior probability density for curves in shape-space $\mathcal{S}$ consistent with the general quadratic regulariser $(\mathbf{X}-\overline{\mathbf{X}})^T\overline{S}(\mathbf{X}-\overline{\mathbf{X}})$ is:

$$p_0(\mathbf{X}) \propto \exp -\frac{1}{2}(\mathbf{X}-\overline{\mathbf{X}})^T\overline{S}(\mathbf{X}-\overline{\mathbf{X}}), \tag{8.2}$$

where the matrix $\overline{S}$ is the "information" matrix introduced in chapter 6. Note that probability density decreases as the curve $\mathbf{X}$ deviates further from the mean $\overline{\mathbf{X}}$ of the distribution, as expected. Probabilistically, $\overline{S}$ has a particular interpretation: its inverse $P_0 = \overline{S}^{-1}$, if it exists, is a *covariance matrix* (appendix A.3), a multi-dimensional measure of the variability of curve shape across a distribution. It can be used to compute the positional variability of a given point $\mathbf{r}(s)$ on a curve as a 2 × 2 covariance matrix

$$P_{\mathbf{r}}(s) = U(s)WP_0W^TU(s)^T, \tag{8.3}$$

in a two-dimensional Gaussian distribution $\mathcal{N}(\overline{\mathbf{r}}(s), P_{\mathbf{r}}(s))$. In general, this distribution can be depicted as an ellipse whose axes are eigenvectors of $P_{\mathbf{r}}(s)$ and whose semi-axes, representing positional variances, are the eigenvalues. The spatial variance $P_{\mathbf{r}}(s)$ is represented by an uncertainty ellipse giving a mean-square displacement

$$\rho_0^2(s) = \mathrm{tr}(P_{\mathbf{r}}(s)) \tag{8.4}$$

where $\mathrm{tr}(\cdot)$ denotes the trace of a matrix. The mean-square displacement along the entire curve can then be computed easily (the proof follows below) as

$$\overline{\rho}_0^2 = \mathrm{tr}(P_0\mathcal{H}). \tag{8.5}$$

An isotropic prior, uniform in s, with mean-square displacement along the curve $\overline{\rho}_0^2$ is projected onto shape-space $\mathcal{S}$ as

$$p_0(\mathbf{X}) \propto \exp -\frac{N_X}{2\overline{\rho}_0^2}\|\mathbf{X} - \overline{\mathbf{X}}\|^2.$$

This can be seen by considering its information matrix in shape-space, which is

$$\overline{S} = \frac{N_X}{\overline{\rho}_0^2}\mathcal{H}\,. \tag{8.6}$$

From (8.5) it is apparent that the mean-square displacement along the curve is

$$\mathrm{tr}\left(\frac{\overline{\rho}_0^2}{N_X}\mathcal{H}^{-1}\mathcal{H}\right) = \frac{\overline{\rho}_0^2}{N_X}\mathrm{tr}(I_{N_X}) = \overline{\rho}_0^2,$$

as required.

Spline space is an interesting special case in which the covariance at a given point $\mathbf{r}(s)$ is $P_{\mathbf{r}}(s) \propto I_2$, a multiple of the identity matrix, so that positional error at each point is isotropically distributed (see below for the derivation). The average displacement of $\mathbf{r}(s)$ from its mean $\overline{\mathbf{r}}(s)$, is $\rho_0(s)$ and in fact $\rho_0(s) \approx \overline{\rho}_0$ for all s. This suggests circular confidence regions for $\mathbf{r}(s)$ (figure 8.1). (The average pointwise displacement $\rho_0(s)$ varies a little with s, relative to $\overline{\rho}_0$ — by around ±10% , for instance, on a regular, closed, quadratic B-spline.)

In chapter 6, there was some discussion of the way in which image-processing operations are applied along curve normals. In the interests of efficiency, processing is limited to a segment of each normal within a search region. The probabilistic

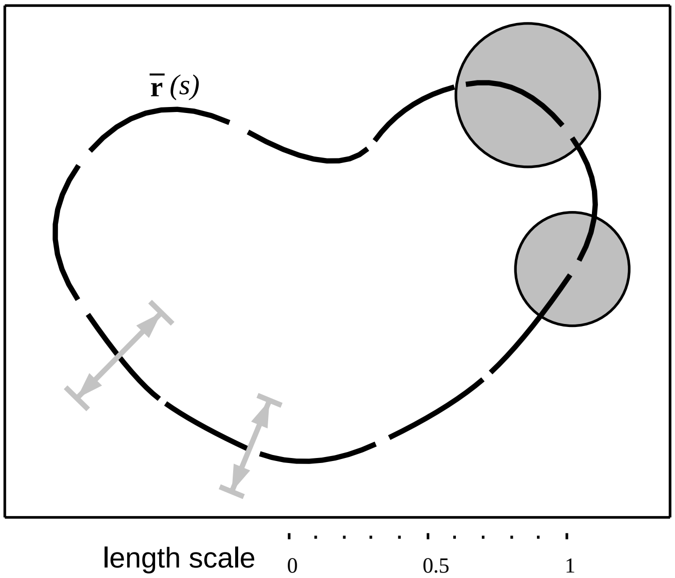


Figure 8.1: Probability distribution for curves. *An example of a Gaussian probability distribution in spline space with mean shape* $\overline{\mathbf{r}}(s)$ *and uniform, isotropic covariance* P_0*. The distribution of the position of a given point* $\mathbf{r}(s)$ *is Gaussian, with circular confidence intervals of radius* $1.73\rho_0(s)$ *(95% confidence). The normal displacement has a Gaussian distribution with standard deviation* $\rho_n(s)$ *and* $\rho_n(s) = \rho_0(s)/\sqrt{2}$ *in the special case of this example. The search segments shown (arrowed) are intervals of length* $\pm 2\rho_n(s)$ *for approximately 95% statistical confidence. (Average displacement* $\overline{\rho}_0 = 0.14$ *here.)*

framework provides a rationale, using the prior distribution, for fixing the search segment along the normal $\overline{\mathbf{n}}(s)$ at $\overline{\mathbf{r}}(s)$. The normal component of displacement of the curve is Gaussian with mean 0 and standard deviation $\rho_n(s)$ where

$$\rho_n^2(s) = \overline{\mathbf{n}}(s)^T P_{\mathbf{r}}(s)\overline{\mathbf{n}}(s).$$

or, directly in terms of shape-space covariance P_0,

$$\rho_n^2(s) = \mathbf{h}(s)^T P_0 \mathbf{h}(s) \tag{8.7}$$

where $\mathbf{h}(s)$ was defined in (6.15) on page 124 in chapter 6. In the special case of the norm-squared prior ($S \propto \mathcal{H}$) in spline space, the isotropy of $P_{\mathbf{r}}(s)$ means that $\rho_n(s) = \rho_0(s)/\sqrt{2}$.

A reasonable search segment is $\overline{\mathbf{r}}(s) \pm 2\rho_n(s)\overline{\mathbf{n}}(s)$, corresponding to a confidence interval (for the one-dimensional Gaussian distribution) at a level of approximately 95%. Such search segments are illustrated in figure 8.1. This idea is elaborated later for validation gates, taking account not only of the prior but also the observation density.

Derivation of average radius and isotropy results

The formula (8.5) for the root-mean-square displacement for the covariance ellipse is derived first. Pointwise, the mean-square displacement is $\rho_0^2(s)$ whose average value along the curve, from (8.4), is

$$\begin{aligned}
& \frac{1}{L}\int_0^L \operatorname{tr}\left(P_{\mathbf{r}}(s)\right)\,ds \\
\underset{(8.3)}{=}\ & \frac{1}{L}\int_0^L \operatorname{tr}\left(U(s)WP_0W^TU(s)^T\right)\,ds \\
=\ & \operatorname{tr}\left(P_0W^T\left[\frac{1}{L}\int_0^L U(s)^TU(s)\,ds\right]W\right) \\
\underset{(3.23)}{=}\ & \operatorname{tr}\left(P_0W^T\mathcal{U}W\right) \\
\underset{(4.7)}{=}\ & \operatorname{tr}\left(P_0\mathcal{H}\right),
\end{aligned}$$

as required.

The special case of the norm-squared prior (8.6) in spline space $\mathcal{S} = \mathcal{S}_Q$ was considered in which $W = I_{N_Q}$ and the pointwise covariance is

$$P_{\mathbf{r}}(s) = U(s)P_0U(s)^T.$$

Since $P_0 \propto \mathcal{U}^{-1}$,

$$P_{\mathbf{r}}(s) \propto (\mathbf{B}(s)^T\mathcal{B}^{-1}\mathbf{B}(s))I_2$$

— a multiple of the identity matrix as claimed.

Random sampling

A graphic illustration of a statistical family of curves can be made by sampling randomly from its distribution. Random curve sampling is a practical technique in its own right, partly as a debugging aid to check whether a particular probabilistic prior model of shape is appropriate to a given application. More importantly it forms the basis of some powerful algorithms for recognising patterns, used when the posterior density for shape given data is too complex to be represented exactly and is represented instead by a set of samples.

Given an N_X-dimensional Gaussian distribution $\mathcal{N}(\overline{\mathbf{X}}, P)$ in shape-space with mean $\overline{\mathbf{X}}$ and covariance matrix P, a random variate $\mathbf{X}$ from the distribution can be generated by taking a vector $\mathbf{w}$ of N_X independent "standard" normal variables each distributed as $\mathcal{N}(0,1)$ and transforming $\mathbf{w}$ linearly:

$$\mathbf{X} = B\mathbf{w} + \overline{\mathbf{X}} \tag{8.8}$$

where

$$B = \sqrt{P}, \tag{8.9}$$

a matrix square root with the property that $BB^T = P$ (Note that this does not uniquely specify B, but any admissible B serves equally well, and generates the desired distribution; see also appendix A.1). This technique is used to illustrate prior distributions of curves in spline space $\mathcal{S}_Q$ (figure 8.2) and in various shape-spaces (figure 8.3). As before, the distributions are given by norm-squared densities over shape-space.

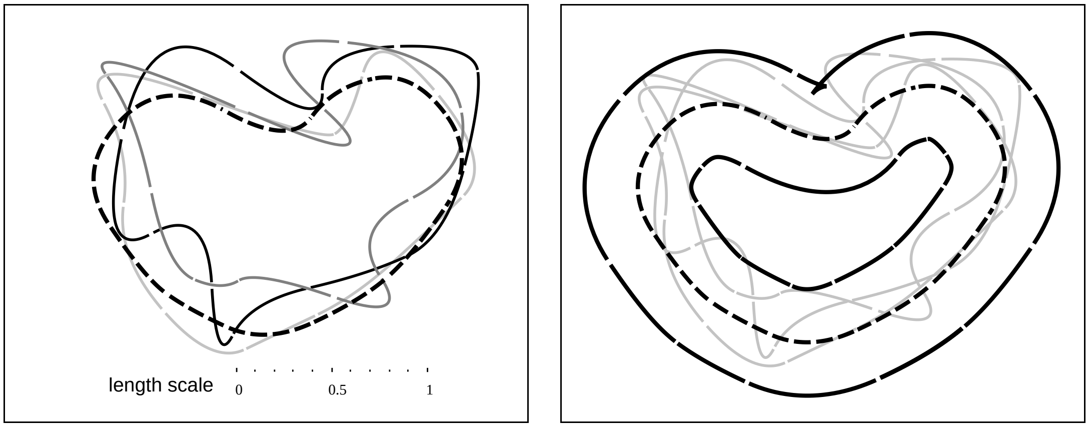


Figure 8.2: Sampling from curve families. *Samples are drawn at random (left) from a uniform, isotropic, Gaussian distribution in the spline space* $\mathcal{S}_Q$ *such that* $\|\mathbf{r}(s) - \overline{\mathbf{r}}(s)\|$ *has a root-mean-square value of 0.2 length units. (Mean shape* $\overline{\mathbf{r}}$ *is shown dashed). Parallel curves depicting confidence intervals for normal displacement (spatially averaged), at 95% significance, contain the random curves as expected (right).*

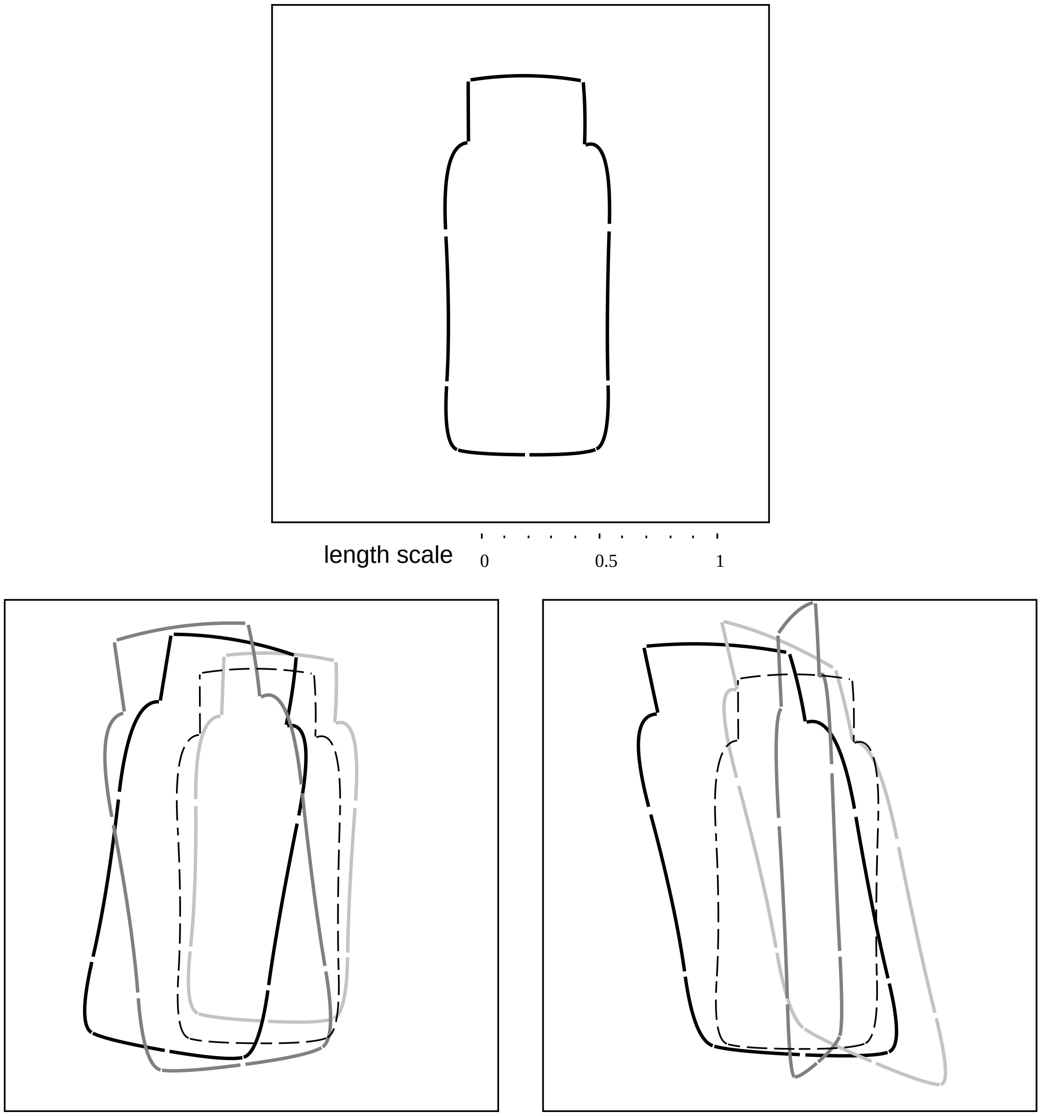


Figure 8.3: Random sampling in shape-space *The families of bottle-outlines shown have mean shape (top) and a uniform, isotropic, Gaussian distribution in shape-space with root-mean-square displacement of 0.3 length units. Euclidean similarities (left); affine space (right).*

8.2 Posterior distribution

The aim of the chapter is to express curve fitting in terms of the posterior distribution for shape $\mathbf{X}$ given observed data $\mathbf{r}_f$. Realistically, $\mathbf{r}_f$ must be obtained by processing image data, and that is addressed in the next section. For now, the posterior is illustrated with artificial data, using the random sampling method. As at the start of chapter 6, the artificial image feature is modelled as a spline curve with control-vector $\mathbf{Q}_f$. What was characterised there, in (6.3) on page 117, as fitting by regularisation, becomes the construction by Bayes' formula (8.1) of the posterior distribution $p(\mathbf{X}|\mathbf{Q}_f)$, given a Gaussian prior $p_0(\mathbf{X})$ and an observation density $p(\mathbf{Q}_f|\mathbf{X})$. The measurement term $\|\mathbf{Q}-\mathbf{Q}_f\|^2$ in the regularisation problem is interpreted as an observation density

$$p(\mathbf{Q}_f|\mathbf{X}) \propto \exp -\frac{N_X}{2\overline{\rho}_f^2}\|\mathbf{Q}-\mathbf{Q}_f\|^2 \tag{8.10}$$

where a variance constant $\overline{\rho}_f^2/N_X$ has been included here to represent explicitly the uncertainty of measurements. The distance constant $\overline{\rho}_f$ can be interpreted as the average displacement, at a given point on the fitted curve, due solely to observation error and without any influence from the prior. Finally, multiplying prior and observation densities, in accordance with Bayes' formula, the posterior distribution proves to be a Gaussian

$$\mathbf{X}|\mathbf{Q}_f \;\sim\; \mathcal{N}(\hat{\mathbf{X}}, P). \tag{8.11}$$

Its mean $\hat{\mathbf{X}}$ is precisely the solution (6.7) to the earlier regularisation problem, adjusted now to allow for the observation variance constant:

$$\hat{\mathbf{X}} = S^{-1}\left(\overline{S}\,\overline{\mathbf{X}} + \frac{N_X}{\overline{\rho}_f^2}\mathcal{H}\mathbf{X}_f\right) \tag{8.12}$$

where

$$S = \overline{S} + \frac{N_X}{\overline{\rho}_f^2}\mathcal{H}. \tag{8.13}$$

The estimate $\hat{\mathbf{X}}$ is also known as a "Maximum A Posteriori" (MAP) estimate because it is the value of $\mathbf{X}$ at which the posterior probability density $p(\mathbf{X}|\mathbf{Q}_f)$ achieves its peak value. The covariance of the posterior distribution is $P = S^{-1}$, and S, the total information matrix for the posterior, is just the sum of information in the prior and observation distributions. (The derivation of the posterior distribution is essentially a replay of the derivation on page 118 of the solution to the regularisation problem.)

The posterior distribution for the curve-fitting problem of figure 6.2 on page 119 can be illustrated by random sampling. Again, regularisation is made invariant to transformations in a subspace $\mathcal{S}_s$. As in chapter 6, this is represented probabilistically by modifying the information matrix (8.6) for a uniform, isotropic, Gaussian prior using a projection E^d outside the invariant subspace:

$$\overline{S} = \frac{N_X}{\overline{\rho}_0^2} {E^d}^T \mathcal{H} E^d. \tag{8.14}$$

This has the effect of producing a prior distribution that is invariant to Euclidean similarities so that

$$p_0(\mathbf{X} + \mathbf{X}^s) = p_0(\mathbf{X}) \;\text{ for any }\; \mathbf{X}^s \in \mathcal{S}_s.$$

The posterior distribution, illustrated by random sampling in figure 8.4, clearly shows the likely range of variability of possible solutions, something that regularisation alone could not convey. This is also reflected mathematically by the fact that there is an extra constant in the system. As before there is a regularisation constant α which is now a ratio of the coefficients of the information matrices for the observation and for the prior:

$$\alpha = \overline{\rho}_f^2 / \overline{\rho}_0^2$$

and which determines the mean $\hat{\mathbf{X}}$ of the posterior distribution. In addition there is now also the spatial variance parameter $\overline{\rho}_0^2$ for the prior which also affects the variability of fitted curves as modelled by the posterior distribution.

Finally, exploiting the additivity of information in (8.13), a mean spatial variance $\overline{\rho}^2$ for the posterior can be computed from

$$\frac{1}{\overline{\rho}^2} = \frac{1}{\overline{\rho}_0^2} + \frac{1}{\overline{\rho}_f^2}$$

and is exact if there is no invariant subspace ($\mathcal{S}_s = 0$), and otherwise is a lower bound on mean variance. A confidence region based on $\overline{\rho}$ is shown in figure 8.4.

Note that the matrix $\overline{S}$ in (8.14) has become singular, so that the covariance matrix P_0 does not exist and the prior density p_0 cannot formally be normalised. In fact p_0 can still be treated as a consistent Gaussian distribution. It consists of a valid Gaussian density restricted to shape-space $\mathcal{S} \ominus \mathcal{S}_s$ excluding the invariant subspace $\mathcal{S}_s$, together with a uniform density

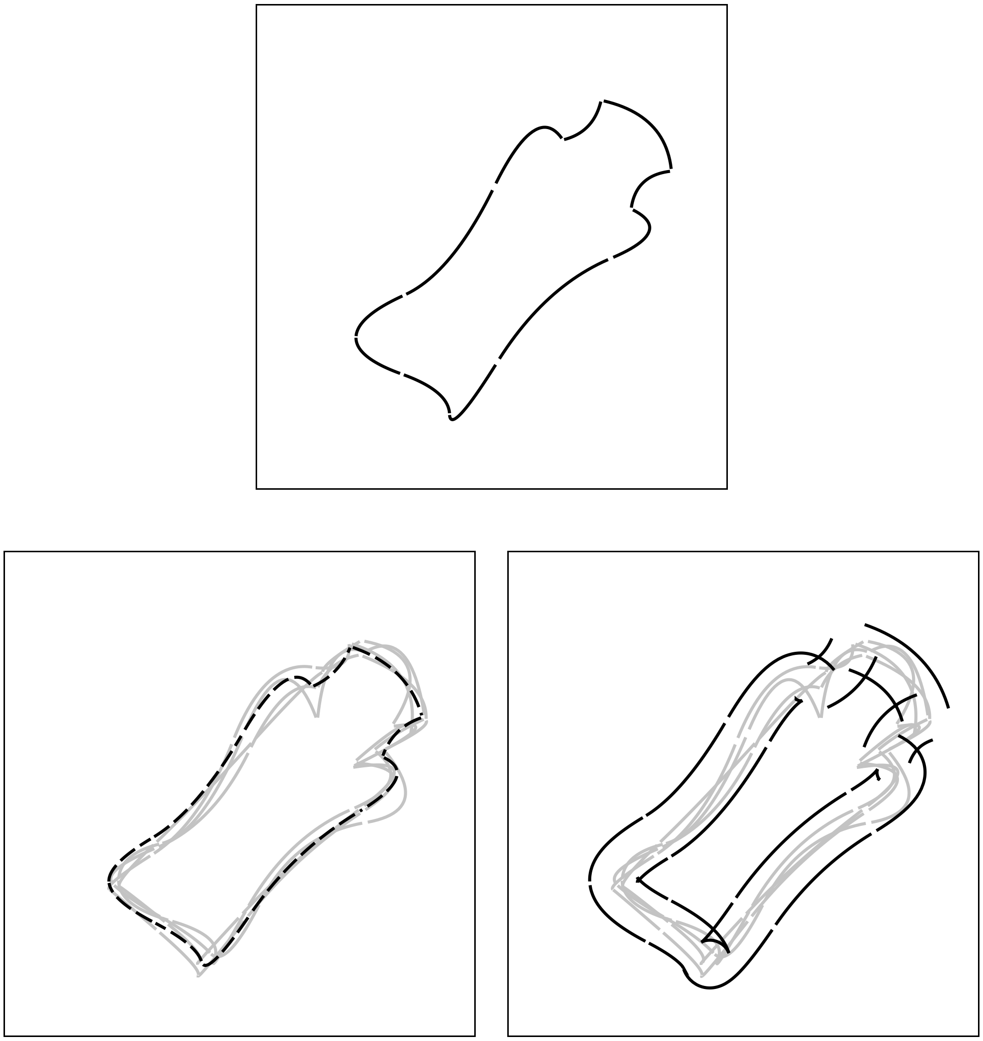


Figure 8.4: Sampling from the posterior *Data (top) is given, together with a prior over spline space with the Euclidean similarities as an invariant subspace (to allow free rotation and translation). Curves sampled from the posterior distribution sweep out a distribution (left), whose mean (dashed line) is simply the regularised estimate obtained in chapter 6 (figure 6.2 on page 119), with* $\alpha = 0.5$*. The sampled curves (right) lie comfortably inside a 95% confidence interval.*

over $\mathcal{S}_s$. It is the uniform density that cannot be normalised and gives rise to unbounded covariance. We could treat it formally as a sequence of Gaussians of increasing variance, understanding that any expression incorporating p_0 will be evaluated in the limit, where one exists. In fact this is achieved for our purposes simply by applying additivity of information to $\overline{S}$ just in the same way as for a non-singular $\overline{S}$. The result is a *posterior* Gaussian whose covariance generally exists and which can be normalised.

8.3 Probabilistic modelling of image features

The example above of a posterior distribution is a contrived one, for tutorial purposes, driven by ideal data in the form of a spline curve $\mathbf{Q}_f$. In chapter 6, methods were developed for fitting to real data $\mathbf{r}_f$ in the form of a curve sampled from an image. The importance of using only the normal component of displacement was explained. Now the probabilistic approach needs to be developed to encompass normal displacements in real image data.

As in the previous section, the data-term in the regularisation problem can be interpreted probabilistically, but now based on normal displacement to give an observation density:

$$p(\mathbf{r}_f|\mathbf{X}) \propto \exp -\frac{N_X}{2\overline{\rho}_f^2}\|\mathbf{r}-\mathbf{r}_f\|_{\overline{\mathbf{n}}}{}^2 \quad \text{with} \quad \mathbf{r}(s) = U(s)(W\mathbf{X}+\mathbf{Q}_0). \tag{8.15}$$

It can be shown (see below) that $\overline{\rho}_f^2$ is the contribution to mean-square value of the normal displacement $(\mathbf{r}-\overline{\mathbf{r}})\cdot\mathbf{n}$, in the fitted curve, due solely to measurement error.

Now the discrete form (6.13) from chapter 6 for $\|\mathbf{r}-\mathbf{r}_f\|_{\overline{\mathbf{n}}}{}^2$ can be used, giving approximately:

$$p(\mathbf{r}_f|\mathbf{X}) \propto \exp -\frac{1}{2\sigma^2}\sum_{i=1}^{N}\left[(\mathbf{r}_f(s_i)-\mathbf{r}(s_i))\cdot\overline{\mathbf{n}}(s_i)\right]^2. \tag{8.16}$$

where

$$\sigma = \overline{\rho}_f\sqrt{\frac{N}{N_X}}. \tag{8.17}$$

One interpretation of this observation density is in terms of the variability of individual measurements. Suppose the marginal density of a single measurement

$$[\mathbf{r}_f(s_i)-\mathbf{r}(s_i)]\cdot\overline{\mathbf{n}}(s_i)$$

were a Gaussian $\mathcal{N}(0, \sigma^2)$ variable. Then the joint density (8.16) could be interpreted as treating the data $\mathbf{r}_f$ as a series of sampled image features

$$\mathbf{r}_f(s_i) \sim \mathcal{N}((\mathbf{r}(s_i), \sigma^2) \tag{8.18}$$

with mutually *independent* Gaussian distributions. To some extent such independence is plausible but it depends rather on what is the chief source of the variability in image-feature measurements. If the variability were thought to derive from noise, for example electrical or optical noise at individual pixels, it is probably defensible to regard noise sources at neighbouring pixels as independent; even then, noise due to lighting flicker (especially fluorescent lighting) or power supply ripple would be highly correlated across pixels. In practice, however, the variability in feature position due to such noise is negligible. The dominant sources of variability reflect grosser discrepancies than mere device noise from the camera.

- Imperfections in the shape-space model mean that the image outline $\mathbf{r}_f$ will not lie entirely in the shape-space $\mathcal{S}$. The difference

  $$\mathbf{e}(s) = \mathbf{r}_f(s) - U(s)(WX_f + \mathbf{Q}_0)$$

  between the outline and its projection onto shape-space acts as spatially correlated noise.

- Background texture and occasional obscuration by passing objects will cause features to be picked up by image processing that do not belong to the modelled object. Such errors may be gross and, again, spatially correlated.

The result is that statistical independence between neighbouring $\mathbf{r}_f(s_i)|\mathbf{X}$ is plausible only if features are not sampled too densely, and this is a limitation of the observation model (8.15). It is therefore important to note that the interpretation of (8.16) in terms of statistically independent measurements $\mathbf{r}_f(s_i)$ is by no means the only consistent interpretation (see discussion at the end of this section).

Algorithm to construct the posterior

Construction of the posterior density $p(\mathbf{X}|\mathbf{r}_f)$ from image measurements $\mathbf{r}_f(s_i)$, $i = 1, \ldots, N$ uses precisely the recursive fitting algorithm from chapter 6 (figure 6.7 on page 127), but now interpreted probabilistically. The algorithm computes an "aggregated observation" $\mathbf{Z}$ and the associated statistical information for estimation of $\mathbf{X}$ is S, also output by the algorithm. In fact

$$\mathbf{Z}|\mathbf{X} \sim \mathcal{N}(S\mathbf{X}, S),$$

so that $\mathbf{Z}$ is an unbiased estimator for $S\mathbf{X}$ (rather than for $\mathbf{X}$ directly). Its covariance is in fact S *(sic)*, following from the fact that the statistical information in $S^{-1}\mathbf{Z}$ is S.

The posterior distribution is a Gaussian

$$\mathbf{X}|\mathbf{Z} \;\sim\; \mathcal{N}(\hat{\mathbf{X}}, P) \quad \text{where} \quad P = (\overline{S} + S)^{-1}. \tag{8.19}$$

An example of a posterior constructed in this way from an image is given in figure 8.5.

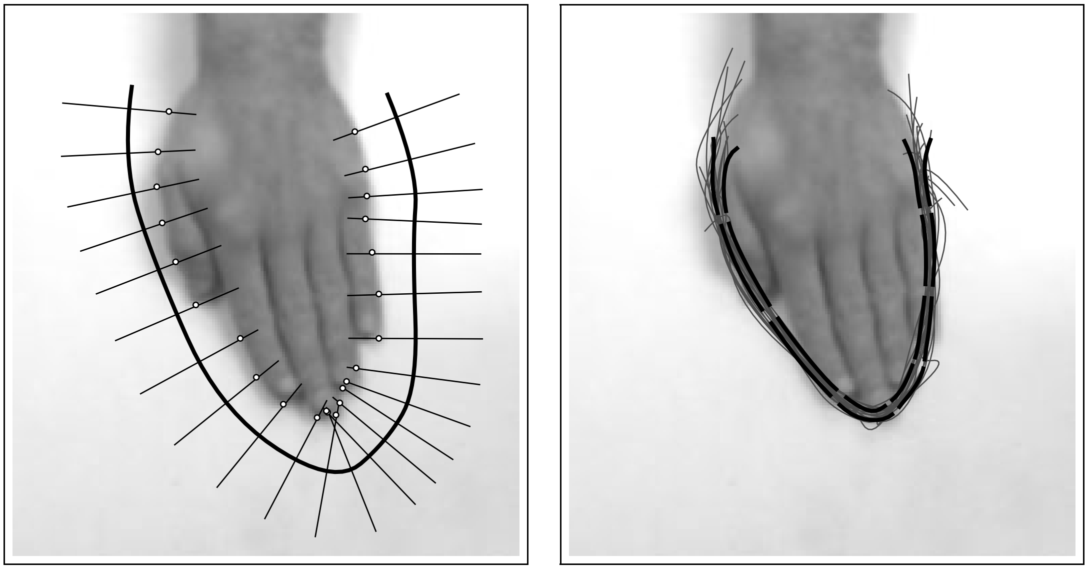


Figure 8.5: Constructing a posterior distribution from an image. *The image data (left) is shown overlaid with a broad search region determined by the chosen prior. When combined with image observations, a relatively tight posterior distribution results, illustrated here by randomly sampled curves and 95% confidence region (right).*

Independence of measurements

The functional form of the density $p(\mathbf{r}_f|\mathbf{X})$ does not in fact uniquely determine the conditional distribution for observations until a parameterisation is specified for $\mathbf{r}_f(s)$, the image feature. For example $\mathbf{r}_f(s)$ could be modelled as a spline with parameters $\mathbf{Q}_f$ of dimension N_f. Then the conditional density can be used to compute probabilities via integrals of the form

$$\int p(\mathbf{r}_f|\mathbf{X})\, d\mathbf{Q}_f.$$

To examine the question of independence of the $\mathbf{r}_f(s_i)$, the conditional density must be transformed from the $\mathbf{Q}_f$ parameterisation to a new parameter set $(\mathbf{r}_f(s_1), \ldots, \mathbf{r}_f(s_N))$. This could be done (without loss of generality) by expressing the new parameters in terms of the first N components in the vector $\mathbf{Q}_f$, and integrating over the remaining $N_f - N$ components. This is possible *only if* $N < N_f$. Otherwise there exists no density in the new variables, let alone an independent one. This is intuitively reasonable. When the underlying parameterisation $\mathbf{Q}_f$ of the image curve has relatively few degrees of freedom (N_f is small), this represents an image curve with strong spatial correlation, such as a spline with few control points. In that case, successive measurements made on the curve cannot be mutually independent. To summarise, the form of (8.16) does not force the deduction that individual image measurements $\mathbf{r}_f(s_i)$ are independent. This conclusion is important to the operation of the validation gate, discussed below.

Mean-square normal displacement

It was claimed above that $\overline{\rho}_f^2$ is the mean-square value of the normal displacement $(\mathbf{r} - \mathbf{r}_f) \cdot \mathbf{n}$. This could be proved directly by integration, as was done earlier for $\overline{\rho}_0$ in the norm-squared prior. A concise alternative is to appeal to the well-known property of Boltzmann distributions in statistical mechanics for random systems whose energy $U(\mathbf{X})$ is a quadratic function of parameters $\mathbf{X}$ and is distributed as $p(\mathbf{X}) \propto \exp -U$. The mean energy of such a system is simply $\overline{U} = N_X/2$, where N_X is the number of degrees of freedom of the parameter set.

8.4 Validation gate

The validation gate was introduced in chapter 6 (section 6.3) as a mechanism for detecting outliers, features that are misplaced or missing when, for instance, part of an object is obscured. The idea was to test the innovation ν_i to ensure that it was not too large; the difficulty was to decide how large is too large. Now in the probabilistic framework, there is a basis for making this test as a statistical hypothesis test on the innovation, whose Gaussian distribution is known. The innovation (6.14) can be expanded as

$$\nu_i = \ [\mathbf{r}(s_i) - \overline{\mathbf{r}}(s_i)] \cdot \overline{\mathbf{n}}(s_i) \ + \ [\mathbf{r}_f(s_i) - \mathbf{r}(s_i)] \cdot \overline{\mathbf{n}}(s_i),$$

in which the first term has a $\mathcal{N}(0, \rho_n^2(s))$, distribution, from (8.7). The second term is a little more difficult to deal with. In the case that the measurements $\mathbf{r}_f(s_i)$ are considered independent, it is Gaussian $\mathcal{N}(0, \sigma^2)$ from (8.18). At the other extreme, in the limit of high spatial interdependence of measurements, its variance should tend towards 0. This means that

$$\nu_i \ \sim \ \mathcal{N}(0, \rho^2(s)) \ \text{ where } \ \rho_n^2(s) \le \rho^2(s) \le \rho_n^2(s) + \sigma^2. \tag{8.20}$$

A conservative assumption of high spatial dependence would suggest

$$\rho(s) = \rho_n(s). \tag{8.21}$$

The distribution of the innovation suggests a test that if $|\nu_i| > \kappa\rho(s)$ then the measurement $\mathbf{r}_f(s)$ is invalidated. Taking the default value $\kappa = 2$, at approximately 95% statistical significance the measurement is regarded as an outlier. In practice this is implemented by using a search segment (figure 8.1) of length $\rho(s)$ so that when $|\nu_i| > 2\rho(s)$, no feature will be found. The recursive fitting algorithm of chapter 6 can be modified to take account of outliers. The iterative step in the original algorithm (figure 6.7 on page 127) is rewritten to ignore outliers, as in figure 8.6.

Note that the modified algorithm requires that $\overline{S}$ have full rank and that excludes the use of an invariant subspace in the prior which always makes $\overline{S}$ singular. This is a real and reasonable limitation. The idea of the invariant subspace is that it allows certain shape variables to be unconstrained. In that case, the absolute position of a given point $\mathbf{r}(s)$ on the curve is also unconstrained by the prior (even though constraints may apply to the relative positions of pairs of points). Validity checks are only effective when the constraints imposed by the prior on individual points are reasonably tight. This requires the prior on the shape $\mathbf{X}$ as a whole to be reasonably tight — none of the eigenvalues of $\overline{S}$ should be too close to 0. The effect of outlier rejection on fitting is illustrated in figure 8.7.

8.5 Learning the prior

The importance of a prior distribution is that it draws interpretations of data towards the more plausible shapes in a class of curves. The value of such stabilising behaviour

Iterative fitting step

```
Iterate, for $i = 1, \dots, N$:
    $\nu_i = (\mathbf{r}_f(s_i) - \overline{\mathbf{r}}(s_i)) \cdot \overline{\mathbf{n}}(s_i);$
    $\mathbf{h}(s_i)^T = \overline{\mathbf{n}}(s_i)^T U(s_i) W;$
    $\rho_i = \sqrt{\mathbf{h}(s_i)^T \overline{P} \mathbf{h}(s_i)};$
    If $(|\nu_i| < \kappa \rho_i)$ then
        $S_i = S_{i-1} + \frac{1}{\sigma_i^2} \mathbf{h}(s_i) \mathbf{h}(s_i)^T;$
        $\mathbf{Y}_i = \mathbf{Y}_{i-1} + \frac{1}{\sigma_i^2} \mathbf{h}(s_i) \nu_i;$
    Else
        $S_i = S_{i-1};\ Y_i = Y_{i-1};$
```

Figure 8.6: Curve fitting with outliers. *The recursive curve-fitting algorithm in figure 6.7 on page 127 can be modified to ignore outliers. Its iterative step is replaced by the one shown here. The search-line length factor is taken to be* $\kappa = 2$ *by default. (Note that* $\overline{P} = \overline{S}^{-1}$ *needs to have been computed at the start of the algorithm, and that* $\overline{S}$ *must therefore be of full rank.)*

is only as good as the prior model itself, embodied by the coefficients in the information matrix $\overline{S}$. So far, we have proposed priors based on uniform, isotropic error ($\overline{S} \propto \mathcal{H}$) and a modification that allows for an invariant subspace. These are reasonable choices if little is known precisely about the class of curves. If, however, a more specific prior could be obtained from actual snapshots of a curve in a series of representative configurations, stabilisation should be very much more effective, and this does indeed prove to be the case.

The first step towards learning is to acquire a training set, a set of curves $\{\mathbf{X}_k,\ k = 1, \dots, M\}$, where $\mathbf{X}_k$ are vectors in a shape-space $\mathcal{S} = \mathcal{L}(W, \mathbf{Q}_0)$ which may either be the spline space $\mathcal{S}_Q$ or a subspace of $\mathcal{S}_Q$. The set is supposed to capture the outline

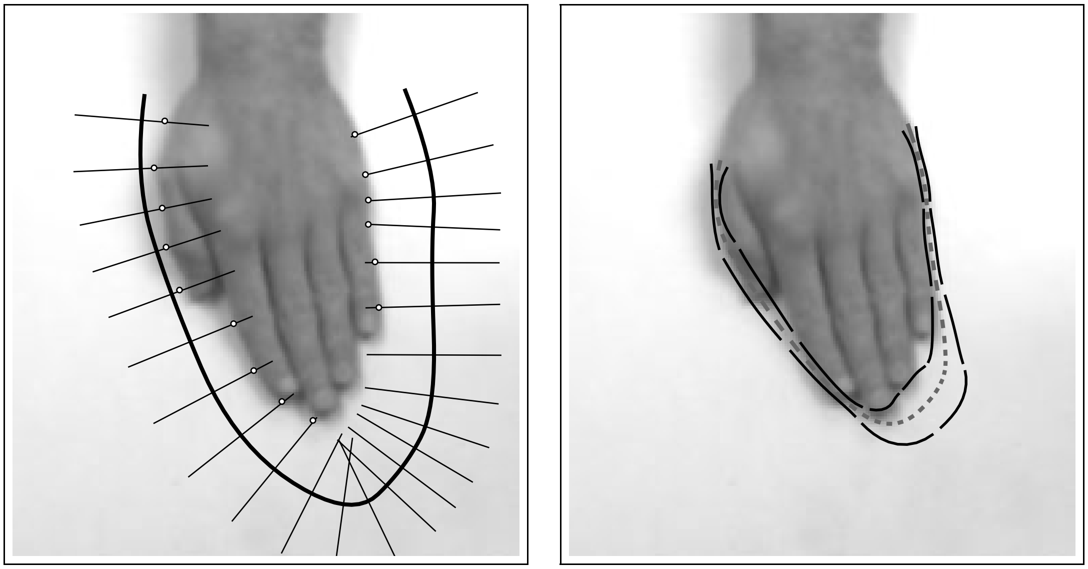


Figure 8.7: Validation failure causes a bulge in the posterior confidence region. *Validation fails around the fingertips (left) resulting in a bulge in the 95% confidence region (right) such that it still contains the true edge of the hand.*

being modelled in many characteristic poses. For current purposes, the order in which poses appear in the sequence is immaterial, though order will matter in later chapters when object dynamics are being modelled. One could think of each $\mathbf{X}_k$ as a contour drawn by hand on a given image but in practice the temporal tracking technology developed later allows these outlines to be acquired automatically.

If the prior distribution is assumed to be Gaussian $\mathcal{N}(\overline{\mathbf{X}}, \overline{P})$, then "maximum likelihood" estimators for the parameters of the distribution are given by the sample mean

$$\overline{\mathbf{X}} = \frac{1}{M}\sum_{k=1}^{M}\mathbf{X}_k \tag{8.22}$$

and sample covariance

$$\overline{P} = \frac{1}{M}\sum_{k=1}^{M}(\mathbf{X}_k - \overline{\mathbf{X}})(\mathbf{X}_k - \overline{\mathbf{X}})^T. \tag{8.23}$$

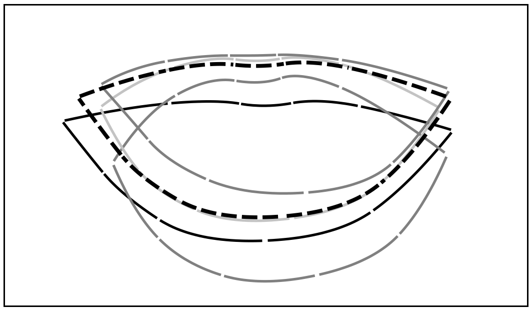


Figure 8.8: Sampling from a prior for lip shape *Lips are tracked during speech, as in figure 1.10 on page 14, to capture a 60 second sequence comprising 3000 successive lip shapes. The sequence is used to estimate a Gaussian prior distribution in shape-space for lip shape; random sampling from the prior generates plausible lip configurations, as shown. (Data courtesy of Robert Kaucic.)*

For example, in figure 8.8 a Gaussian prior distribution has been learned from a long sequence of lip motion during speech. The resulting prior is simulated by random sampling and the sampled curves are plausible lip configurations.

The matrix $\overline{P}$ is a $N_X \times N_X$ matrix and should be invertible so that $\overline{S} = \overline{P}^{-1}$ can be calculated as required for the recursive curve-fitting algorithm. However, from (8.23),

$$\operatorname{rank}(\overline{P}) \leq M - 1$$

so that $\overline{P}$ certainly cannot be inverted unless $M > N_X$, and in practice one would expect a data set several times that minimum size. Alternatively, if a sufficiently large data set is not available and the estimated $\overline{P}$ is singular then one approach is to restrict the shape-space sufficiently that the reduced covariance matrix is no longer singular, and this is discussed next.

8.6 Principal Components Analysis (PCA)

Even if the estimated prior covariance matrix $\overline{P}$ is not technically singular, it is frequently close to singularity even when the training sequence is long ($M > N_X$). This happens when the typical motions of the object under study are largely accounted for by a few independent modes of motion. If $\overline{P}$ is found to have a large condition number, larger say than 100, this suggests that, on the basis of the available data, the shape-space $\mathcal{S}$ in which data was collected is unnecessarily large. Efficiency of the matching algorithms described earlier, and of later tracking algorithms, would be considerably enhanced if it could be replaced by a smaller shape-space. The covariance matrix $\overline{P}$ can be used to determine a smaller shape-space $\mathcal{S}' = \mathcal{L}(W', \mathbf{Q}'_0) \subset \mathcal{S}$, a subspace of $\mathcal{S}$ with dimension N'_X, that spans, at least approximately, all of the shapes in a training sequence $\mathbf{X}_1, \ldots, \mathbf{X}_M$. This idea was first introduced by Cootes and Taylor, in the special case of polygonal contour models, which they dubbed the "Point Distribution Model" or PDM. The PCA method is explained starting with "classical PCA" which is well-known and easily accessible in textbooks, via two refinements, to "L_2-norm PCA in shape-space," the recommended method for dealing with curves. In the special case that $M \le N_X$ and the covariance matrix $\overline{P}$ is actually singular, the simplest way to build a reduced shape-space is to take appropriate linear combinations of the frames in the training sequence — so-called key-frames, as described in chapter 4.

Classical PCA

The classical approach to PCA would allow the following version of the approximation problem to be solved. Given a long ($M > N_X$) training sequence, solve:

$$\min_{W', \mathbf{Q}'_0, \mathbf{X}'_1, \ldots, \mathbf{X}'_{N'_X}} \left(\sum_{k=1}^{M} |\mathbf{Q}_k - \mathbf{Q}'_k|^2 \right) \tag{8.24}$$

where

$$\mathbf{Q}'_k = W'\mathbf{X}'_k + \mathbf{Q}'_0 \quad \text{and} \quad \mathbf{Q}_k = W\mathbf{X}_k + \mathbf{Q}_0.$$

The distance measure $|\cdot|$ is the Euclidean norm, that is, $|\mathbf{Q}|^2 \equiv \mathbf{Q}^T\mathbf{Q}$. The well known solution to this problem gives $\mathbf{Q}'_0 = \overline{\mathbf{Q}}$, the mean of the training sequence, and W' is a matrix whose columns are the first N'_X of the orthonormal eigenvectors of the covariance matrix

$$\Sigma = \frac{1}{M} \sum_{k=1}^{M} (\mathbf{Q}_k - \overline{\mathbf{Q}})(\mathbf{Q}_k - \overline{\mathbf{Q}})^T,$$

ordering the eigenvectors in descending order of their (necessarily positive) eigenvalues. The intuitive interpretation is that the eigenvalues represent variance in the training set in the mutually orthogonal directions of the eigenvectors; the first N'_X eigenvectors form a basis for the subspace of dimension N'_X that "explains" as much as possible of the variance in the training set. Note that since the data $\mathbf{Q}_k$ all live within a shape-space of dimension N_X there will be at most N_X non-zero eigenvalues of Σ.

L_2-norm PCA

In previous chapters, it has been argued that the induced L_2-norm is the natural norm over spline space. In that case, the classical problem above is not entirely applicable. Instead, the approximation problem should be posed as follows:

$$\min_{W',\mathbf{Q}'_0,\mathbf{X}'_1,\ldots,\mathbf{X}'_{N'_X}} \left(\sum_{k=1}^{M} \|\mathbf{Q}_k - \mathbf{Q}'_k\|^2 \right), \tag{8.25}$$

in which the L_2-norm $\|\cdot\|$ has been substituted where Euclidean norm $|\cdot|$ appeared before. The solution to this problem, which can be derived from the solution to the classical problem (see below), is that $\mathbf{Q}'_0 = \overline{\mathbf{Q}}$ as before and W' is a matrix whose columns are the first N'_X eigenvectors of the matrix $\Sigma\mathcal{U}$.

L_2-norm PCA over shape-space

Finally L_2-norm PCA can be re-expressed in terms of training-set covariance $\overline{P}$ in shape-space, without recourse to the (considerably larger) covariance matrix Σ in spline space. This gives the recommended algorithm in figure 8.9. Results from the application of the algorithm to the lip-motion sequence are shown in figure 8.10. PCA analysis for full facial expression is illustrated in figure 8.11. In neither case are individual PCA components recognisable as particular expressions; rather they are mixtures of expressions. It is when they are taken as a set that they are meaningful, as a basis for the repertoire of commonly occurring deformations.

Remember that Euclidean norm $|\mathbf{X}|$ in shape-space has no clear geometrical meaning. As a result classical PCA applied to the shape-space training set $\mathbf{X}_1, \ldots, \mathbf{X}_M$ would certainly not be meaningful.

Residual PCA

A constructive description of shape-space, in which the shape-matrix is composed from transformations of a template or from key-frames, is rather desirable because each of the components of the shape-vector $\mathbf{X}$ has a clear interpretation. For example, the shape-space (4.20) composed of key-frames under Euclidean similarity is a constructive shape-space for which a given $\mathbf{X}$ represents an explicit combination of rigid transformation and facial expression. In contrast, deriving a shape-matrix from PCA loses the clear interpretation, but has the advantage that no prior insight into

Problem: given training data $\mathbf{X}_1, \ldots, \mathbf{X}_M$ over shape-space $\mathcal{S}$, find a subspace $\mathcal{S}' = \mathcal{L}(W', \mathbf{Q}'_0)$ of dimension N'_X to minimise

$$\sum_{k=1}^{M} \|\mathbf{Q}_k - \mathbf{Q}'_k\|^2$$

where

$$\mathbf{Q}_k = W\mathbf{X}_k + \mathbf{Q}_0 \quad \text{and} \quad \mathbf{Q}'_k = W'\mathbf{X}'_k + \mathbf{Q}'_0.$$

Algorithm

1. Construct the training-set mean

$$\overline{\mathbf{X}} = \frac{1}{M}\sum_{k=1}^{M} \mathbf{X}_k.$$

2. Construct the training-set covariance

$$\overline{P} = \frac{1}{M}\sum_{k=1}^{M} (\mathbf{X}_k - \overline{\mathbf{X}})(\mathbf{X}_k - \overline{\mathbf{X}})^T.$$

3. Find eigenvectors $\mathbf{v}_1, \ldots, \mathbf{v}_{N_X}$ of $\overline{P}\mathcal{H}$, in descending order of eigenvalue.

4. Construct $W'' = \left(\mathbf{v}_1, \ldots, \mathbf{v}_{N'_X}\right)$.

5. The parameters $\mathbf{Q}'_0$ and W' of the shape-subspace are then:

$$\begin{aligned} \mathbf{Q}'_0 &= W\overline{\mathbf{X}} + \mathbf{Q}_0 \\ W' &= WW''. \end{aligned}$$

Figure 8.9: Algorithm for L_2 PCA in shape-space.

likely transformations or deformations is required. It is possible, to some extent, to enjoy the best of both worlds. Residual PCA operates on a constructive shape-space that does not totally cover a certain data set, and fills in missing components by PCA. Then the constructive subspace retains its interpretation and only the residual

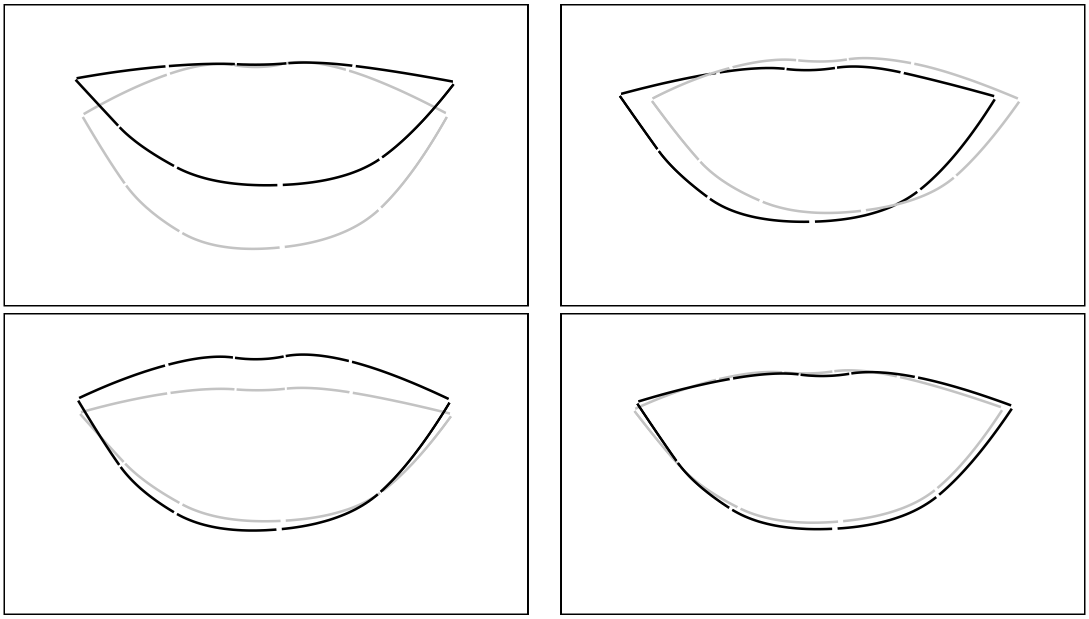


Figure 8.10: *The lip-motion sequence of figure 8.8 on page 176 is used here in PCA. The first four eigenvectors, illustrated here, capture over 95% of the variance in the data set. (Eigenvectors are displayed here as displacements either side of the mean, with magnitude equal to their standard deviation over the sequence.) (Figures by courtesy of Robert Kaucic.)*

components, covered by PCA, cannot be directly interpreted.

Given training data $\mathbf{X}_1, \ldots, \mathbf{X}_M$, collected from a shape-space $\mathcal{S}$, and a constructive subspace $\mathcal{S}_c = \mathcal{L}(W^c, \mathbf{Q}_0^c)$, it is desired to augment $\mathcal{S}_c$, increasing its dimension by N'_X, to cover the training-data set more completely. Residual PCA achieves this by computing, after steps 1,2 of the algorithm of figure 8.9, the mean and variance of the residue of the training set:

$$\overline{\mathbf{X}}^r = (I - E^c)\overline{\mathbf{X}} \tag{8.26}$$

$$\overline{P}^r = (I - E^c)\overline{P}(I - E^c)^T \tag{8.27}$$

where $E^c = W(W^c)^+$ is the matrix for projection onto the subspace $\mathcal{S}_c$. Step 3 of the algorithm computes the first N'_X eigenvectors $\mathbf{v}_1, \ldots, \mathbf{v}_{N'_X}$ now of $\overline{P}^r$, rather than $\overline{P}$ as in the original algorithm. Steps 4 and 5 proceed as before, computing the template $\mathbf{Q}'_0$ and shape-matrix W' which now form a shape-space for the *residual* variation in

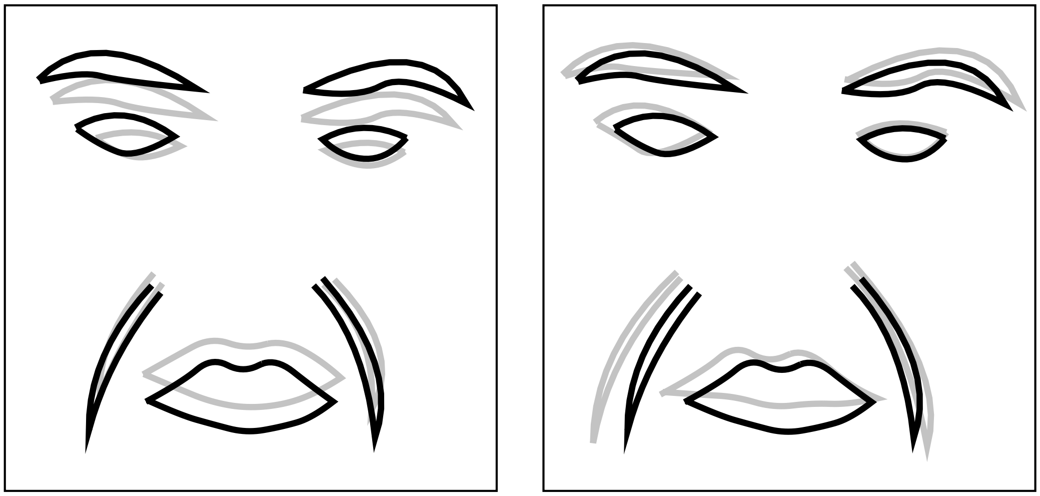


Figure 8.11: *Facial expression as in figure 1.2 on page 6 is analysed by PCA. The first two eigenvectors are illustrated here. (Figures by courtesy of Benedicte Bascle.)*

the training set that was not covered by $\mathcal{S}_c$. The augmented shape-space is then

$$\mathcal{S}_a = \mathcal{L}(W^a, \mathbf{Q}_0^a)$$

where

$$\mathbf{Q}_0^a = E^c \mathbf{Q}_0^c + W' \overline{\mathbf{X}}^r$$

and the shape-matrix

$$W^a = (W^c | W^r),$$

a concatenation of columns from the shape-matrices for the constructive and residual spaces.

As an illustration, the Residual PCA algorithm is applied to the lip-motion sequence used earlier. Instead of applying PCA directly to the motion sequence as in figure 8.13, a constructive shape-space $\mathcal{S}_c$, a two-dimensional space of rigid translations, is used. Translation accounts for about 58% of the motion of the training set (horizontal 24%, vertical 34%) and since head motion is likely to be somewhat independent of speech, it is natural to try to discount it. The covariance $\overline{P}^r$ of the residue of the sequence with translation removed can be used in its own right to construct a prior, as in figure 8.13. Then the first two components from residual PCA, illustrated

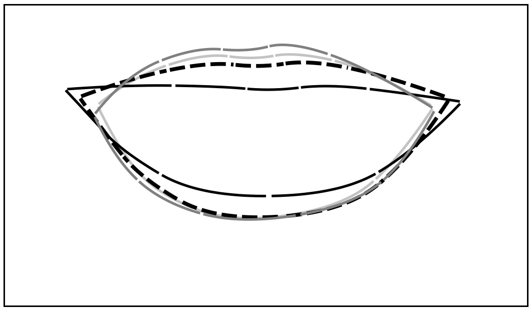


Figure 8.12: Sampling from a prior for lip shape, excluding translation *Random sampling illustrates how a learned prior generates plausible lip configurations, as in figure 8.8, but now with the rigid translation due to head motion excluded, leaving just the lip motions associated with speech.*

in the figure, can be expected to bear the bulk of the visual information that is related to speech.

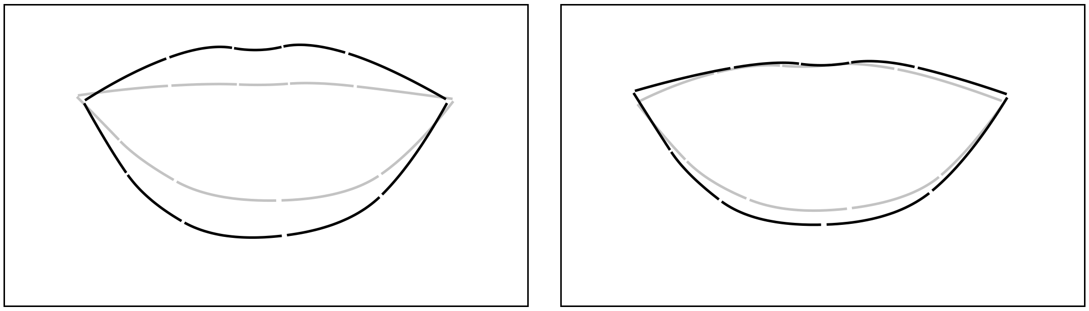


Figure 8.13: The lip-motion sequence is analysed by residual PCA. *All rigid translation is first excluded from the motion sequence by projection. Translation and the further 2 eigenvectors shown here account for over 95% of the variance in the data set.*

Derivation of PCA algorithms

First L_2-norm PCA is derived from classical PCA. Noting that $\|\mathbf{Q}\| = |\mathcal{U}^{1/2}\mathbf{Q}|$, the problem (8.25) can be recast as a minimisation, in the form of the classical problem (8.24), of

$$\sum_{k=1}^{M} |\mathcal{U}^{1/2}\mathbf{Q}_k - \mathcal{U}^{1/2}\mathbf{Q}'_k|^2$$

with

$$\mathcal{U}^{1/2}\mathbf{Q}'_k = \mathcal{U}^{1/2}W'\mathbf{X}'_k + \mathcal{U}^{1/2}\mathbf{Q}'_0,$$

Its solution is therefore that

$$\mathcal{U}^{1/2}\mathbf{Q}'_0 = \mathcal{U}^{1/2}\overline{\mathbf{Q}}$$

and that $\mathcal{U}^{1/2}W'$ is made up of columns which are eigenvectors of the sample variance

$$\mathrm{Var}(\mathcal{U}^{1/2}\mathbf{Q}_k) = \mathcal{U}^{1/2}\Sigma\mathcal{U}^{1/2}.$$

This means that $\mathbf{Q}'_0 = \overline{\mathbf{Q}}$. Furthermore, a column $\mathcal{U}^{1/2}w$ of $\mathcal{U}^{1/2}W'$ satisfies

$$\mathcal{U}^{1/2}\Sigma\mathcal{U}^{1/2}\left(\mathcal{U}^{1/2}w\right) = \lambda\left(\mathcal{U}^{1/2}w\right)$$

which simplifies to

$$\Sigma\mathcal{U}w = \lambda w,$$

each column of W' is an eigenvector of $\Sigma\mathcal{U}$, as claimed. Note that equivalently w is a *generalised* eigenvector of the real, symmetric matrices $\mathcal{U}\Sigma\mathcal{U}$ and $\mathcal{U}$ (i.e. solutions of $\mathcal{U}\Sigma\mathcal{U}\mathbf{v} = \lambda\mathcal{U}\mathbf{v}$) and hence the eigenvalues are still positive, so that their descending order remains well-defined.

The shape-space version of the algorithm (figure 8.9) is obtained from the L_2-norm algorithm above by substituting

$$\mathbf{Q}_k = W\mathbf{X}_k + \mathbf{Q}_0 \quad \text{and} \quad \mathbf{Q}'_k = W'\mathbf{X}'_k + \mathbf{Q}'_0$$

so that

$$\overline{\mathbf{Q}} \to W\overline{\mathbf{X}} + \mathbf{Q}_0,$$

as in the algorithm, and

$$\Sigma\mathcal{U} \to W\overline{P}W^T\mathcal{U}$$

whose eigenvectors give W'. Writing $W' = WW''$, W'' must therefore be composed of eigenvectors of $\overline{P}W^T\mathcal{U}W$, but $W^T\mathcal{U}W = \mathcal{H}$ and this gives the required form for W'.

Bibliographic notes

The chapter is based on probabilistic analysis — for a review of basic probability, there is an excellent introductory book (Papoulis, 1990).

The use of a statistically estimated $\overline{P}$ in a Gaussian prior is equivalent to replacing the norm $\|\cdot\|$ in the regulariser of chapter 6 with a "Mahalanobis metric" (Rao, 1973) that emphasises improbable distortions, so that they are penalised more than probable ones.

Confidence regions for shapes were delimited by parallel curves. These are a well-known geometrical construct; one pitfall is that a smooth base curve can have offset curves that are not smooth but contain cusps (Bruce and Giblin, 1984). They are also known in CAD as "offset" curves (Faux and Pratt, 1979).

Principal Components Analysis (PCA) is a standard statistical technique (Rao, 1973). The idea of constructing a Gaussian curve model of reduced dimension, using PCA, was first developed for the case of polygonal outlines and termed the "Point Distribution Model" or PDM (Cootes et al., 1993), and subsequently extended to splines (Baumberg and Hogg, 1994). A related idea based on image intensities rather than curves is the eigen-image (Turk and Pentland, 1991) which can be used directly in tracking e.g. (Bregler and Omohundro, 1994; Black and Jepson, 1996). PCA is sometimes applied jointly to outline shape and image intensity distributions (Lanitis et al., 1995; Beymer and Poggio, 1995; Vetter and Poggio, 1996), to impressive effect. In standard PCA, it is common to pre-process the data by applying a diagonal scaling transformation (Ripley, 1996). This proved to be too restrictive for building curve models, for which general linear transformations were needed. In particular the idea of performing PCA in the L_2 norm, using the metric matrix $\mathcal{H}$ was developed in (Baumberg and Hogg, 1995a), and this is equivalent to scaling the data by $\mathcal{H}^{1/2}$. Generalised eigenvalues (Golub and van Loan, 1989) are used to combine scaling and PCA efficiently.

Shape priors discussed here have had a Gaussian form in shape-space. In the case of a norm-squared density over spline space ($S \propto \mathcal{U}$), the prior is actually a Gaussian Markov Random Field (MRF), of second-order in the case of quadratic splines. MRFs have been used widely for modelling prior distributions for curves (Grenander et al., 1991; Ripley and Sutherland, 1990; Storvik, 1994).

Chapter 9

Dynamical models

The remainder of the book aims to establish effective procedures for tracking curves in *sequences* of images. As with single images, the importance of powerful prior models of shape holds good, but now prior models can be extended to capitalise on the *coherence* of typical motions through a sequence. Crudely this could mean a repeated application of the regularised curve-fitting of chapter 6, in which the fitted curve in the $k-1$th frame of a sequence is used as an initial estimate of curve position and shape for the kth frame. In the probabilistic context of chapter 8 this would involve applying, to each frame, a Gaussian prior distribution with fixed covariance but whose mean was simply the estimated shape from the previous frame. This immediately suggests a more subtle approach. Rather than fixing the form of the prior via one constant covariance for all frames, it seems more natural to take the *posterior* from frame $k-1$ as the prior for frame k. In that way, it would not be merely an estimated shape that would pass from time-step to time-step but an entire probability distribution.

This idea is still too crude as it stands, for two reasons. First, it lacks a mechanism for extrapolation of motion between successive time-steps, and this is reasonable only for tracked objects which are moving slowly (figure 9.1). Secondly, as the posterior at one time-step is handed on to be the prior at the next, statistical information from measurements steadily accumulates. Accumulation proceeds without bound so that the statistical information $S(t_k)$ in the posterior continues to increase in successive frames and the corresponding covariance $P(t_k)$ decreases towards zero. This is counterintuitive — if the prior at time t_k has a vanishing covariance, its regularising effect will become too strong and image measurements will have an ever decreasing influence on estimated shape.

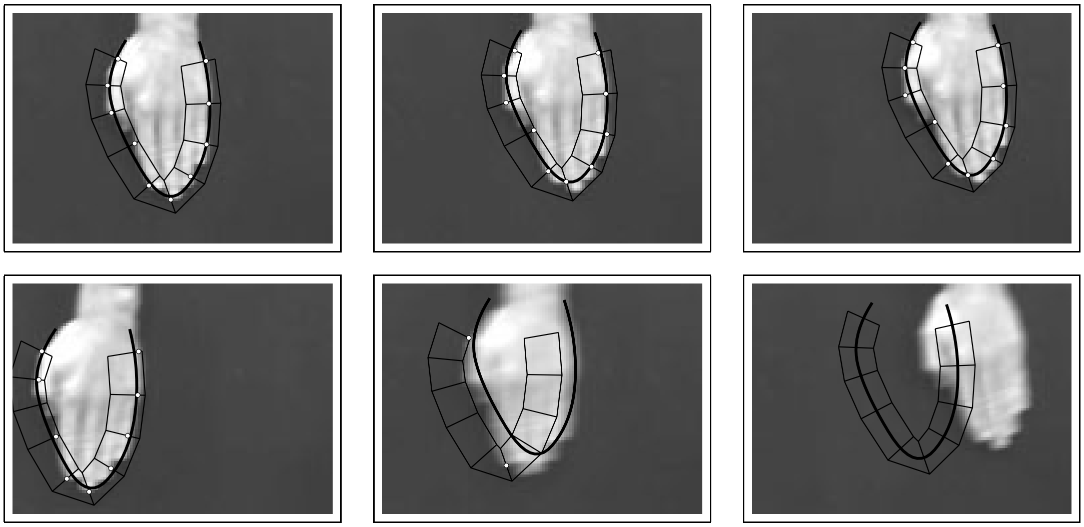


Figure 9.1: The need to exploit coherence of motion. *Example of hand tracking, showing that using the fitted curve at time* t_{k-1} *as the initial frame at time* t_k *is of limited use — it tracks slow motions (top) but not faster ones (bottom).*

An adequate statistical framework for motion tracking must therefore provide not only a prior for the first frame, similar to the prior in the static contour fitting problem but, far more importantly, a prior for possible motions, in the broad sense of rigid motion plus deformation of shape. This dynamical prior distribution should apply between all pairs $k-1, k$ of successive frames. It must have a deterministic part, giving the expected displacement between successive frames, and this addresses the first objection above about extrapolation of motion. Less obviously perhaps, it also needs a stochastic component, an injection of randomness, causing a steady "leakage" of information to counteract the otherwise unlimited accumulation of information that led to the second objection above.

In the previous chapter, the probabilistic framework for the fitting of curves to static images was expounded in three parts. First, the encoding of prior knowledge of curve shape had the form of a Gaussian prior probability distribution in shape-space. Secondly, the posterior probability distribution of curve shape given image data was computed by the recursive algorithm. Thirdly, rather than conjuring up a prior distribution which is merely plausible, a specific prior was actually learned from a training-data set. Now it is time to extend these three ideas, in this and the next two chapters respectively, to deal with curves in moving images. First, this chapter deals with prior models for curve shape and motion.

9.1 Some simple dynamical prior distributions

Consider the problem of building an appropriate dynamical model for the position of a hand-mouse engaged in an interactive graphics task. This is the problem that was illustrated in figure 1.16 on page 20 in which a hand whose motion is monitored visually controls a simulated object rendered in three dimensions. A typical trace in the xy plane of a finger drawing letters is shown in figure 9.2. If the entire trajectory were treated as a training set, the methods of the previous chapter could be applied to learn a Gaussian prior distribution for finger position. The learned prior is broad, spanning a sizable portion of the image area, and places little constraint on the measured position at any given instant. Nonetheless, it is quite clear from the figure that successive positions are tightly constrained. Although the prior covariance ellipse spans about

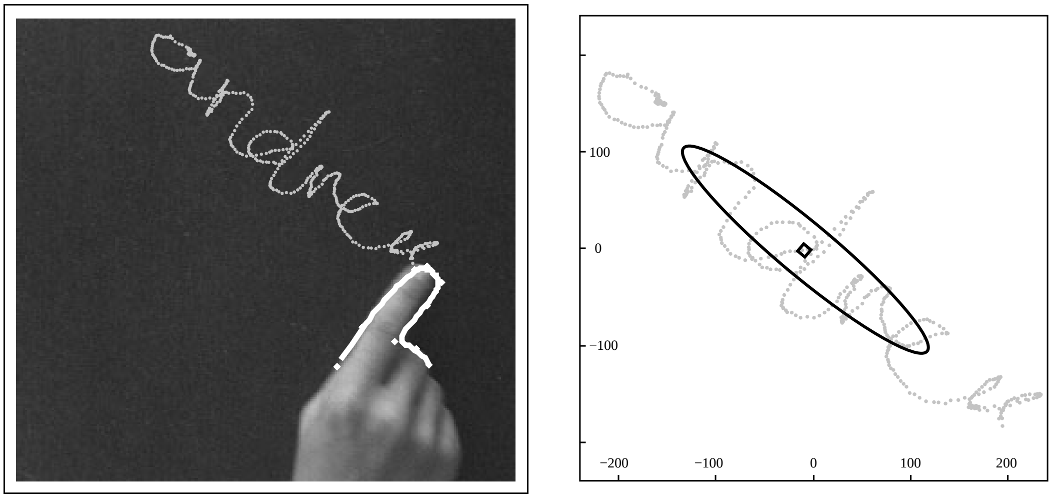


Figure 9.2: The moving finger writes. *The finger trajectory (left) which has a duration of about 10 seconds executes a broad sweep over the plane. If the trajectory is treated as a training set, the learned Gaussian prior is broad, as the covariance ellipse (right) shows. Clearly though, successive positions (individual dots represent samples captured every 20 ms) are much more tightly constrained.*

300×50 pixels, successive sampled positions are seldom more than 5 pixels apart!

For sequences of images, then, a global prior $p_0(\mathbf{X})$ is not enough. What is needed is a conditional distribution $p(\mathbf{X}(t_k)|\mathbf{X}(t_{k-1}))$ giving the distributions of possibilities for the kth shape $\mathbf{X}(t_k)$ *given* the $k-1$th shape $\mathbf{X}(t_{k-1})$. This amounts to a "first-order Markov chain" model in shape-space in which, although in principle $\mathbf{X}(t_k)$ may be correlated with all of $\mathbf{X}(t_1)\ldots\mathbf{X}(t_{k-1})$, only correlation with the immediate predecessor is explicitly acknowledged — the "Markov" assumption is made that

$$p(\mathbf{X}(t_k)|\mathbf{X}(t_1)\ldots\mathbf{X}(t_{k-1})) = p(\mathbf{X}(t_k)|\mathbf{X}(t_{k-1})).$$

This is an assumption that greatly simplifies the probabilistic model. (It will be argued later that it is actually an oversimplification and that a second-order Markov model, taking two predecessors into account, is needed.)

Continuing the Gaussian theme of earlier chapters, a simple, isotropic, first-order Gaussian Markov process in shape-space is

$$p(\mathbf{X}(t_k)|\mathbf{X}(t_{k-1})) \propto \exp -\frac{1}{2b^2}\|\mathbf{X}(t_k) - \mathbf{X}(t_{k-1})\|^2. \tag{9.1}$$

For example, in a shape-space $\mathcal{S}$ of translations in the plane, this process represents simply a two-dimensional, random walk of the centroid of a rigid shape, in which the average (root-mean-square) step length is $b\sqrt{2}$ and is isotropic — steps are equally likely to occur in all directions (figure 9.3 (top)). This particular form of random walk is known as "Brownian motion" and is well-known as a physical model of the thermodynamics of microscopic particles. Such a distribution looks too random to be generally useful for modelling the motion of real, massive objects. A predominantly deterministic motion with an added random component is more plausible (figure 9.3 (left)). A modest elaboration of the model above achieves this by including a constant offset $\mathbf{D}$:

$$p(\mathbf{X}(t_k)|\mathbf{X}(t_{k-1})) \propto \exp -\frac{1}{2b^2}\|\mathbf{X}(t_k) - \mathbf{X}(t_{k-1}) - \mathbf{D}\|^2. \tag{9.2}$$

In the case of planar translation, $\mathbf{D}$ is a vector representing the mean displacement in each time-step. The endpoints of random walks consisting of N steps, starting from a fixed origin, are distributed isotropically as a Gaussian in the plane (figure 9.3 (right)). Its distribution is the sum of N Gaussians, which is the single Gaussian whose covariance ellipse (a circle), and any confidence interval derived from it, grows steadily, in

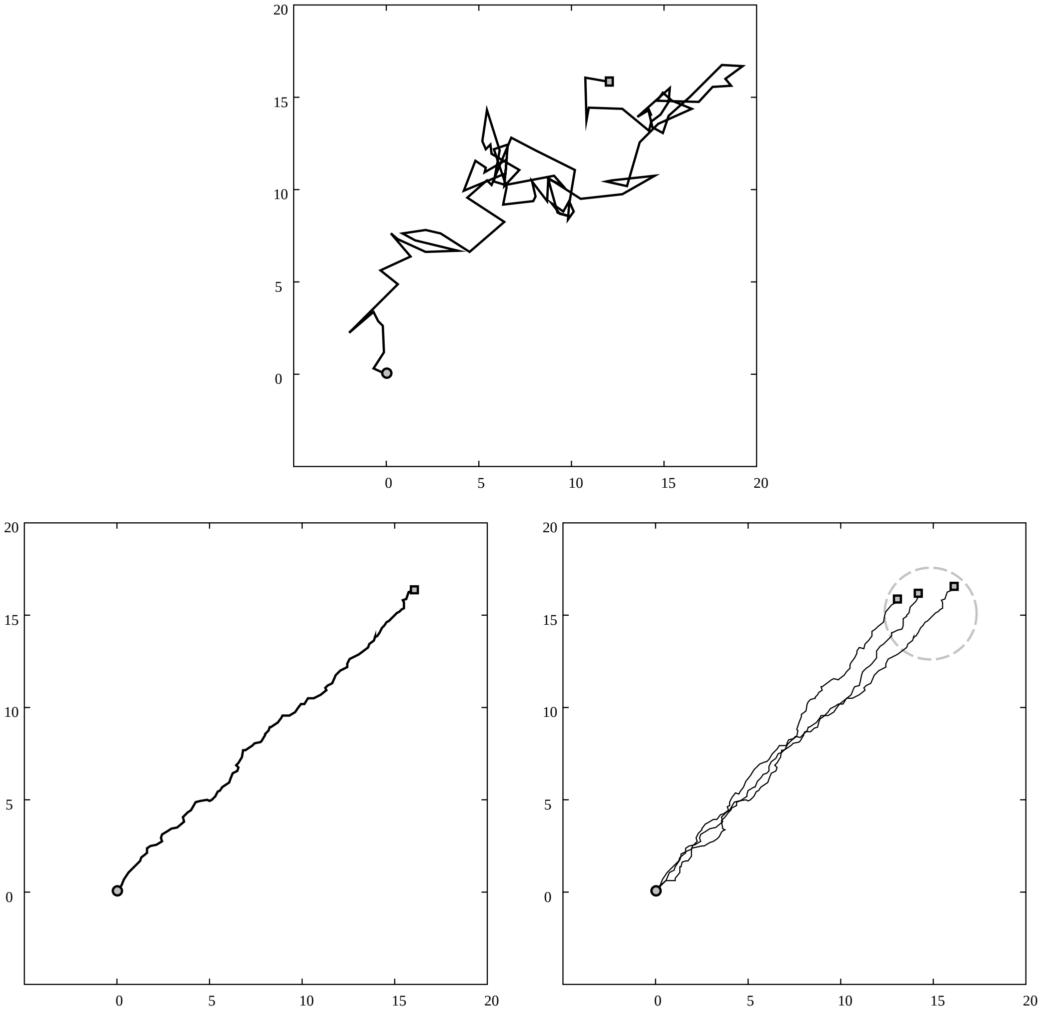


Figure 9.3: Brownian motion in the plane. *Plots show successive positions of the centroid of some rigid shape undergoing Brownian motion. (Top) an isotropic random walk (100 steps, $b = 1$). A random walk (left) with a superimposed drift velocity (100 steps, $b = 0.1$, drift (0.15,0.15) per time-step). Several samples (right) of random walk with drift; the endpoints of the random walks (marked with squares) have an isotropic Gaussian distribution in the plane (95% confidence ellipse shown dotted).*

fact in proportion to $\sqrt{N}$. This is precisely the steady leakage of information which was claimed earlier to be an essential component of a dynamical model.

Let us accept for the time being the random walk with drift as a plausible model of the translational motion of an object (though later in the chapter it will be argued that its trajectories are insufficiently smooth as a model of the motion of real physical bodies). Now consider instead deformations of the object, governed by the components of, say, a 4-dimensional subspace $\mathcal{S}_d$ containing all the affine components except translation. The random walk with drift is certainly not an appropriate model for deformation because, as we saw in the examples above, its variance grows unboundedly over time. In the case of the bottle example of the previous chapter this would imply a bottle distorting progressively, without limit. Somehow the small deformations that occur in a continuous fashion from frame to frame need to be forced to stay with an overall envelope. What is needed is a Markov process that looks like a random walk on a short time-scale but sweeps out some bounded distribution, presumably a Gaussian one, over long times. It turns out that a minor modification of discrete Brownian motion (9.1):

$$p(\mathbf{X}(t_k)|\mathbf{X}(t_{k-1})) \propto \exp -\frac{1}{2b^2}\|(\mathbf{X}(t_k)-\overline{\mathbf{X}}) - a(\mathbf{X}(t_{k-1})-\overline{\mathbf{X}})\|^2 \tag{9.3}$$

with $0 \leq a \leq 1$ gives a *constrained* Brownian process that has exactly the desired properties (figure 9.4). Initially it appears to behave like the unconstrained process but as the long-time limit is approached it becomes more and more evident that it lies within a limiting Gaussian envelope. Exactly what that distribution is and how it depends on a and b is clarified later.

Now, such a distribution, visualised so far only in terms of points in the plane, can be illustrated over a realistic shape-space. Take the bottle example of the last chapter, over an affine space $\mathcal{S}$, partitioned into 2-dimensional translational and 4-dimensional deformation subspaces $\mathcal{S}_s$ and $\mathcal{S}_d$ respectively. A plausible dynamical model for a translating bottle whose shape undergoes small distortions about a mean shape $\overline{\mathbf{X}}$ due to projective effects on the changing viewpoint might have two components. One is a constrained Brownian process in the deformation space $\mathcal{S}_d$. The other is an unconstrained Brownian process in the translational space $\mathcal{S}_s$. As figure 9.3 (left) showed, adding drift to Brownian motion models directional motion, but unfortunately only in a preassigned, fixed direction. If the direction of motion is not known *a priori* then the only model we have so far for translation (see later for better models) is pure Brownian motion. The combined model is expressed in terms of the following

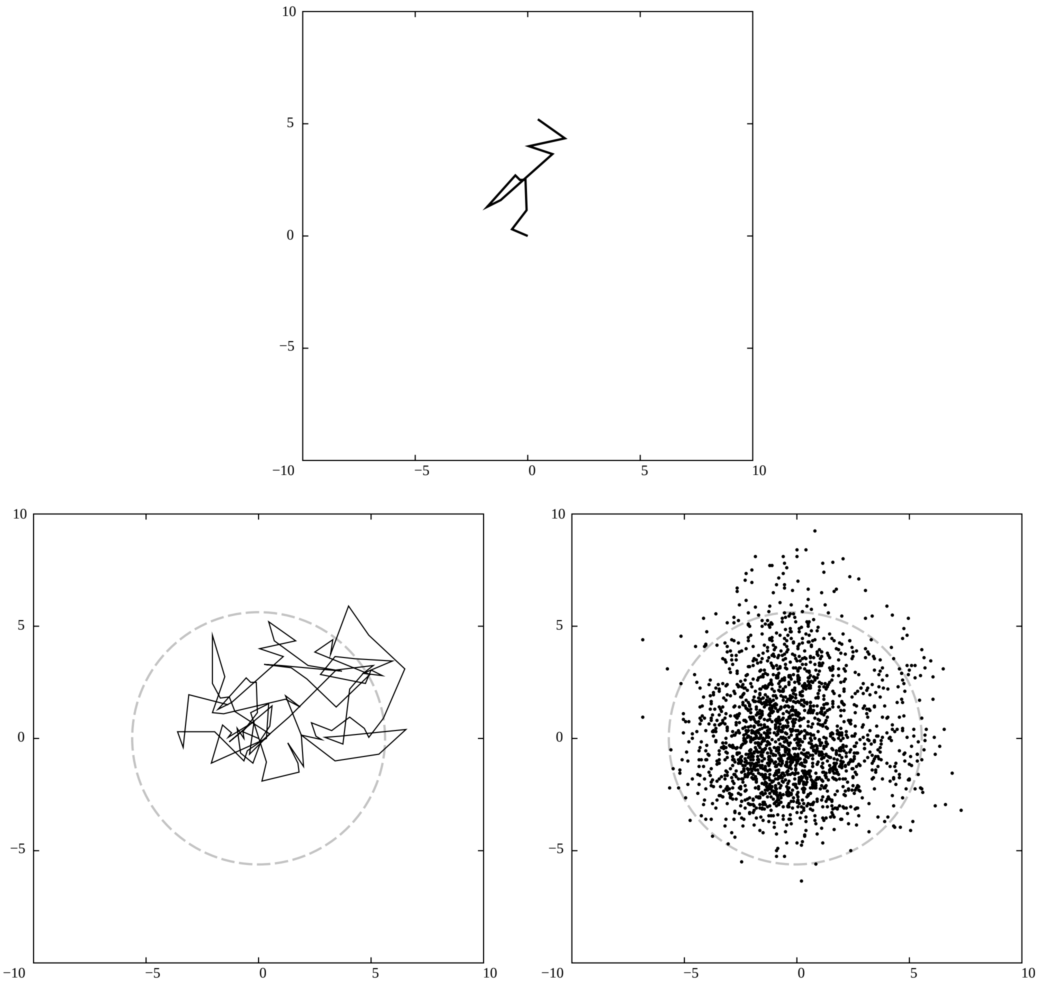


Figure 9.4: Constrained Brownian motion *with* $b = 1$ *and* $a = 0.9$*. On a small scale, the process appears like unconstrained Brownian motion with* $b = 1$ *so that (top) the first 12 steps are similar to the first 12 steps of figure 9.3 (top) but after 100 steps (left) things look quite different — the process is constrained by an overall Gaussian envelope whose 95% confidence ellipse is marked. After 2000 steps (right) the steady-state Gaussian envelope is quite evident (positions only shown, for clarity).*

first-order Markov conditional density:

$$p(\mathbf{X}(t_k)|\mathbf{X}(t_{k-1})) \;\propto\; \exp -\frac{1}{2}\left\| \frac{1}{b^s}E^s\left(\mathbf{X}(t_k)-\mathbf{X}(t_{k-1})\right) + \frac{1}{b^d}E^d\left((\mathbf{X}(t_k)-\overline{\mathbf{X}})-a^d(\mathbf{X}(t_{k-1})-\overline{\mathbf{X}})\right)\right\|^2 . \tag{9.4}$$

(Subspace projection matrices E^s and E^d were defined two chapters back, in (6.5) on page 117.) A sample path from a simulation of such a dynamical model is illustrated in figure 9.5. It is perhaps not altogether clear from the Markov density (9.4) exactly how simulation can be done: in fact, expressing the density instead as an "Auto-regressive" process amounts to an explicit prescription for simulation and this is explained next.

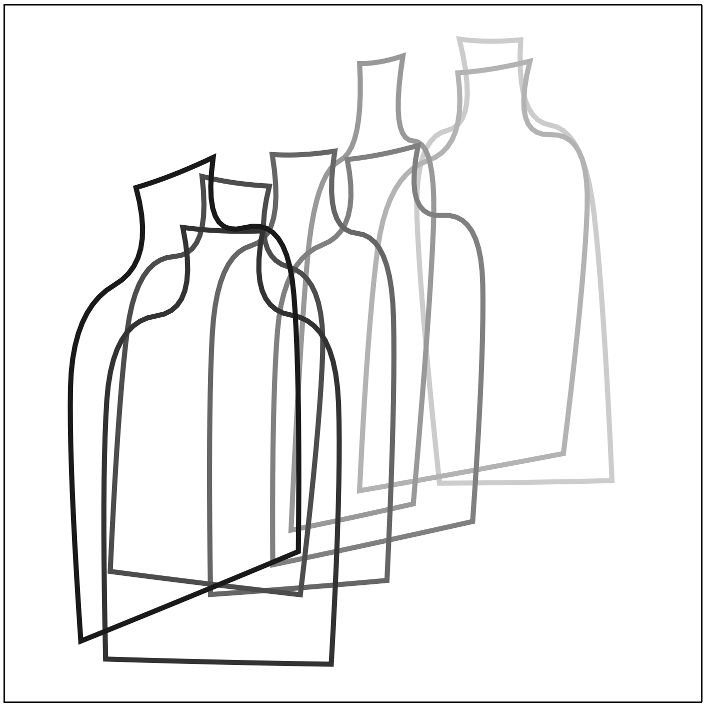


Figure 9.5: First-order stochastic motion. *The figure shows a sample path from a distribution (9.4) which translates as an unconstrained Brownian process ($b^s = 30$ pixels) and deforms as a constrained Brownian process ($a^d = 0.5,\ b^d = 15$ pixels). The sequence shown displays every 10th frame.*

9.2 First-order Auto-regressive processes

We have seen some specific examples of useful Markov processes and generally they can be expressed in the form

$$p(\mathbf{X}(t_k)|\mathbf{X}(t_{k-1})) \propto \exp -\frac{1}{2}\|B^{-1}(\mathbf{X}(t_k) - A\mathbf{X}(t_{k-1}) - \mathbf{D})\|^2. \tag{9.5}$$

Just as Gaussian priors $\mathbf{X} \sim \mathcal{N}(\overline{\mathbf{X}}, \overline{P})$ in the previous chapter could be expressed as $\mathbf{X} = \overline{\mathbf{X}} + B\mathbf{w}$ which was amenable to simulation, so also the Markov process can be expressed in a generative form:

$$\mathbf{X}(t_k) = A\mathbf{X}(t_{k-1}) + \mathbf{D} + B\mathbf{w}_k, \tag{9.6}$$

specifying a "sample path" for the process, in which each $\mathbf{w}_k$ is a vector of N_X independent random $\mathcal{N}(0,1)$ variables and $\mathbf{w}_k$, $\mathbf{w}_{k'}$ are independent for $k \neq k'$. A form which is not quite as general but will prove convenient is

$$\mathbf{X}(t_k) - \overline{\mathbf{X}} = A(\mathbf{X}(t_{k-1}) - \overline{\mathbf{X}}) + B\mathbf{w}_k \tag{9.7}$$

and this is the standard form for a first-order "Auto-regressive" (AR) process. The constants in the two forms are related by

$$(I - A)\overline{\mathbf{X}} = \mathbf{D}.$$

The AR standard form has the advantage that the parameter $\overline{\mathbf{X}}$ has a clear interpretation: it is the steady-state limit of the mean value of $\mathbf{X}(t_{k-1})$. As $k \to \infty$, and provided the process is stable (a condition for stability is $\|A\|_2 < 1$ — see appendix B.1), the distribution $\mathbf{X}(t_{k-1})$ approaches a steady state $\mathcal{N}(\overline{\mathbf{X}}, P_\infty)$ for some limiting covariance P_∞. The steady-state distribution is exactly the overall envelope distribution for constrained Brownian motion.

In fact the AR form of the process can be used to derive the distribution $\mathbf{X}(t_k) \sim \mathcal{N}(\hat{\mathbf{X}}(t_k), P(t_k))$ at all times t_k. Since, by definition, $\hat{\mathbf{X}}(t_k) = \mathcal{E}[\mathbf{X}(t_k)]$ and $P(t_k) = \mathcal{V}[\mathbf{X}(t_k)]$, taking the expectation and variance of (9.7) gives, respectively, the "mean-state" equation and

$$\hat{\mathbf{X}}(t_k) - \overline{\mathbf{X}} = A(\hat{\mathbf{X}}(t_{k-1}) - \overline{\mathbf{X}}) \tag{9.8}$$

or, in the alternative form,

$$\hat{\mathbf{X}}(t_k) = A\hat{\mathbf{X}}(t_{k-1}) + \mathbf{D}, \tag{9.9}$$

and the "covariance" or Riccati equation

$$P(t_k) = AP(t_{k-1})A^T + BB^T. \tag{9.10}$$

It is clear by inspection that $\mathbf{X}(t_k) = \mathbf{X}(t_{k-1}) = \overline{\mathbf{X}}$ satisfies (9.8) so that the mean of the steady-state distribution must be $\overline{\mathbf{X}}$. The steady-state covariance $P_\infty \equiv \lim_{k\to\infty} P(t_k)$ must be a fixed point of (9.10):

$$P_\infty = AP_\infty A^T + BB^T \tag{9.11}$$

and although this equation can be solved exactly (by diagonalising A) the most straightforward method, if not the most efficient, is simply to iterate (9.10) to convergence.

Examples of AR processes

A variety of Markov processes can be succinctly expressed in the AR formalism.

1. **Unconstrained Brownian motion**

   Setting $A = I$, the AR process (9.7) simplifies to

   $$\mathbf{X}(t_k) = \mathbf{X}(t_{k-1}) + B\mathbf{w}_k$$

   which is independent of $\overline{\mathbf{X}}$. The fact that $\overline{\mathbf{X}}$, the steady-state mean, should not need to be specified is reasonable given the process has no steady state. In fact the mean value is constant throughout: $\hat{\mathbf{X}}(t_k) = \mathbf{X}(t_0)$, and, from (9.10), the covariance evolves as

   $$P(t_k) = P(t_{k-1}) + BB^T$$

   so that, given an exact initial value for $\mathbf{X}(t_0)$, $P(t_k) = kBB^T$ which is unbounded, with no limiting value as $k \to \infty$. The covariance ellipse has a mean-square radius of $\mathrm{tr}(P)$ and therefore grows in proportion to $\sqrt{k}$. There remains the choice of B which is effectively a scaling transformation for the process. For example, in the plane, $B = I_2$ produces a statistically isotropic random walk, whereas

   $$B = \begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}$$

   is a random walk which has been stretched out by a factor of 2 in the horizontal direction. In any shape-space, a norm-squared Brownian process (9.1) has the special form $B = b_0\mathcal{H}^{-1/2}$ so that $P(t_k) = k\, b_0^2\, \mathcal{H}^{-1}$, and $\mathbf{X}(t_k)$ is distributed as a norm-squared Gaussian in which average curve displacement is proportional to $\sqrt{k}$.

2. **Unconstrained Brownian motion with drift**

   In fact the addition of drift is outside the scope of the standard form (9.7) but can be expressed in the alternative form (9.6) with $A = I$ and $\mathbf{D}$ being the drift per unit time-step. Still there is no steady state but, from (9.9) and (9.10), the distribution $\mathbf{X}(t_k) \sim \mathcal{N}(\hat{\mathbf{X}}(t_k), P(t_k))$ is given by

   $$\hat{\mathbf{X}}(t_k) = k\mathbf{D} + \hat{\mathbf{X}}(t_0) \quad \text{and} \quad P(t_k) = kBB^T,$$

   so that the covariance ellipse grows in proportion to $\sqrt{k}$ as before but also drifts at a constant rate.

3. **Constrained Brownian motion**

   The constrained Brownian motion model of (9.3), with its norm-squared Gaussian density, can be expressed in the standard form of the AR process with coefficients $A = aI$, $B = b\mathcal{H}^{-1/2}$. If $a^2 = 1 - \epsilon$ with $0 < \epsilon \ll 1$ then, on a small time-scale, the process is almost indistinguishable from the case $\epsilon = 0$ of unconstrained Brownian motion, but in the longer term the distinction which was clear graphically in figure 9.4 is apparent also from the covariance equation. Since $\|A\|_2 = a < 1$, the covariance reaches a steady state, at a fixed point (9.11)

   $$P_\infty = \frac{1}{\epsilon}BB^T = \frac{b^2}{\epsilon}\mathcal{H}^{-1}$$

   and the limiting Gaussian envelope for the process is then $\mathcal{N}(\overline{\mathbf{X}}, P_\infty)$. The results of the previous chapter imply that the curve has an average displacement from the template of $\overline{\rho} = b\sqrt{N_X/\epsilon}$ in the steady state.

   In the simple example of figure 9.4, in which $a = 0.9$, $b = 1$ and the space is translational, we get $\overline{\rho} = \sqrt{2/(1 - 0.9^2)} = 3.24$ units. The 95% confidence circle has a radius approximately 1.73 times as great, approximately 5.6 units, as illustrated in the figure.

4. **Motion in affine space**

   A Markov process suitable for an affine shape-space was set out above (9.4) and simulated in figure 9.5. It can be expressed as an AR process in standard form (9.7) with $\overline{\mathbf{X}}$ as the mean shape for the object and the other coefficients can be shown to be

$$A = E^s + a^d E^d,$$

simply applying a different multiplier in each subspace, and similarly

$$B = \left(b^s E^s + b^d E^d\right) \mathcal{H}^{-\frac{1}{2}},$$

partitioned across the two subspaces.

This model does not reach a steady state because the Brownian motion in the translational subspace causes unbounded variance $P(t_k)$. However, the deformation component $E^d \mathbf{X}(t_k)$ is a constrained Brownian process which reaches a steady state as above.

9.3 Limitations of first-order dynamical models

First-order AR processes seem to meet some of the requirements for dynamical modelling. They are stochastic and can model entire families of motions. They deal with changes of position and shape and so can impose strong incremental constraints when global constraints are weak. They allow a global distribution to be chosen in addition to and independently of the local process. They appear to be the simplest available models that achieve these properties. Certainly, a first-order model is better than no dynamical model at all, as far as effective tracking is concerned. They do have limitations however.

First, unconstrained Brownian motion with drift, although it seems to model noisy directional motion at a constant average velocity, has an average direction that is fixed over time and must be known in advance. However, a good model for the motion of the writing finger would be a prior distribution that favours smooth motion of approximately constant velocity, but in an arbitrary direction, and allowing that velocity to change slowly in magnitude and direction. Another example is tracked vehicle motion (figure 9.6) in which slow, gradual changes of velocity are typical. More rapid changes in a given component of velocity are likely to occur during vehicle manoeuvres. Again, a model is called for that allows for substantial changes in velocity components over time, albeit mostly slow changes, and this is beyond the scope of the first-order model.

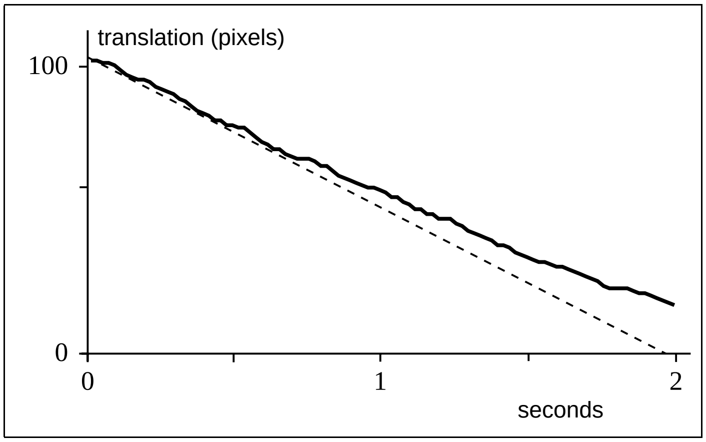


Figure 9.6: Tracking moving vehicles. *One translational component of tracked image motion for cars on a highway (figure 1.3 on page 7) is plotted here. Velocity is approximately constant over short time-scales. Here image velocity decreases gradually as the vehicle recedes.*

A second requirement is to be able to model oscillations that occur in many dynamical systems of interest. For example the tracked motion of the beating heart in the introductory chapter of the book is, not surprisingly, strongly oscillatory, as figure 9.7 shows. Any first-order AR process $\mathbf{X}(t_k)$ is a regularly sampled version of an underlying continuous-time process $\mathbf{X}(t)$, sampled at regular intervals $t_k = k\tau$ (see appendix B.1). The underlying continuous process has a characteristic time-course or "impulse response" that follows an exponential decay of the form $\exp -\beta t$, without any oscillatory component. Such a process would appear, therefore, to be unable to model oscillatory signals adequately.

Clearer evidence of the inadequacy of first-order models comes from examining the "spectral" characteristics of motion. In an AR process, the decaying or resonant response is driven by Gaussian random noise. Thus a first-order AR process, despite its decaying impulse response, does not actually decrease in magnitude because it is continuously excited by noise. The result is a complex superposition of exponentials which is succinctly characterised by its "power spectrum," the distribution of signal power as a function of frequency. For a first-order system, the power density is greatest at low frequency, falling off steadily as frequency increases.

Consider the example of lip motion during speech. It does not approach perfect periodicity as did the motion of the heart. Its motion does appear to have distinct

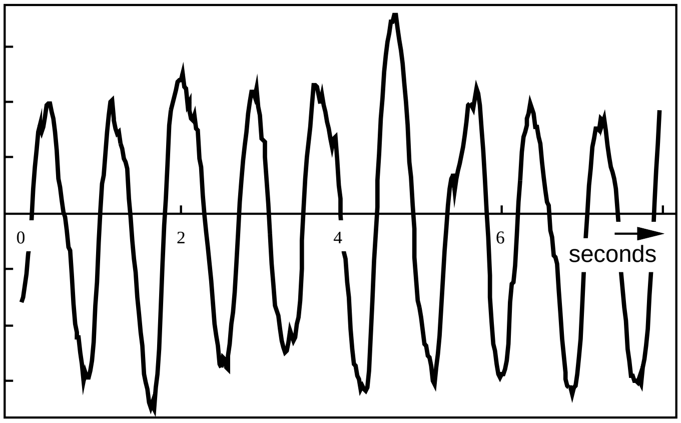


Figure 9.7: Tracked motion of a beating heart. *The first principal component of the motion of a beating heart, tracked in an ultrasound image sequence (figure 1.12 on page 16) is, of course, highly periodic. (Data courtesy of Gary Jacob and Alison Noble.)*

periodic elements (figure 9.8) but over a spread of frequencies. A spectral analysis of the signal (figure 9.9) shows the "power spectrum" of the speech signal — its distribution of power as a function of frequency. This is done by computing the

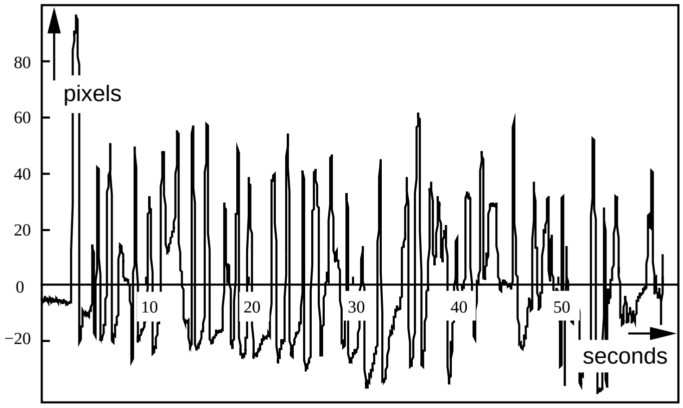


Figure 9.8: Lip motion during speech is oscillatory *This plots the opening/shutting motion of frontally viewed lips during a 60 second sequence of continuous speech. The density of peaks and troughs suggests oscillation at frequencies around 0.5–1 Hz.*

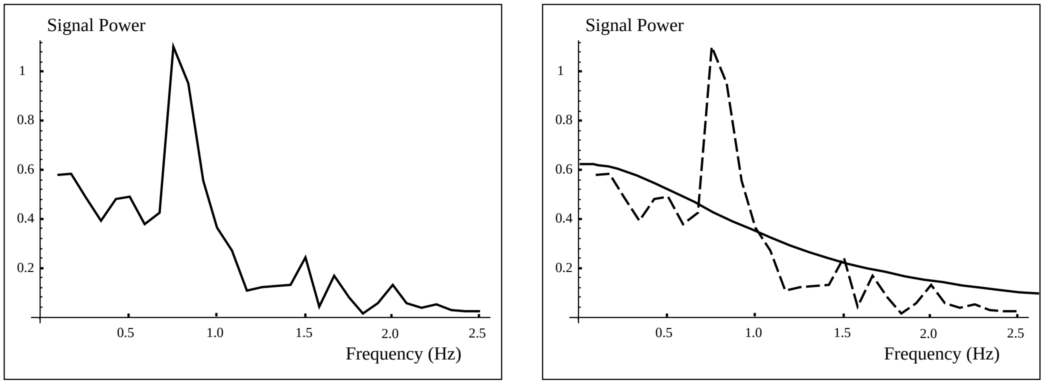


Figure 9.9: Spectral analysis of lip motion suggests second-order dynamics *Signal power in the lip-motion signal of figure 9.8 peaks (left) at frequency of about 0.8 Hz, reflecting the average spacing of peaks and troughs. A first-order AR process can be chosen with a spectrum (right) that models the background power level effectively but does not capture the peak.*

Discrete Fourier Transform (DFT) of the signal and plotting the square of its complex amplitude. The resulting power spectrum shows a background power distribution across all frequencies, decreasing in magnitude as frequency increases. Superimposed on this is a clear "resonant" peak at a frequency which appears to correspond well with the spacing of peaks and troughs in the original signal. Now it is known that for a first-order AR process in 1 dimension

$$x(t_k) = ax(t_{k-1}) + bw_k$$

the corresponding power spectrum has the form (see appendix B.1):

$$S_{xx}(f) \propto \frac{1}{1+\gamma f^2} \tag{9.12}$$

where γ is a constant that depends on the parameters a and b of the AR process. This spectrum always has maximum power at zero frequency $f = 0$ and hence cannot have a resonant peak at some frequency $f > 0$. Choosing a value of γ by hand to fit the lip-motion spectrum, it is possible to explain the background power distribution (figure 9.9) but the peak cannot be modelled. When the model is used to generate a sample (figure 9.10), the simulated signal is a poor replica of the original data.

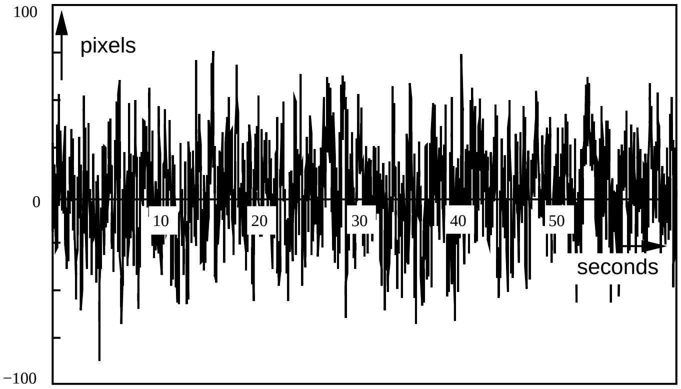


Figure 9.10: Simulated first-order lip motion *Parameters for a first-order AR process are chosen to correspond to the fitted power spectrum in figure 9.9 (left). A sample path for the process is illustrated here and should be compared with the original signal in figure 9.8. The simulated signal comprises far more rapid transitions than the original data it is supposed to model.*

Transitions in the simulated signal are too rapid, and this is consistent with the excessive signal power at high frequencies in the first-order model. The simulated signal does not appear to contain an underlying oscillation and this is consistent with the absence of a resonant peak in the first-order fitted spectrum.

9.4 Second-order dynamical models

Randomly excited harmonic motion

The simplest auto-regressive processes that meet the additional requirements, both for oscillatory and translational motion, are "second-order" processes (see below). They are typically resonant with a characteristic time-course in the form of a damped oscillation $\exp -\beta t \cos 2\pi f t$. The power spectrum of a second-order AR process (see appendix B.2) is:

$$S_{xx}(f) \propto \frac{1}{\pi^2(f_0^2 - f^2)^2 + \beta_0^2 f^2} \tag{9.13}$$

where β_0 is a "damping" constant with the dimensions of inverse time. It is clear that this spectral density reaches a maximum at approximately $f = f_0$ (provided $\beta_0 \ll f_0$), and is therefore capable of representing resonant or frequency-tuned behaviour. The width of the resonant peak, determined jointly by parameters f_0 and β_0, is $\Delta f_0 \approx f_0^2/\beta_0$. Choosing f_0 and β_0 by hand to match the centre and width of the resonant peak in the lip-motion data of figure 9.8 gives a much improved fit, as figure 9.11 shows. Now the peak in the spectral power distribution is represented, at least approximately, although the background power density may have been approximated better by the first-order model, especially at high frequency — power density in the second-order model tails off rather too fast. As a final piece of empirical evidence in favour of second-order models, the parameters f_0 and β_0 can be used to specify an AR process representing a second-order Markov chain (see next section) which can be sampled by driving it with randomly generated noise $\mathbf{w}_k$, as was done earlier with first-order AR processes. The result (figure 9.12) appears quite plausibly to be a signal of a similar type to the original, real lip-motion data.

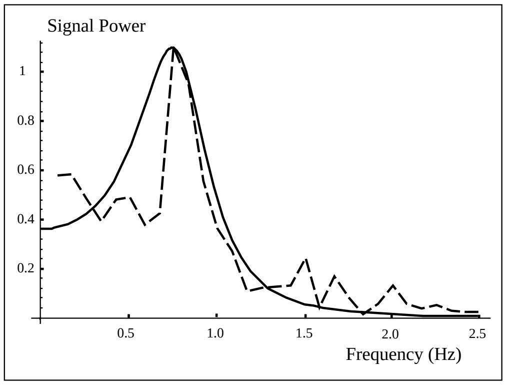


Figure 9.11: Second-order power spectrum model *A second-order model can represent the spectral peak in the lip-motion data of figure 9.8.*

Physical realisability

A further point of principle that favours second-order models over first-order ones comes from considering a temporal process $x(t)$ with power spectrum $S_{xx}(f)$ as a component of the physical motion $\mathbf{X}(t)$ of a body with mass. In that case, the corresponding velocity $v(t) = \dot{x}(t)$ has a power spectrum proportional to $f^2 S_{xx}(f)$ and the mean kinetic energy associated with that motion is proportional to

$$\mathcal{E}[v(t)^2] \propto \int_{-\infty}^{\infty} f^2 S_{xx}(f)\, df$$

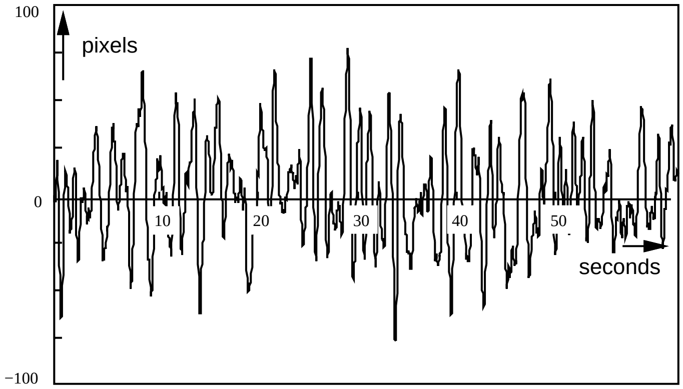


Figure 9.12: Simulated second-order lip motion *Parameters for a second-order AR process are chosen to correspond to the fitted power spectrum in figure 9.11. A sample path for the process is illustrated here and should be compared with the original signal in figure 9.8 — the distribution of spacings between peaks appears to be similar.*

and this integral must give a finite result for the model to be realisable as a physical process. For the first-order power spectrum $S_{xx}(f)$ in (9.12) the integral diverges, ruling out first-order models as unrealisable (other than with ideal, massless bodies). For the second-order spectrum (9.13) however, the integral converges and this is true of higher order AR models also.

It was noted above that although the resonant peak in the lip-motion spectrum was captured by a second-order model, the tail of the spectrum was actually modelled better by the first-order model. This might suggest that a weighted sum of the first- and second-order processes would fit better than either first- or second-order alone. This may be so but the mixed model would share the unrealisability of the pure first-order model; this is because the first-order spectrum dominates at large f, causing divergence of the kinetic energy integral. Mixtures of second- and higher-order models would satisfy realisability constraints. (In the language of "poles" and "zeros", popularly used to describe spectra, a pure nth order AR model has n poles and no zeros — an "all-pole" spectrum. The mixed first- and second-order model has 2 poles and 1 zero and is unrealisable. Generally physical realisability, in the mechanical sense, demands at least two more poles than zeros.) Furthermore, learning dynamics (see chapter 11) is rather more difficult for the mixed model than for the pure second-order model.

If physical realisability demands second-order or higher, conventional dynamical modelling suggests that second-order may be sufficient. For a system with just one degree of freedom, the situation is relatively straightforward. If it is assumed that potential energy and frictional power dissipation are each quadratic functions $\frac{1}{2}kx^2$ and $\frac{1}{2}\nu\dot{x}^2$ respectively then the equation of motion is not only second-order but also linear:

$$m\ddot{x} = -\nu\dot{x} - kx + f$$

where f is an externally applied force. Note that the quadratic assumptions cover the cases of potential energy due to elasticity (Hooke's law) or gravity and dissipation due to viscous forces at relatively low speeds. If f comes from some class of possible temporal force functions the result will be a class of motions $x(t)$. The simplest random model for f is as a "white noise" signal $f(t) = bw(t)$, where $w(t)$ is a Wiener process, the continuous counterpart to the uncorrelated discrete noise signal $\mathbf{w}_k$ in AR processes. In that case, the spectrum of $x(t)$ is second-order and all-pole.

Multidimensionally, in shape-space, motion $\mathbf{X}(t)$ is a little more difficult to describe. A general formalism ("Lagrangian dynamics" — see bibliographic notes) is available to convert a physical description of mass distribution, potential energy and dissipation into equations of motion. As before, the equations are of second-order, and linear if potential energy and power dissipation are quadratic:

$$\ddot{\mathbf{X}} = F_0\mathbf{X} + F_1\dot{\mathbf{X}} + G_0\mathbf{w}, \tag{9.14}$$

where $\mathbf{w}(t)$ is a vector of N_X independent Wiener processes and F_0, F_1 and G_0 are $N_X \times N_X$ matrices. Physical realisability means that F_0 and F_1 must be negative definite, with real eigenvalues and eigenvectors, but are not generally symmetric. These conditions guarantee stability of $\mathbf{X}(t)$, as might be expected for a physical process.

Lastly, note that $\mathbf{X}(t)$, being a shape-space variable, refers to the image of a physical object, rather than to the object itself. However, it was argued in chapter 4 that projection from world to image is (approximately) a linear mapping. That ensures that the general form of a linear model in world coordinates is preserved in the image plane, and also in shape-space given that its parameter $\mathbf{X}$ is defined to be linear.

Signal phase

So far, the argument about model order has concentrated on the power spectrum of a signal. This represents the amplitude of the signal spectrum but neglects its phase. Phase is also important but is less amenable to graphical arguments about goodness of fit. In fact, attempting to fit also the phase distribution of the lip-motion signal inevitably causes the power spectrum fit to deteriorate somewhat. This is evident later, in chapter 11, where an automatic fitting algorithm is developed. It fits models directly to signals, rather than to their power spectra, and therefore is bound to take both amplitude and phase into account.

9.5 Second-order AR processes in shape-space

A second-order AR process in shape-space is a natural extension of the first-order process (9.7). It has the form

$$\mathbf{X}(t_k) - \overline{\mathbf{X}} = A_2(\mathbf{X}(t_{k-2}) - \overline{\mathbf{X}}) + A_1(\mathbf{X}(t_{k-1}) - \overline{\mathbf{X}}) + B_0\mathbf{w}_k \tag{9.15}$$

in which A_2, A_1 and B_0 are all $N_X \times N_X$ matrices. In a second-order process, the shape-vector at a given time depends on two previous time-steps, rather than just one as in the first-order process. It can be expressed more compactly by defining a "state-vector"

$$\mathcal{X}(t_k) = \begin{pmatrix} \mathbf{X}(t_{k-1}) \\ \mathbf{X}(t_k) \end{pmatrix} \tag{9.16}$$

and then writing

$$\mathcal{X}(t_k) - \overline{\mathcal{X}} = A(\mathcal{X}(t_{k-1}) - \overline{\mathcal{X}}) + B\mathbf{w}_k \tag{9.17}$$

where

$$A = \begin{pmatrix} 0 & I \\ A_2 & A_1 \end{pmatrix}, \quad \overline{\mathcal{X}} = \begin{pmatrix} \overline{\mathbf{X}} \\ \overline{\mathbf{X}} \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} 0 \\ B_0 \end{pmatrix}. \tag{9.18}$$

It is straightforward to show that (9.17) is precisely equivalent to (9.15). Replacing the shape-vector $\mathbf{X}$ by a state-vector $\mathcal{X}$ of double size allows the notation for a first-order process to continue to be used, albeit with some restrictions (9.18) on the form of the coefficient matrices A and B.

The second-order state $\mathcal{X}$ has mean and covariance

$$\hat{\mathcal{X}}(t_k) = \mathcal{E}[\mathcal{X}(t_k)] \quad \text{and} \quad \mathcal{P}(t_k) = \mathcal{V}[\mathcal{X}(t_k)] \tag{9.19}$$

where $\mathcal{P}(t_k)$ is a $2N_X \times 2N_X$ matrix that is naturally decomposed into submatrices:

$$\mathcal{P}(t_k) = \begin{pmatrix} P''(t_k) & P'(t_k)^T \\ P'(t_k) & P(t_k) \end{pmatrix} \tag{9.20}$$

in which

$$\begin{aligned} P''(t_k) &= \mathcal{E}[(\mathbf{X}(t_{k-1}) - \hat{\mathbf{X}}(t_{k-1}))(\mathbf{X}(t_{k-1}) - \hat{\mathbf{X}}(t_{k-1}))^T] \\ P'(t_k) &= \mathcal{E}[(\mathbf{X}(t_k) - \hat{\mathbf{X}}(t_k))(\mathbf{X}(t_{k-1}) - \hat{\mathbf{X}}(t_{k-1}))^T] \\ P(t_k) &= \mathcal{E}[(\mathbf{X}(t_k) - \hat{\mathbf{X}}(t_k))(\mathbf{X}(t_k) - \hat{\mathbf{X}}(t_k))^T]. \end{aligned} \tag{9.21}$$

The most interesting of these is $P(t_k)$ which represents covariance in shape-space at time t_k. The other submatrices need to be "carried" by $\mathcal{P}$ simply in order to allow second-order propagation of $P(t_k)$. (Note that, as a consequence of second-order propagation, $P''(t_k) = P(t_{k-1})$.) Propagation is governed by mean-state and covariance equations for the second-order model, by analogy with the first-order case (9.8) and (9.10):

$$\hat{\mathcal{X}}(t_k) - \overline{\mathcal{X}} = A(\hat{\mathcal{X}}(t_{k-1}) - \overline{\mathcal{X}}) \tag{9.22}$$

and

$$\mathcal{P}(t_k) = A\mathcal{P}(t_{k-1})A^T + BB^T. \tag{9.23}$$

A steady-state $\mathcal{P}_\infty$ can be computed as before, by iterating (9.23) to convergence. Its lower-right submatrix P_∞ then represents the covariance of the Gaussian envelope in shape-space for the second-order Brownian process.

The AR process (9.16) is defined over discrete time, but can in fact be regarded as a regular sampling of an underlying continuous-time process. In the next section this discrete–continuous correspondence maps damped harmonic motion, described in continuous time, into discrete time. It serves as a "synthetic" predictive dynamical model, used in tracking in the next chapter. The correspondence is also useful in the reverse direction, to decompose and interpret an AR process in physical terms. This will be important in chapter 11 where models are learned from training sequences and learned coefficients A and B are interpreted as collections of damped harmonic oscillators, each associated with a vibrational mode — effectively a one-dimensional shape-subspace. Relevant details of the discrete–continuous correspondence are set out, for completeness, in appendix B.2.

9.6 Setting dynamical parameters

The best dynamical models for tracking, in the sense of being most appropriately tuned to expected motions, are obtained by learning and this is discussed fully in chapter 11. Until such learned models are available, we have to be content to use default models, synthesised by hand to match general, intuitive expectations about the motions to be observed. These expectations are addressed partly by the static constraints embodied in the choice of shape-space, and this has been dealt with in earlier chapters. Dynamical characteristics must also be specified in order to define fully an operational tracker (see the next chapter) which could be regarded as an end-product in its own right. Alternatively, the tracker may be applied just once, to capture a training sequence which is then used to learn a more refined shape-space via PCA (chapter 8) and more refined dynamics (chapter 11).

This section offers a systematic approach to synthesising dynamical models. The synthetic models are based on harmonic oscillators driven by spatially homogeneous noise. Shape-space is decomposed into natural subspaces and a stochastic harmonic oscillator is set up in each subspace.

Stochastic, harmonic motion in one dimension

First, to describe the harmonic oscillator which is the building block for synthesised models, an oscillation with damping rate β and frequency of oscillation f is expressed discretely by

$$A = \begin{pmatrix} 0 & 1 \\ a_2 & a_1 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} 0 \\ b_0 \end{pmatrix} \tag{9.24}$$

where

$$a_2 = -\exp(-2\beta\tau) \quad \text{and} \quad a_1 = 2\exp(-\beta\tau)\cos(2\pi f\tau). \tag{9.25}$$

(The derivation of these relations is given below.) A practically important special case, is *critical damping* in which $f = 0$, particularly suitable for motions which are not expected to be oscillatory; that leaves just $1/\beta$ to be specified, as a characteristic time-scale for the motion. Note that $f = 0$ implies the constraint on discrete parameters that $-a_1^2 = 4a_2$. Provided $\beta > 0$, the process is stable and has a steady-state distribution with finite spatial variance. To obtain a desired steady-state variance $\overline{\rho}^2$, choose

$$b_0 = \overline{\rho}\sqrt{1 - a_2^2 - a_1^2 - 2\frac{a_2 a_1^2}{1 - a_2}}. \tag{9.26}$$

(This formula comes directly from the steady-state solution of the covariance equation (9.23).) A more intuitive relationship is obtained by taking the "continuous time" limit that $\beta\tau \ll 1$, (an assumption that holds in most cases of practical interest):

$$b_0 = \overline{\rho}\left[2(\beta\tau)^{1/2}\sin(2\pi f\tau)\right]. \tag{9.27}$$

Constant-velocity model

A further sub-case of harmonic motion, useful particularly for translational motion, is the *constant-velocity* model in which $\beta = f = 0$, so that $a_2 = -1$ and $a_1 = 2$. As expected, for constant-velocity parameters the formulae (9.25) and (9.26) give $b_0 = 0$ or, equivalently, for any $b_0 > 0$ the variance $\overline{\rho}^2$ is unbounded. Given that there is no steady state, $\overline{\rho}$ cannot be used to characterise the stochastic component of the model. Instead, a rate of growth parameter γ_0 is specified. It can be shown that, asymptotically, the root-mean-square search-region diameter grows semicubically:

$$\overline{\rho}(t) = \gamma_0 t^{3/2} \quad \text{where} \quad b_0 = \gamma_0 \tau^{3/2}. \tag{9.28}$$

Harmonic motion in shape-space

A straightforward way to construct a dynamic model for a shape-space S_X is to extend the one-dimensional harmonic motion just described, to the entire space. This is done by choosing

$$A_2 = a_2 I_{N_X}, \quad A_1 = a_1 I_{N_X} \quad \text{and} \quad B_0 = \frac{b_0}{\sqrt{N_X}} \mathcal{H}^{-\frac{1}{2}}$$

where a_2, a_1 and b_0 are chosen as above. The resulting A-matrix then represents "degenerate" modes, N_X independent motions, with common frequency and damping constants, that span the shape-space. The driving Brownian noise has the usual norm-squared density, giving spatial uniformity and isotropy but subject to the constraints of the shape-space. The steady-state covariance $\mathcal{P}_\infty$ can be shown from (9.23) to be

$$P_\infty = \frac{\overline{\rho}^2}{N_X} \mathcal{H}^{-1}, \tag{9.29}$$

where $\overline{\rho}$ and b_0 are related as in (9.26). This represents a Gaussian envelope with mean $\overline{\mathbf{X}}$ and root-mean-square displacement $\overline{\rho}$ along the curve.

Partitioned harmonic motion across shape-subspaces

Finally, it is usually desirable to partition shape-space into several subspaces and apply different harmonic models to each. Earlier, affine space was partitioned into translation and deformation components using projection matrices E^s and E^d, and a first-order model (9.4) on page 192, defined across the partitioned space. This was chosen to allow relatively free translation but tightly constrained deformation of shape. A partitioned second-order model can similarly be specified by defining A and

B matrices as follows:

$$\begin{aligned} A_2 &= a_0^s E^s + a_0^d E^d, \quad A_1 = a_1^s E^s + a_1^d E^d \\ B_0 &= \left(\frac{b_0^s}{\sqrt{N_s}} E^s + \frac{b_0^d}{\sqrt{N_d}} E^d \right) \mathcal{H}^{-\frac{1}{2}}, \end{aligned} \tag{9.30}$$

where N_s and N_d are the dimensions of the shape-subspaces. Parameters a_0^d, a_1^d and b_0^d are set for appropriate frequency f^d, damping rate β^d and average displacement $\overline{\rho}^d$ for deformation, using (9.26) above, and similarly for translation. Any desired number of partitions of shape-space can be chosen, and partitioned dynamics defined additively as above. Variances of the Gaussian envelopes are additive so that, in the two-component case above,

$$\overline{\rho}^2 = \sqrt{(\overline{\rho}^s)^2 + (\overline{\rho}^d)^2} \tag{9.31}$$

is the root-mean-square displacement in model as a whole.

Examples

Examples of dynamical models set up by hand are given here. The first is for vehicles on a motorway, as in the traffic-monitoring application described in chapter 1. Sample trajectories are displayed in figure 9.13. Translational dynamics are constant-velocity, with characteristic semicubical growth of the region of positional uncertainty, reaching 35 pixels (RMS) after one second. Affine deformation is set to vary slowly, over a 10-second time-scale, to allow the shrinkage of the template under perspective scaling, as vehicles recede into the distance. Ideally this shrinkage should be coupled with the translational motion in the dynamical model itself, rather than simply in the initial conditions, and limited to scaling rather than allowing all affine deformations. That is rather more elaborate than can reasonably be programmed by hand. What can be done is to use the model above as a predictor in a tracker that is just capable of gathering a training set. The training set can then be used in the learning algorithm to be described in chapter 11 to build automatically a model that incorporates the appropriate couplings.

A second example of a hand-programmed dynamical model is designed to represent the motion of a dancing girl, as in figure 9.14. This time the model is explicitly oscillatory, with independent dynamics for horizontal and vertical motion. As with the example of traffic above, a dynamical model such as this is sufficient for use as a predictor in a tracker that can capture a training set, which is then used to learn

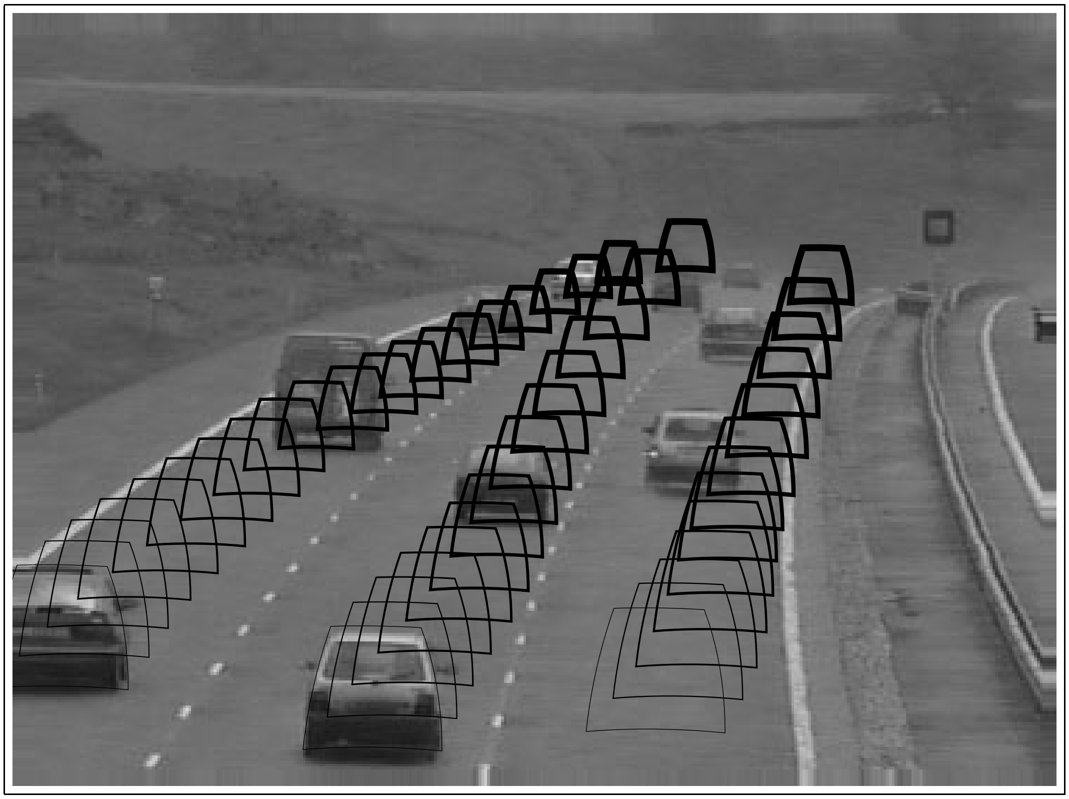


Figure 9.13: Dynamical model for traffic. *Three simulated trajectories for an ARP set up to represent the traffic pattern, each initialised with a velocity in shape-space corresponding to translation along the road, coupled with shrinkage consistent with perspective scaling. Translational parameters are* $\beta = f = 0$ *(constant-velocity) with* $\gamma_0 = 35\,\text{pixel.s}^{-3/2}$*, allowing the paths to veer over longer times. Affine deformations are critically damped* $(f = 0)$ *with a long time-constant* $(1/\beta = 10\,s)$*, and largely deterministic* $(\overline{\rho} = 2\,pixels)$*, allowing the outline to shrink steadily. Contours are plotted here at intervals of* 200 ms.

automatically an auto-regressive process model that is more specifically tuned to the observed motions.

Derivation of simple harmonic oscillator coefficients a_2, a_1, in terms of f, β as in (9.25): given that a_2, a_1 are to be chosen to correspond to the damped exponential $\exp -\beta t \exp \pm 2\pi f t$, over a time-step τ, the eigenvalues of A in (9.24) must be $\exp -\beta\tau \exp \pm 2\pi f \tau$, whose product

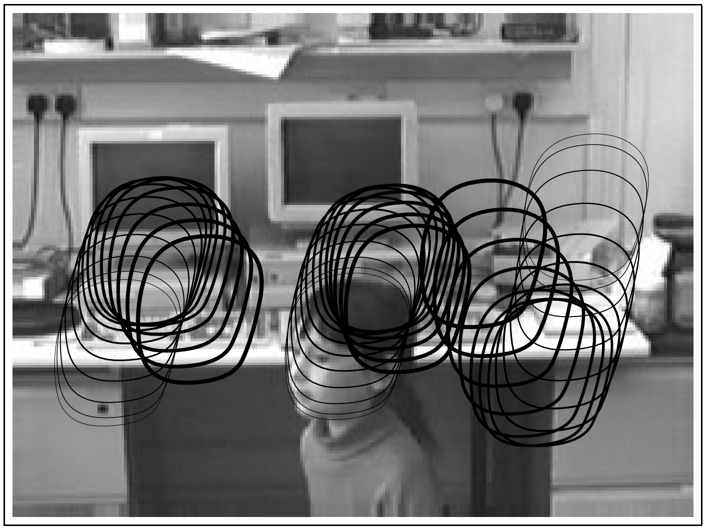


Figure 9.14: Dynamical model for a dancer. *Three simulated trajectories, each of 0.84* s *duration, are shown of an ARP model in a shape-space of translations. Horizontal oscillation is slow (*$\beta = 0.2\,\mathrm{s}^{-1},\ f = 0.2\,\mathrm{Hz}$*) and broadly distributed (*$\overline{\rho} = 300\,\mathrm{pixels}$*), representing the motion of the dancer to and fro, across the room. Vertical oscillation is faster (*$\beta = 0.5\,\mathrm{s}^{-1},\ f = 1\,\mathrm{Hz}$*) and more tightly distributed (*$\overline{\rho} = 100\,\mathrm{pixels}$*), representing the bobbing motion of the head.*

and sum, respectively the determinant and trace of A, give

$$a_2 = -\det A = -\exp(-2\beta\tau) \quad \text{and} \quad a_1 = \operatorname{tr} A = 2\exp(-\beta\tau)\cos(2\pi f\tau),$$

as required.

Bibliographic notes

Dynamical models exploit “coherence of motion” — a term coined by Yuille and Grzywacz (Yuille and Grzywacz, 1988) to refer to the spatial continuity of motion across an image, and used here to refer both to spatial but especially also temporal continuity. Dynamical models were based on the Markov process. This is a development of the Markov Chain (see for instance (Rabiner and Bing-Hwang, 1993)) in a continuous-valued form, and is a core tool in control theory (Astrom and Wittenmark, 1984). Markov processes can be expressed either in continuous or discrete time and the relation between the two is explained in (Gelb, 1974). Continuous-time stochastic processes form the basis of the arguments about physical realisability of models in the chapter, and a detailed treatment of continuous-time stochastic processes is given in (Astrom, 1970). Furthermore continuous-time analysis establishes the semicubical growth of the search region under constant-velocity dynamics (Blake et al., 1993).

Markov processes can also be viewed in spectral terms as the output of certain linear systems driven by noise (Papoulis, 1991), and often displaying resonances in its spectral power distribution. The use of resonant modes in visual motion modelling was first developed by Pentland and Horowitz (1991) and applied to spline contour models by Baumberg and Hogg (1994). In both cases the dynamical models are purely deterministic, essentially the deterministic component of the stochastic models described in the chapter. Such deterministic dynamical models describe distributed, massive systems (Landau and Lifshitz, 1972), and can be derived using "Lagrangian analysis." Second order differential equations, expressed in terms of positions, velocities and accelerations, are obtained from the functions of position and velocity that specify the potential energy and power dissipation of a system. A useful introduction is given in (Terzopoulos and Szeliski, 1992).

Chapter 10

Dynamic contour tracking

In the previous chapter, dynamical models were characterised by a second-order state density $p(\mathcal{X}(t))$, evolving temporally, and representing the *prior* distribution for the state $\mathcal{X}$ at each time t. In this chapter, both the prior dynamical model and visual measurements are to be taken into account. The result is a fusion of information, both prior and observational, as was set out in chapter 8 for single images, but done now for image sequences, to track motion.

The natural mechanism for temporal fusion, when distributions are Gaussian, is the Kalman filter. It computes the evolution of the Gaussian density for the state of the tracked object, as figure 10.1 illustrates. A simple practical example consists of a planar affine shape-space, based on a hand-shaped template, with constant-velocity dynamics, which is capable of tracking motion of an outstretched hand (figure 10.2). All results in this chapter show tracking tasks that can be performed in real time on a desktop workstation.

10.1 Temporal fusion by Kalman filter

Observation history

The Kalman filter maintains a Gaussian distribution of the state

$$\mathcal{X}(t_k) \sim \mathcal{N}(\hat{\mathcal{X}}(t_k), \mathcal{P}(t_k))$$

given both the prior dynamical model for $\mathcal{X}(t_k)$ *and* the measurement history $\underline{\mathbf{Z}}(t_k)$:

$$\underline{\mathbf{Z}}(t_k) = (\mathbf{Z}(t_1), \ldots, \mathbf{Z}(t_k)) \tag{10.1}$$

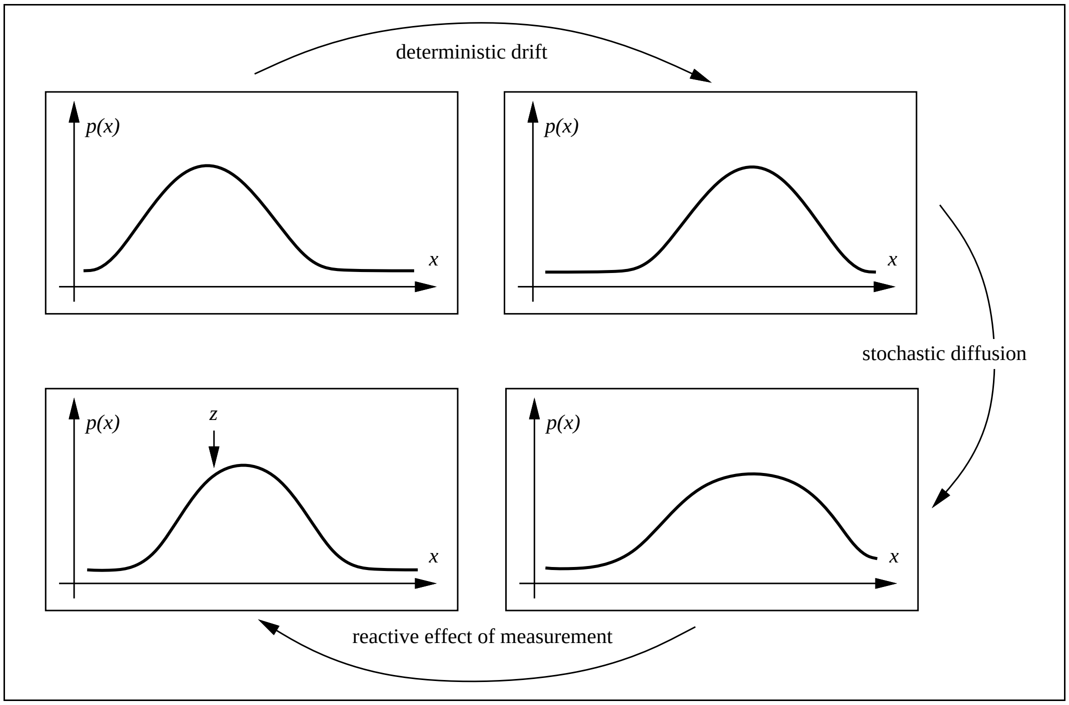


Figure 10.1: Kalman filter as density propagation. *In the case of Gaussian prior, process and observation densities, and assuming linear dynamics, the state density propagates as a Gaussian represented completely by its evolving (multi-variate) mean and variance.*

where $\mathbf{Z}(t_k)$ is the version at time $t = t_k$ of the aggregated observation $\mathbf{Z}$ that is produced by the recursive image-curve fitting algorithm of chapter 6 (figure 6.7 on page 127). As in that static algorithm, so also now in the dynamic case, the statistical information associated with $\mathbf{Z}(t_k)$ for estimating $\mathbf{X}$ is $S(t_k)$ (S in the fitting algorithm) and $\mathbf{Z}(t_k)$ is an unbiased estimate of $S(t_k)\mathbf{X}(t_k)$ (rather than of $\mathbf{X}(t_k)$ itself).

In the previous chapter, $\hat{\mathcal{X}}(t_k)$ and $\mathcal{P}(t_k)$ were defined (in (9.19) on page 204) to be the *prior* mean and covariance of the state. Now $\hat{\mathcal{X}}$ and $\mathcal{P}$ are redefined as the *posterior* mean and variance

$$\hat{\mathcal{X}}(t_k) = \mathcal{E}[\mathcal{X}(t_k)|\underline{\mathbf{Z}}(t_k)] \quad \text{and} \quad \mathcal{P}(t_k) = \mathcal{V}[\mathcal{X}(t_k)|\underline{\mathbf{Z}}(t_k)], \tag{10.2}$$

conditioned on the measurement history $\underline{\mathbf{Z}}(t_k)$. The principle behind the evolution of $\hat{\mathcal{X}}(t_k)$, together with $\mathcal{P}(t_k)$, is straightforward. The estimate at time t_{k-1} is propagated

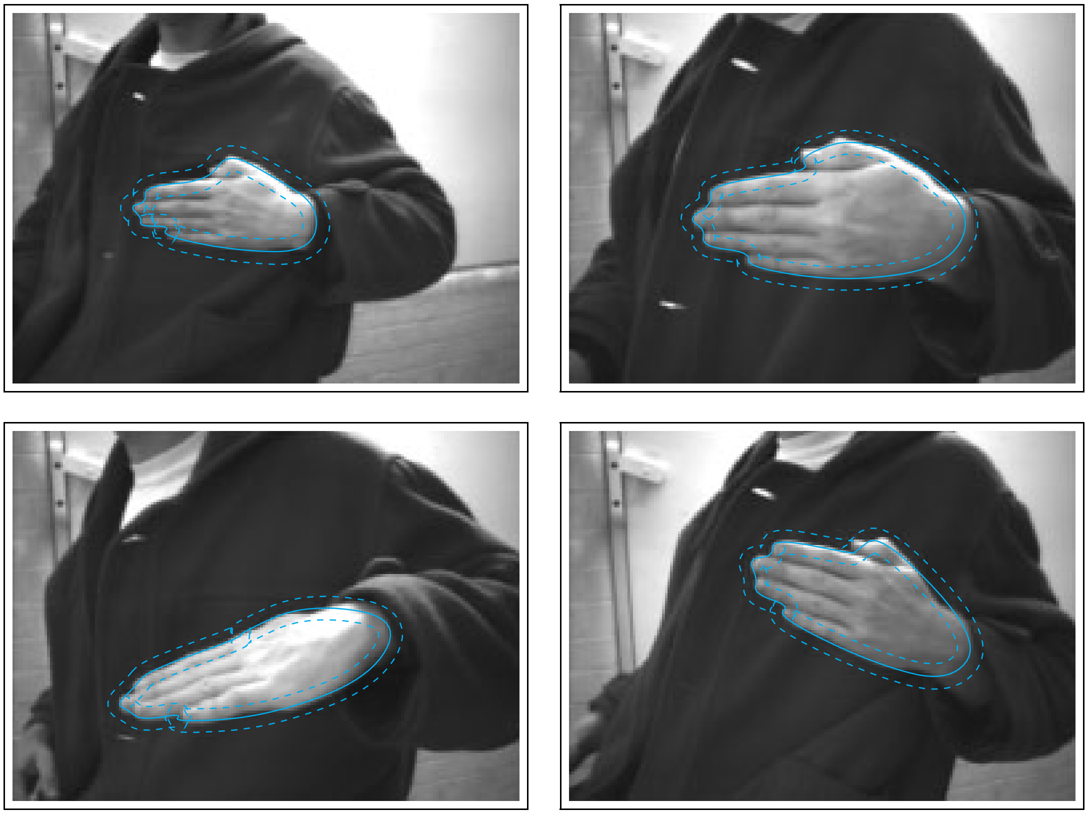


Figure 10.2: Hand tracking in planar affine shape-space. *A planar affine shape-space with constant-velocity dynamics is sufficient to track movements of an outstretched hand at modest speed.*

to time t_k in the two standard steps of the Kalman filter: prediction, and assimilation of observations. This engages all of the probabilistic machinery developed so far in the book: prediction is based on the dynamical models of the previous chapter; assimilation of observations is based on information weighting as in chapters 6 and 8.

Prediction

A single time-step of the dynamical model ((9.22) on page 205) is applied to $\hat{\mathcal{X}}(t_{k-1})$ to obtain a predicted state $\tilde{\mathcal{X}}(t_k)$:

$$\begin{aligned} \tilde{\mathcal{X}}(t_k) - \overline{\mathcal{X}} &= A(\hat{\mathcal{X}}(t_{k-1}) - \overline{\mathcal{X}}) \\ \tilde{\mathcal{P}}(t_k) &= A\mathcal{P}(t_{k-1})A^T + BB^T. \end{aligned} \tag{10.3}$$

Assimilation by information weighting

In its most direct form, assimilation computes an information weighted mean between the prediction and the latest measurement:

$$\begin{aligned} \mathcal{P}(t_k) &= \left(\tilde{\mathcal{P}}^{-1}(t_k) + H^T S(t_k) H\right)^{-1} \\ &\text{and} \\ \hat{\mathcal{X}}(t_k) &= \tilde{\mathcal{X}}(t_k) + \mathcal{P}(t_k) H^T \mathbf{Z}(t_k) \end{aligned} \tag{10.4}$$

where the matrix

$$H = \begin{pmatrix} 0 & I \end{pmatrix} \tag{10.5}$$

maps $\mathcal{X}(t_k) \rightarrow \mathbf{X}(t_k)$ from state-space into shape-space.

It is clear that in the first of these equations, information is summed and that in the second, information weighting is applied. However, this assimilation procedure is somewhat inefficient because of the need to invert large $(2N_X \times 2N_X)$ matrices. Fortunately the Kalman filtering canon offers a more efficient alternative in which only a single $N_X \times N_X$ matrix inversion is required at each time-step.

Assimilation by Kalman gain

An exactly equivalent algorithm for assimilation, free of inversions of the full state covariance $\mathcal{P}$, is based on a "Kalman gain" matrix $\mathcal{K}$, as follows:

$$\begin{aligned} \mathcal{K}(t_k) &= \tilde{\mathcal{P}}(t_k) H^T \left(S(t_k) H \tilde{\mathcal{P}}(t_k) H^T + I\right)^{-1} \\ \hat{\mathcal{X}}(t_k) &= \tilde{\mathcal{X}}(t_k) + \mathcal{K}(t_k)\mathbf{Z}(t_k) \\ \mathcal{P}(t_k) &= \left(I - \mathcal{K}(t_k) S(t_k) H\right) \tilde{\mathcal{P}}(t_k). \end{aligned} \tag{10.6}$$

Note that the only matrix inversion occurs in the expression for the Kalman gain and involves an $N_X \times N_X$ matrix, as promised. A particular form has been used here for the Kalman gain such that if the aggregated measurement $\mathbf{Z}(t_k)$ is deficient so that $S(t_k)$ has less than full rank, the assimilation step is still well-defined and free of singularities. In fact, even if measurement fails altogether so that $S(t_k) = 0$ and $\mathbf{Z}(t_k) = \mathbf{0}$, the assimilation step simply becomes $\hat{\mathcal{X}}(t_k) = \tilde{\mathcal{X}}(t_k)$, accepting the prediction without modification.

Block decomposition

Finally, the algorithm above can be expressed most efficiently, using submatrix decomposition of the state-space covariance $\mathcal{P}$ into submatrices P, P' and P'', as earlier in (9.20) on page 204. In addition, the Kalman gain is decomposed into two $N_X \times N_X$ submatrices:

$$\mathcal{K} = \begin{pmatrix} K' \\ K \end{pmatrix}. \tag{10.7}$$

This allows the special form of the dynamical matrix A and the symmetry of $\mathcal{P}$ to be exploited. The full algorithm is given in figure 10.3. It consists of one predict–observe–assimilate cycle for each time-step. The predicted shape $\tilde{\mathbf{X}}(t_k)$ is used as the basis for obtaining the aggregated measurement vector $\mathbf{Z}(t_k)$. It replaces, in steps 2 and 4 of the fitting algorithm of figure 6.7 on page 127, the shape-vector $\overline{\mathbf{X}}$ which defined the estimated curve $\overline{\mathbf{r}}(s)$ from which image features ν_i are measured. Finally, the estimated curve shape $\mathbf{X}(t_k)$ is computed, as required. Other variables $P(t_k)$, $P'(t_k)$, $P''(t_k)$ and $\hat{\mathbf{X}}'(t_k)$, computed in the assimilation step, are carried forward for use in prediction at the following time-step.

Initial conditions

The algorithm of figure 10.3 sets out the formation of the estimate at time t_k from the one at time t_{k-1}. All that remains to complete the algorithm is to specify initial conditions at time t_0. There are various natural ways to do this corresponding to various possible assumptions about the initial state of the object to be tracked. In general, values of $\hat{\mathbf{X}}(t_0)$ and $\mathcal{P}(t_0)$ are set to reflect the prior distribution for object state. The state distribution specifies the distributions of configurations at times t_0, t_{-1}, that is $\mathbf{X}(t_0)$ and $\mathbf{X}'(t_0)$ (recall $\mathbf{X}'(t_0) \equiv \mathbf{X}(t_{-1})$). Alternatively, it can be

Algorithm for propagation over one time-step

Predict

$$
\begin{aligned}
\tilde{P}''(t_k) &= P(t_{k-1}) \\
\tilde{P}'(t_k) &= A_2 {P'}^T(t_{k-1}) + A_1 P(t_{k-1}) \\
\tilde{P}(t_k) &= A_2 P''(t_{k-1}) A_2^T + A_1 P'(t_{k-1}) A_2^T \\
&\quad + A_2 {P'}^T(t_{k-1}) A_1^T + A_1 P(t_{k-1}) A_1^T + B_0 B_0^T
\end{aligned}
$$

$$
\begin{aligned}
\tilde{\mathbf{X}}'(t_k) &= \hat{\mathbf{X}}(t_{k-1}) \\
\tilde{\mathbf{X}}(t_k) &= A_2 \hat{\mathbf{X}}'(t_{k-1}) + A_1 \hat{\mathbf{X}}(t_{k-1}) + (I - A_2 - A_1)\overline{\mathbf{X}}.
\end{aligned}
$$

Measure

Apply the algorithm of figure 6.7 on page 127, to the image at time t_k using $\tilde{\mathbf{X}}(t_k)$ as estimated contour from which normals are cast. Use $\tilde{P}(t_k)$ as the value of $\overline{P}$ to compute validation-gate width (figure 8.6 on page 174). Obtain the aggregated observation vector $\mathbf{Z}$ and information matrix S, denoted here as $\mathbf{Z}(t_k)$ and $S(t_k)$.

Assimilate

$$
\begin{aligned}
K'(t_k) &= \tilde{P}'(t_k) \left[S(t_k) \tilde{P}(t_k) + I \right]^{-1} \\
K(t_k) &= \tilde{P}(t_k) \left[S(t_k) \tilde{P}(t_k) + I \right]^{-1}
\end{aligned}
$$

$$
\begin{aligned}
P''(t_k) &= \tilde{P}''(t_k) - K'(t_k) S(t_k) \tilde{P}'(t_k) \\
P'(t_k) &= \tilde{P}'(t_k) - K(t_k) S(t_k) \tilde{P}'(t_k) \\
P(t_k) &= \tilde{P}(t_k) - K(t_k) S(t_k) \tilde{P}(t_k)
\end{aligned}
$$

$$
\begin{aligned}
\hat{\mathbf{X}}'(t_k) &= \tilde{\mathbf{X}}'(t_k) + K'(t_k) \mathbf{Z}(t_k) \\
\hat{\mathbf{X}}(t_k) &= \tilde{\mathbf{X}}(t_k) + K(t_k) \mathbf{Z}(t_k)
\end{aligned}
$$

Figure 10.3: Second-order Kalman filter for image sequences.

thought of as specifying a prior distribution for initial position $\mathbf{X}(t_0)$ and velocity $\mathbf{V}(t_0) = \dot{\mathbf{X}}(t_0)$.

Known, static, initial position. This is an appropriate assumption for interactive applications such as the hand-mouse of figure 1.16 on page 20 in which the hand might typically be inserted into a marked outline with shape $\mathbf{X}_0$, and held still while tracking is switched on. In that case, suitable initial conditions are given by the *closed-loop* steady state of the filter. This can be obtained most simply by running the tracking algorithm with constant "artificial" measurements $S(t_k) = S,\ \mathbf{Z}(t_k) = S\mathbf{X}_0$ in which S takes the value for a full set of successful image measurements in the curve-fitting algorithm. Under these circumstances, values of $P(t_\infty),\ P'(t_\infty),\ P''(t_\infty)$, and of $\hat{\mathbf{X}}(t_\infty), \mathbf{X}'(t_\infty)$ should settle to a steady state. This serves as a suitable initial condition once tracking is switched on by suspending the artificial measurements and allowing real image measurements to flow in.

Unknown initial position. If initial position is unknown, then the only available prior information is what is embodied in the dynamical model itself. Then the assumption is that, at switch-on, the object may be anywhere within the Gaussian envelope implied by the model. It amounts to setting initial conditions to the *open-loop* steady state, determined by running the dynamical model in the absence of measurement. This is obtained by running the tracker on artificial measurements, this time with $S(t_k) = 0,\ \mathbf{Z}(t_k) = \mathbf{0}$, until tracking is switched on as before. Note that the method can be used only if the AR process has a steady state and that excludes "constant velocity" models (see previous chapter, page 206).

Known initial velocity. In many applications a prior estimate of the initial velocity of the object is available, at least over a subspace of the shape-space. For example, observing plants on the ground from a moving tractor (figure 1.6 on page 10) or objects moving on a conveyor belt. In that case the estimated velocity is incorporated in a velocity offset vector $\mathbf{V}_0$ in shape-space and the initial state becomes:

$$\hat{\mathbf{X}}(t_0) = \mathbf{X}_0, \quad \hat{\mathbf{X}}'(t_0) = \mathbf{X}_0 - \mathbf{V}_0\tau.$$

with $\mathcal{P}$ initialised to the closed-loop steady-state value as before.

Alternative Kalman Filter

An alternative version of the Kalman filter is possible with an assimilation step that is equivalent to (10.6) but with a modified presentation. It is based on the alternative curve-fitting algorithm of chapter 6 (figure 6.10 on page 133), rather than the original recursive algorithm. The detailed implementation of the algorithm is omitted here. The comparison between the two algorithms, here in the dynamic case, is similar to that for the static case: for small N_X the original algorithm is more efficient, but as N_X increases the alternative algorithm becomes relatively more efficient.

Computational cost

The computational cost of the tracking algorithm is in fact $O(N_X^3)$ so that increasing the dimension N_X of shape-space carries a heavy cost penalty. The cost is comprised of two components. Prediction and assimilation (figure 10.3) incur a cost of $O(N_X^3)$ in the multiplication and inversion of the various $N_X \times N_X$ matrices. The generation of the aggregated measurements $\mathbf{Z}(t_k)$ also costs (argued in chapter 6) approximately $O(N_X^3)$.

10.2 Tracking performance

At last, all three components of a fully dynamic, active contour tracker are in place: visual measurement, the dynamic model and temporal filtering. The validation-gate mechanism of chapter 8, used in the recursive fitting algorithm (figure 8.6 on page 174) to acquire the aggregated observations, is crucial here. It relates the width of the search region to the position covariance P, now a time-varying quantity $\tilde{P}(t_k)$. Combining the three components in the algorithm of figure 10.3, and using a validation gate, produces a tracker that is robust in a number of respects.

Initialisation

One sense in which tracking is robust is that it is capable, in the initialisation phase, of "locking on" to an object: uncertainty in object position and shape is tolerated initially but rapidly resolved as the continuity of tracked motion is recognised. This is reflected in the collapsing search region in figure 10.4.

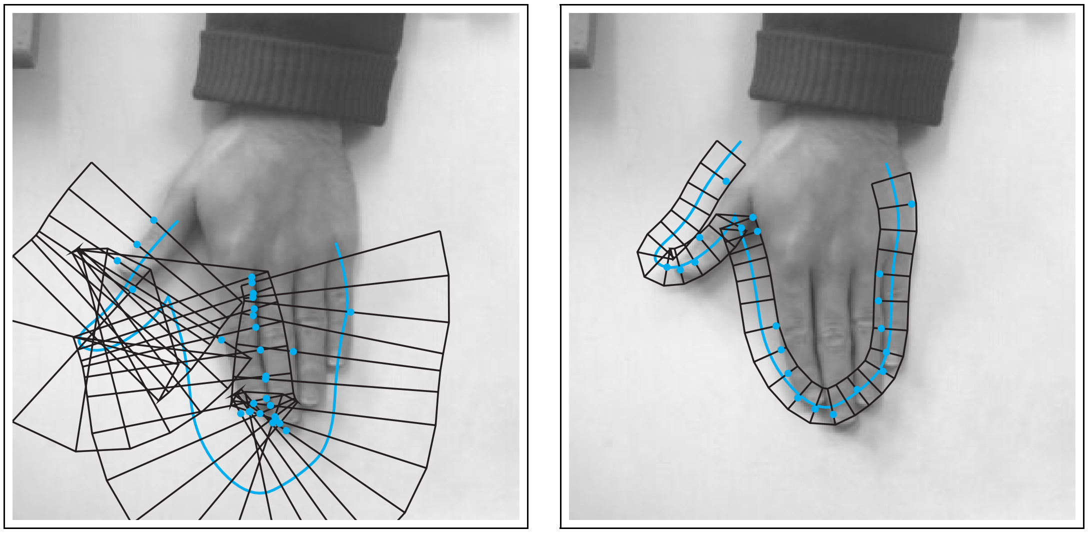
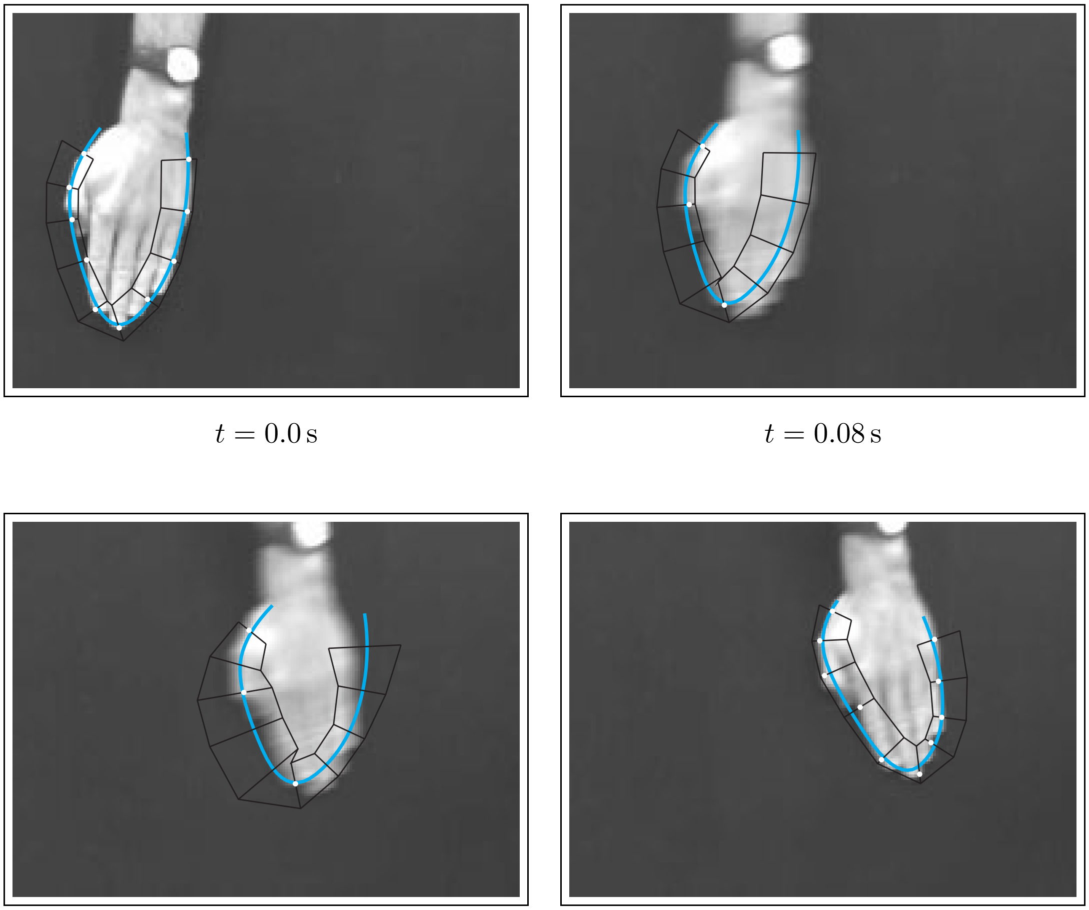


Figure 10.4: Locking on from the open-loop steady state. *The search region begins at its largest, reflecting initial uncertainty of location. It rapidly collapses as observations decrease the uncertainty.*

Adaptive search

If image observations fail along the contour, the search region expands over the affected segments until observations succeed again, as figure 10.5 shows. The extreme case is when observations fail along the entire length of the contour, in which case the search region regresses towards its open-loop steady state. Note the characteristically rapid recovery of tracking, in the figure, between $t = 0.16\,\mathrm{s}$ and $t = 0.20\,\mathrm{s}$. This is explained by the increased positional variance at $t = 0.16\,\mathrm{s}$ ($\tilde{P}(t_k)$ in figure 10.3), leading to a correspondingly increased Kalman gain ($K(t_k)$). Once back in the steady state, covariance decreases and Kalman gain also decreases so that the relative influence of the predictive dynamical model becomes stronger again. To summarise, there is an inverse relationship between spatial scale and temporal scale. In the special case of constant-velocity dynamics, it is an inverse square law; the time-scale for smoothing varies as the inverse square of the spatial extent of the search region. This inverse relationship can be observed even within one contour, as figure 10.6 shows.



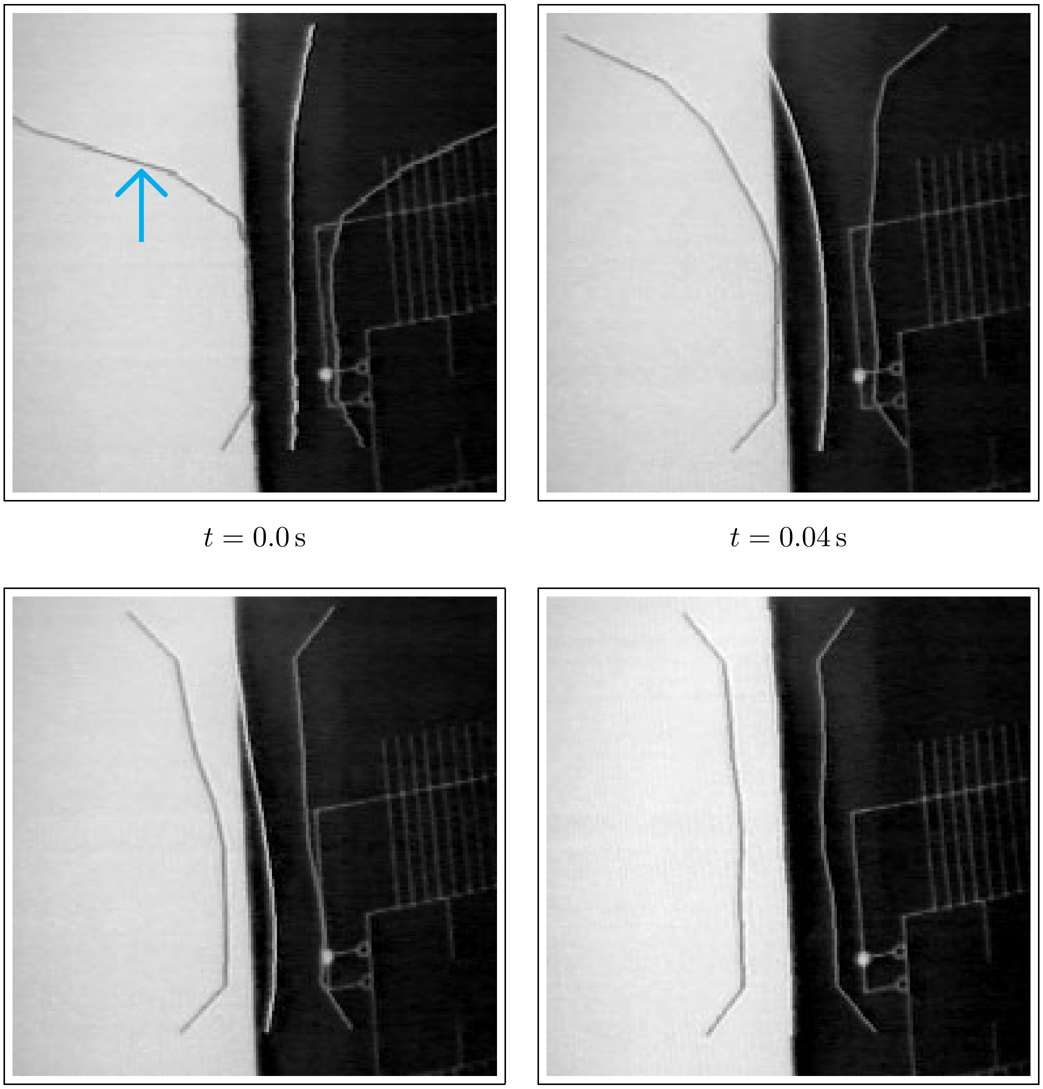
$t = 0.0\,\mathrm{s}$

$t = 0.08\,\mathrm{s}$

$t = 0.16\,\mathrm{s}$

$t = 0.20\,\mathrm{s}$

Figure 10.5: Recovery of lock. *As the hand begins to move, observations (white dots) succeed initially along all normals. As the accelerating hand leaves the dynamic contour lagging behind, measurements are lost and the search region automatically expands, reflecting increasing uncertainty. The expanded region now contains the edges of the hand and observations recover.*



$t = 0.0\,\mathrm{s}$

$t = 0.04\,\mathrm{s}$

$t = 0.08\,\mathrm{s}$

$t = 2.0\,\mathrm{s}$

Figure 10.6: Temporal scale varies inversely with spatial scale *The search region (bounded by grey lines, arrowed in the first frame) is initialised with varying width. Where the search region is wide, temporal scale is short, and the contour deflects more rapidly, as shown. (Figure by courtesy of R. Curwen.)*

Resistance to clutter and obscuration

Background clutter can disrupt tracking by distracting the observation process. The validation gate is helpful here, especially when the search region is narrow, because it excludes as much of the clutter as possible, as in figure 10.7. Similarly, the validation gate helps with loss of visibility, occurring either because the object passes partly out of the image or from partial obscuration when, for instance, a tracked person passes

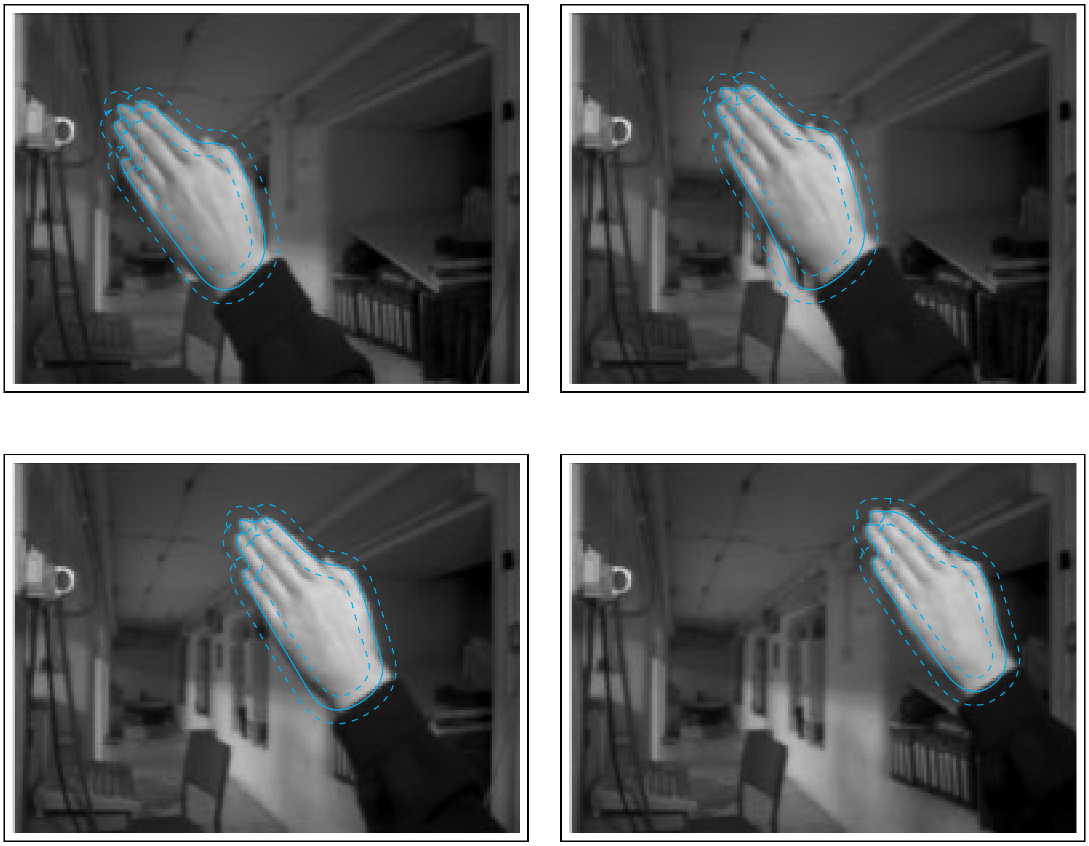


Figure 10.7: Resistance to clutter. *A hand accelerates across a cluttered background. Note that in the second frame the lower left corner of the contour is momentarily distracted by the chair but the disturbance is successfully filtered out over time. (Figure by courtesy of R. Curwen.)*

behind a lamp post. A good example is the tracking of fish viewed underwater, as in figure 10.8. Of course if the obscuring object is known, it can actually be *predicted* that measurements will fail over the corresponding image region. In that case attempts to measure ν_i along normals, in the curve-fitting algorithm, are suspended over the region that is predicted to be obscuring. Where the obscuring object has texture, which could generate false measurements, predicting the obscuration allows those false features to be suppressed.

Agile motion

Agility is a challenge because the continuity of motion is disrupted when sudden changes of speed and direction occur. The validation gate helps here also by signaling the loss of "lock" on the moving object and allowing the tracker to regress towards the "open-loop initialisation" state in which greater positional error is tolerated, as indicated by the widening of the search region. As recovery of lock proceeds, the tracker progressively returns to the steady state associated with continuous motion (figure 10.9). Figure 10.10 illustrates the robustness of agile tracking due the temporal variations of Kalman gain and search-region width.

10.3 Choosing dynamical parameters

This section outlines some design principles for building trackers. How can the parameters of a synthesised AR process of the sort described in section 9.6, and the observation parameters, be chosen to meet performance requirements? The AR parameters to specify are the frequency, damping rate and average displacement

$$f_i \, \mathrm{s}^{-1}, \quad \beta_i \, \mathrm{s}^{-1} \quad \text{and} \quad \overline{\rho}_i \, \text{pixels},$$

for each shape-subspace $\mathcal{S}_i$ (of dimension N_i) used. For the observation process, the parameter $\overline{\rho}_f$ must be specified, representing the average displacement error of image measurements along a curve.

AR parameters

In many applications it is reasonable to maintain the critical damping condition $f_i = 0$ so that there is a single, natural time-scale $1/\beta_i$ to be chosen for each space $\mathcal{S}_i$. In other

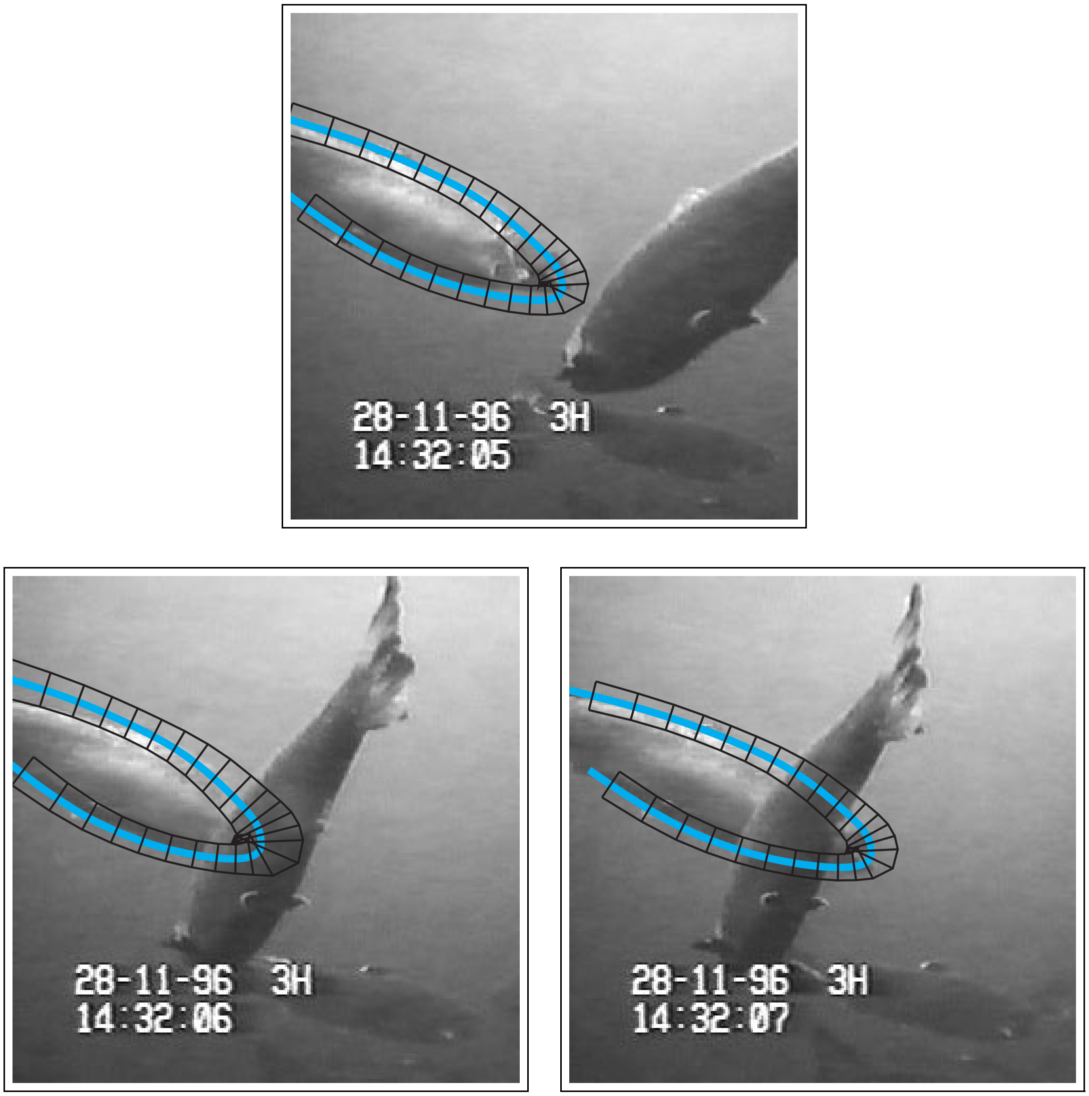


Figure 10.8: Expansion of the search region aids recovery from occlusion. *A tracked fish (top) passes behind another fish (left). The remaining successful observations, together with the dynamical model in shape-space, compensate for lost measurements around the head. Observations around the head resume as it re-emerges (right), one second later.*

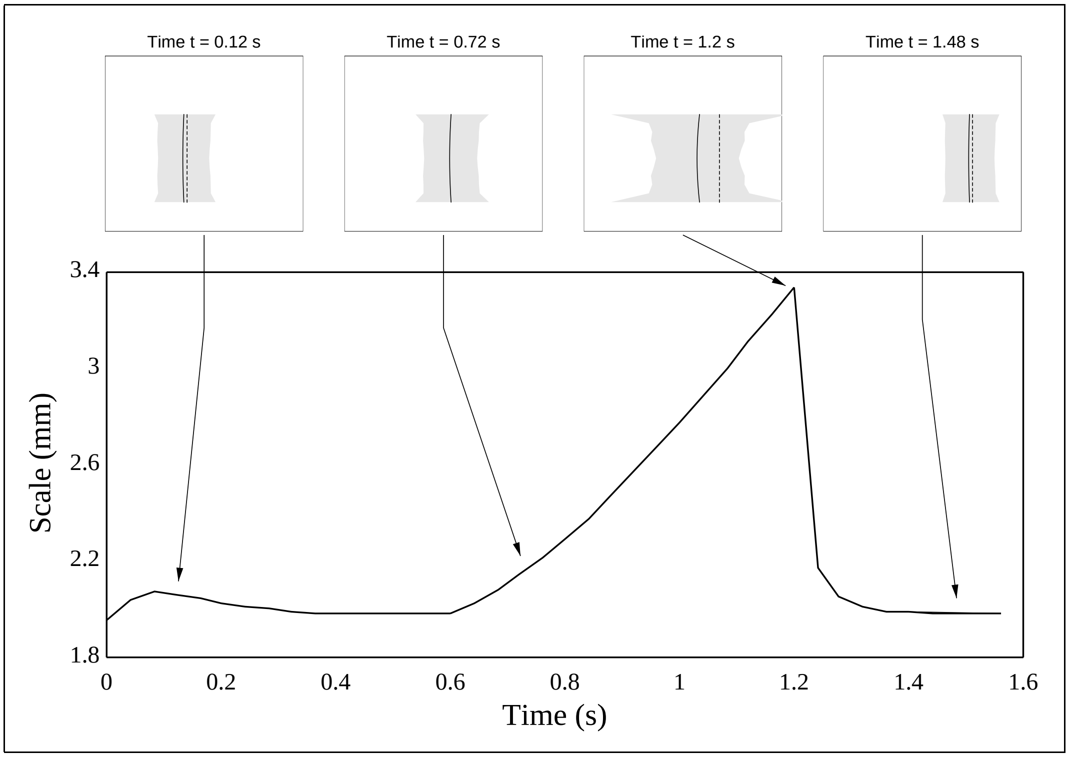
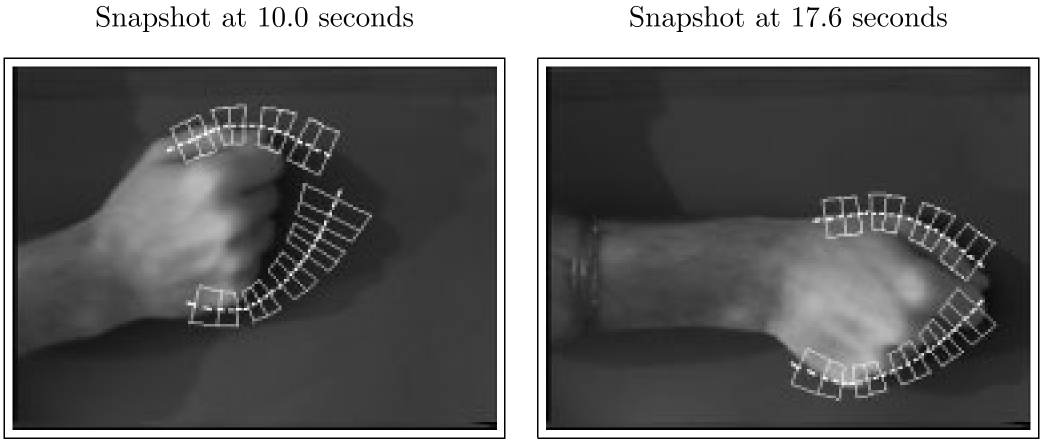


Figure 10.9: Loss and recovery of lock. *When an object falls outside its validation gate, owing to a sudden discontinuity of motion for example, the validation gate widens rapidly. Interestingly, in the case illustrated of a constant-velocity model, it can be shown that gate width grows in proportion to* $t^{3/2}$*. The expanding gate will therefore catch up with any object fleeing at a constant velocity. Once lock is recovered, the gate collapses rapidly. (Figure courtesy of Rupert Curwen.)*

applications, such as lip motion with its characteristic oscillations during speech, it is more appropriate to set $f_i > 0$ over shape-subspaces associated with lip deformation.

Once dynamical constants f_i and β_i are fixed, it remains to choose the average displacements $\overline{\rho}_i$. That fixes the open-loop search-region width, which is proportional to the average displacement $\overline{\rho}_o$, over the whole shape-space, under open-loop conditions:

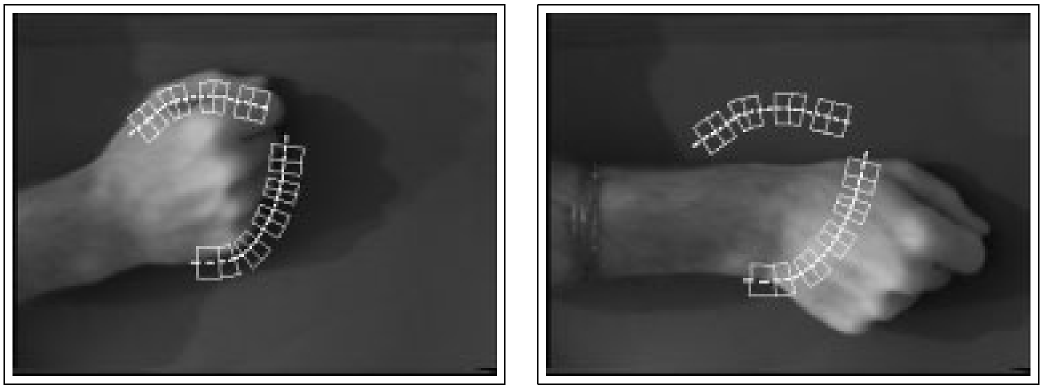
$$\overline{\rho}_o = \sqrt{\sum_i \overline{\rho}_i^2}.$$



Snapshot at 10.0 seconds

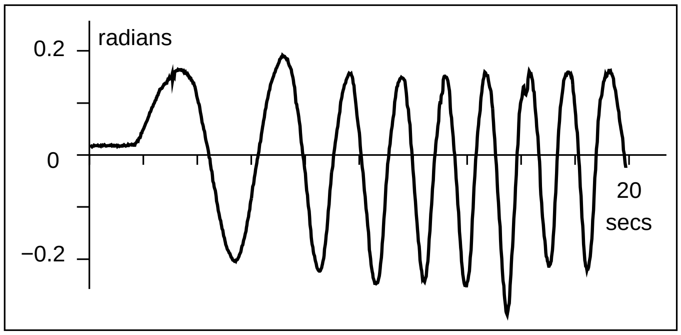
Snapshot at 17.6 seconds

Varying gain and search region

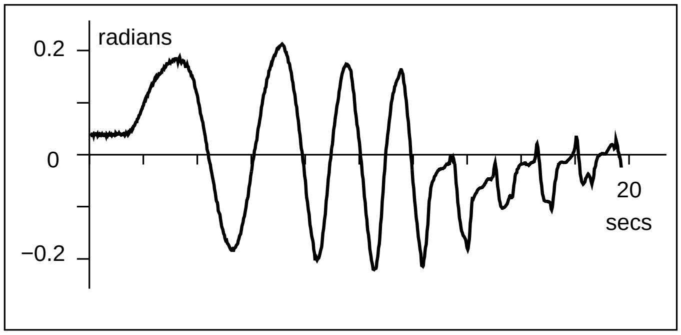


Fixed gain and search region

Figure 10.10: Validation gate strengthens tracking performance for agile motion. *The "chirp" test sequence used here, of 20 seconds duration, involves oscillatory motions of steadily increasing frequency. At around 10 seconds, accelerations are such that the tracked curve lags the hand appreciably. Variation of gain and search-region width allows lost lock to be recovered (left). However, if variation is inhibited by fixing covariance at* P_∞ *(right), lock is lost irretrievably. The time-course of the translational component of motion is shown in figure 10.11.*



Varying gain and search region



Fixed gain and search region

Figure 10.11: *Time-course of the translational component of motion for the test of figure 10.10. In the fixed gain case, the loss of lock after 12 seconds or so is clearly visible.*

For example, for the hand-mouse it is appropriate to use just two shape-subspaces $\mathcal{S}_s$ and $\mathcal{S}_d$ for translation and deformation respectively. Time constants of the order of a second are appropriate in both subspaces but $\overline{\rho}_s$ might be set an order of magnitude greater than $\overline{\rho}_d$ to reflect relatively large translational excursions across the field of view. The open-loop steady-state distribution will then have a correspondingly large average displacement from the template shape $\overline{\rho}_o \approx \overline{\rho}_s$, dominated by its translational component.

Observation error

Given complete image observations, the search region collapses to the size and shape determined by the prediction covariance $\tilde{P}(t_\infty)$ in the closed-loop steady state. The associated average displacement $\overline{\rho}_c$ relative to the predicted shape, given by

$$\overline{\rho}_c^2 = \operatorname{tr}(\tilde{P}(t_\infty)\mathcal{H}),$$

is determined by the setting of $\overline{\rho}_f$, together with the AR parameters above, and this gives a guide to the size of the search region during steady-state tracking. It is difficult to predict exactly what value $\overline{\rho}_c$ takes but in the case that $\overline{\rho}_f$ is small, including the case $\overline{\rho}_f = 0$,

$$\overline{\rho}_c \approx \sqrt{\operatorname{tr}(B_0 B_0^T \mathcal{H})}, \tag{10.8}$$

approximately independent of the precise value of $\overline{\rho}_f$. For the partitioned harmonic model this gives

$$\overline{\rho}_c \approx \sqrt{\sum_i (b_0^i)^2 N_i}. \tag{10.9}$$

For example, in the case of simple harmonic motion over an unpartitioned shape-space this becomes

$$\overline{\rho}_c \approx b_0 \sqrt{N_X}$$

which can be related to $\overline{\rho}_c$ under the continuous-time assumption (9.27) to give

$$\frac{\overline{\rho}_c}{\overline{\rho}_o} = 2\left(\beta\left(\beta^2 + 2\pi^2 f^2\right)\right)^{1/2} \tau^{3/2}. \tag{10.10}$$

This is a useful design rule, giving the factor by which the search region contracts between the open and closed-loop conditions. Note that this case of *exact* measurement is extreme; as measurements become less precise, the ratio between search-region dimensions in the open condition versus the closed one decreases.

Setting $\overline{\rho}_f$ to a small value also ensures accurate position estimates — average positional error is bounded by $\overline{\rho}_f$ in general — but only if observations are successful. If the background is clutter-free, tracking may succeed with a small $\overline{\rho}_f$, and estimated positions will be accurate. If there is significant background clutter, this must be reflected by assuming a larger value of $\overline{\rho}_f$, the average observation error. The effect is to widen the search region, tolerating greater levels of clutter-induced observation error, at the expense of less accurate position estimates.

10.4 Case study

To conclude the chapter, typical, practical settings of dynamical parameters are given for an application — the digital desk. First the case of tracking hand motion over a clean desk is given, and this applies also to a cluttered desk in which the background has been modelled and can be discounted. The aim is to achieve tracking of the agile motions that occur in drawing and pointing gestures. Then the more difficult problem of tracking over unmodelled clutter is addressed.

Three different settings of dynamical parameters, in a shape-space of Euclidean similarities, are illustrated in figures 10.12 and 10.13: "slow," "fast" and "tight."

"Slow" dynamics are:

translation:

$$f = 0,\ \beta = 2\,\mathrm{s}^{-1} \ \text{ and } \ \overline{\rho}_o = 2000\,\text{pixels},$$

which is critically damped, slow given its $\frac{1}{2}$ second time-constant, and very loosely constrained spatially, to allow free rein over the 500 pixel extent of the image.

rotation/scaling:

$$f = 0,\ \beta = 10\,\mathrm{s}^{-1} \ \text{ and } \ \overline{\rho}_o = 50\,\text{pixels},$$

which is tightly constrained, reflecting relatively strong prior knowledge about the shape of the outline.

Observed features are edges detected with a Gaussian operator of width 2 pixels, and with a contrast threshold of 8 grey-levels. Typically several features are detected along each search line; the one with strongest contrast is chosen but with a modest bias for zero innovation. Observation uncertainty is set to

$$\overline{\rho}_f = 5\,\text{pixels},$$

a reasonable value to allow for measurement error, unmodelled shape variations, and the possibility of reporting clutter as an object feature.

"Fast" More agile performance is allowed by increasing the damping rate for translation to

$$\beta = 5\,\mathrm{s}^{-1}.$$

This has the side-effect of increasing the uncertainty in predicted position at each time-step and increasing the width of the search region, increasing the tendency for distraction by spurious features internal to the hand. To compensate, account is taken of contrast polarity, accepting only light to dark transitions on the hand outline. To help further, the contrast threshold is increased to 25 grey-levels, capitalising on the darkness of the background. Now tracking is fast and accurate.

“Tight” A cluttered background (figure 10.13) that is entirely static can be modelled and suppressed, and the problem is essentially no harder than before. Otherwise, clutter seriously distracts the “fast” tracker, so that dynamical parameters need to be tightened up for tracking to work at all. The clutter problem is exacerbated further by the fact that contrast with the desk is no longer so pronounced; the contrast threshold has to be increased to 12 grey-levels, and this increases the effective density of clutter. What is more, polarity of contrast can no longer be relied upon, since some of the desktop is darker than the hand and some lighter. An alternative that works well is to use colour (chapter 5), relying on the distinctive pink hue of skin to discriminate from the background. Sensitivity to clutter needs to be reduced further. This is done by reducing freedom to rotate/scale, by setting

$$\overline{\rho}_o = 25\,\text{pixels},$$

half its previous value. The search region along each line is artificially restricted to 60% of its normal value by reducing the length factor κ in the validation test of figure 8.6 from the usual $\kappa = 2$ to $\kappa = 1.2$. Now slower hand motions can be tracked satisfactorily over clutter.

To summarise, good performance can be obtained with manually set dynamics, for tracking with a clean background, or a background that is cluttered but static so that it can be discounted. If clutter persists, performance is more limited. In that case, the next chapter shows how to set more acutely tuned dynamical parameters, using examples of motion tracked against a clean background for training. This helps somewhat to deal with the cluttered background by strengthening prior knowledge of shape and motion. More radically, the methods of chapter 12 offer powerful, non-Gaussian methods to handle clutter which are very effective but more costly computationally.

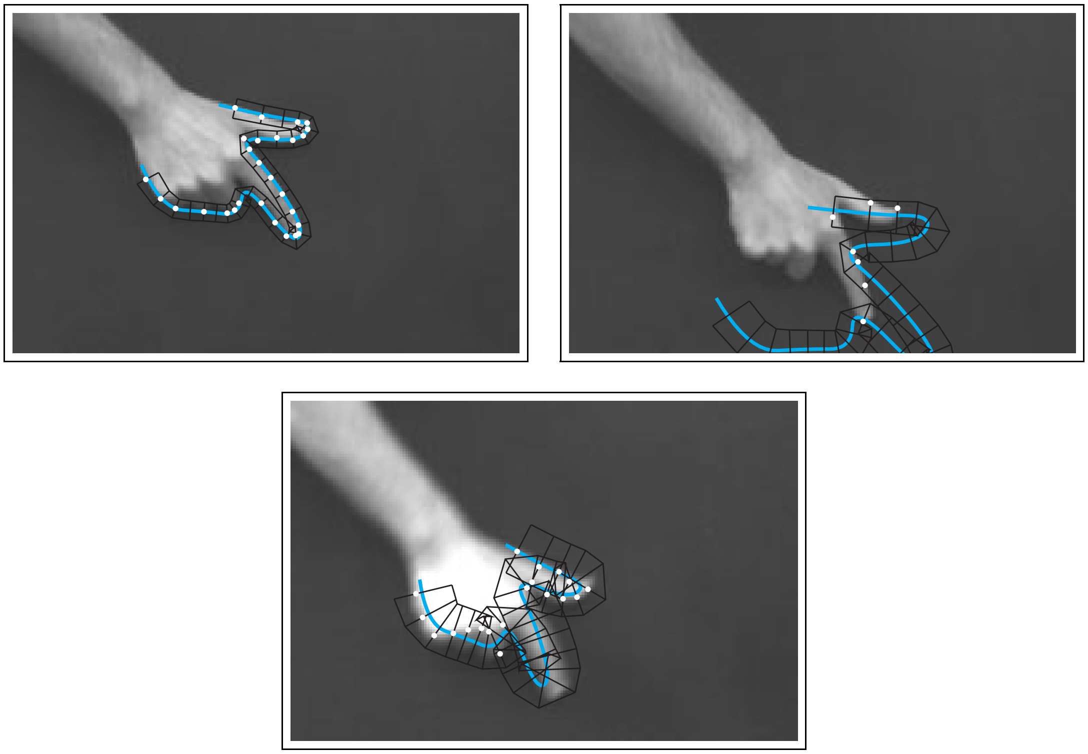


Figure 10.12: Case study: setting up for the digital desk. *"Slow" tuning (top) is stable, works well for gentle motions, but is derailed by faster ones. "Fast" tuning (bottom) is not quite so stable but is capable of following rapid motions — note the motion blur in the still image.*

Bibliographic notes

Established uses of Kalman filtering in machine vision include (Bolles et al., 1987; Broida and Chellappa, 1986; Bolles et al., 1987; Dickmanns and Graefe, 1988b; Matthies et al., 1989; Gennery, 1992). Terzopoulos and Szeliski (1992) recognised the connection between snakes with dynamics and the formalism of the Kalman filter. Low-dimensional curve parameterisations such as the B-spline make it practicable to implement a snake as a Kalman filter (Blake et al., 1993). The idea of using a spatially extended validation gate in a contour tracker was proposed and developed in (Curwen and Blake, 1992; Blake et al., 1993).

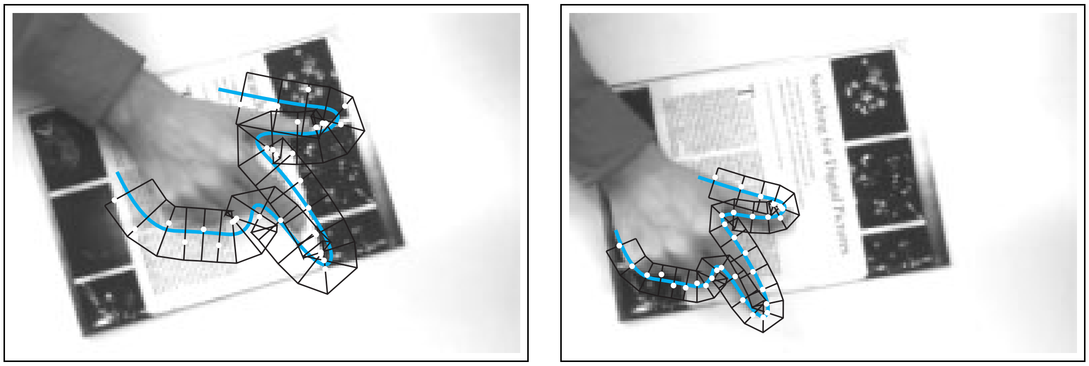


Figure 10.13: Case study: heavy background clutter. *The "fast" tracker of figure 10.12 is too easily distracted by clutter (left) to be used successfully against an unmodelled background of text and pictures. "Tight" parameter settings (right) secure adequate stability, but can only track slower motions.*

In its standard form, the Kalman filter is formulated in terms of Kalman gain (Gelb, 1974), which is derived from the information weighted mean using the "matrix inversion lemma" (Bar-Shalom and Fortmann, 1988). The form of Kalman filter used in this chapter, that avoids singularity problems when measurements are degenerate or fail altogether, was taken from (Harris, 1992b) in which Kalman filtering was applied to the non-linear parameters for position/orientation of a single, polyhedral rigid body. It achieved impressive results in terms of efficiency and agility of tracked motion. Lowe (1992) addressed the related problem of tracking an articulated pair of rigid bodies, in terms of its extended, non-linear parameterisation.

Chapter 11

Learning motion

In the previous chapter, dynamic contour tracking was based on prediction using dynamical models of the kind set out in chapter 9. The parameters of the models were fixed by hand to represent plausible motions such as constant velocity or critically damped oscillation. Experimentation allows these parameters to be refined by hand for improved tracking but this is a difficult and unsystematic business, especially in high-dimensional shape-spaces which may have complex couplings between the dimensions. What is far more attractive is to learn dynamical models on the basis of training sets. Initially, a hand-built model is used in a tracker to follow a training sequence which must be not too hard to track. This can be achieved by allowing only motions which are not too fast, and limiting background clutter or eliminating it using background subtraction (chapter 5). Once a new dynamical model has been learned, it can be used to build a more competent tracker, one that is specifically tuned to the sort of motions it is expected to encounter. That can be used either to track the original training sequence more accurately, or to track a new and more demanding training sequence, involving greater agility of motion. The cycle of learning and tracking is described in figure 11.1. Typically two or three cycles suffice to learn an effective dynamical model.

In mathematical terms, the problem is to estimate the coefficients A_1, A_2, $\overline{\mathbf{X}}$ and B_0 which best model the motion in a training sequence of shapes $\mathbf{X}_1, \ldots, \mathbf{X}_M$, where now $\mathbf{X}_k \equiv \mathbf{X}(t_k)$, gathered at the image sampling frequency. A general algorithm to do this is described below. Note that the learning algorithm as presented treats estimated shape-vectors $\mathbf{X}_k$ in a training sequence as if they were exact observations of the physical process, rather than noisy estimates obtained from a visual tracker. In

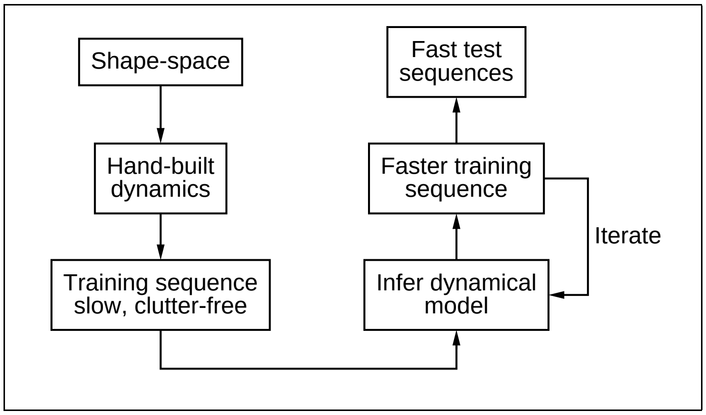


Figure 11.1: Iterative learning of dynamics. *The model acquired in one cycle of learning is installed in a tracker to interpret the training sequence for the next cycle. The process is initialised with a hand-built tracker of the sort described in chapter 9 and 10.*

practice this often works quite well but can give surprising results with highly periodic training motions, and this is discussed later.

11.1 Learning one-dimensional dynamics

First a learning algorithm is described for the simple case of a particle in one dimension — no curve or splines are involved here for the sake of tutorial simplicity, just one number describing the position of the particle along a rail. The problem is to estimate a discrete-time, second order model for particle position x_k so the state-space is defined in terms of $\mathbf{X}_k \equiv x_k$. Then the dynamical coefficients A_1, A_2 and B_0 become scalars denoted a_1, a_2 and b_0. For simplicity, assume that the mean $\overline{x} = 0$ is known, so that the quantity $x_k - a_2 x_{k-2} - a_1 x_{k-1}$ is an independent, zero-mean, scalar, normal variable $b_0 w_k$, for each k, with unknown variance b_0^2. The algorithm is summarised in figure 11.2.

The problem is expressed in terms of a "log-likelihood" function, defined up to an additive constant by

$$L(x_1, \ldots, x_k | a_1, a_2, b_0) \equiv \log p(x_1, \ldots, x_k | a_1, a_2, b_0) \quad + \quad \text{const}$$

where, since the w_k are independent,

$$p(x_1, \ldots, x_k | a_1, a_2, b_0) \propto \prod_k p_{b_0 w_k}(x_k - a_2 x_{k-2} - a_1 x_{k-1})$$

so, using the fact that the $p_{b_0 w_k}(\cdot)$ are standard normal distributions,

$$L(x_1, \ldots, x_k | a_1, a_2, b_0) = -\frac{1}{2b_0^2} \sum_{k=3}^{M} (x_k - a_2 x_{k-2} - a_1 x_{k-1})^2 \; - \; (M-2) \log b_0, \quad (11.1)$$

up to an additive constant. Maximum Likelihood Estimates (MLE) for the coefficients a_1, a_2 and b_0 are obtained by maximising the function L, and this is straightforward because L is quadratic. First, the maximisation over b_0 factors out, leaving estimates $\hat{a}_1$ and $\hat{a}_2$ to be determined by minimising

$$\sum_{k=3}^{M} (x_k - a_2 x_{k-2} - a_1 x_{k-1})^2$$

whose derivatives with respect to a_1 and a_2 are set to 0 to give $\hat{a}_1, \hat{a}_2$ as the solution of the simultaneous equations in step 2 of the algorithm in figure 11.2. Now a_1 and a_2 can be regarded as constants in L, fixed at their estimated values, and L can be maximised with respect to b_0 to give $\hat{b_0}$:

$$\hat{b_0}^2 = \frac{1}{M-2} \sum_{k=3}^{M} (x_k - \hat{a}_2 x_{k-2} - \hat{a}_1 x_{k-1})^2 \quad (11.2)$$

which, it can be shown, can be computed directly in terms of auto-correlation coefficients, as in step 3 of the algorithm.

Exercising the learning algorithm

Talking lips

Learned univariate motion is illustrated by the example used earlier, in chapter 9, of the opening/shutting motion of a pair of talking lips. Once the learning is done, the learned model $(\hat{a_2}, \hat{a_1}, \hat{b_0})$ can be displayed in spectral form, as was done in chapter 9 for models fit by hand. Figure 11.3 shows that the spectrum of the learned model

Dynamical learning problem

Given a training set $\{x_1, \ldots, x_M\}$ of shapes from an image sequence, learn the parameters a_1, a_2, and b_0 for a second-order AR process that describes the dynamics of the moving shape.

Algorithm

1. First, auto-correlation coefficients r_{ij} for $i, j = 0, 1, 2$ are computed:

$$r_{ij} = \sum_{k=3}^{M} x_{k-i} x_{k-j} \quad \text{for} \quad i, j = 0, 1, 2$$

2. Estimated parameters $\hat{a}_1$ and $\hat{a}_2$ are obtained by solving the simultaneous equations

$$\begin{aligned} r_{02} - \hat{a}_2 r_{22} - \hat{a}_1 r_{12} &= 0 \\ r_{01} - \hat{a}_2 r_{21} - \hat{a}_1 r_{11} &= 0, \end{aligned}$$

3. The covariance coefficient b_0 is estimated as

$$\hat{b_0}^2 = \frac{1}{m-2} \left(r_{00} - \hat{a}_2 r_{20} - \hat{a}_1 r_{10} \right)$$

Figure 11.2: Algorithm for learning one-dimensional dynamics.

fits fairly well, having a resonant frequency of around 1 Hz, which is a little higher than the peak of 0.8 Hz in the spectrum of the data. The hand-fitted spectrum in figure 9.11 on page 201 was constrained to have its resonant peak coinciding with that of the data. However, as a trade-off for the shift in its resonant peak, the learned model fits the high frequency spectrum rather better. The learned model should also capture some of the phase coherence of the data, something that was not taken into

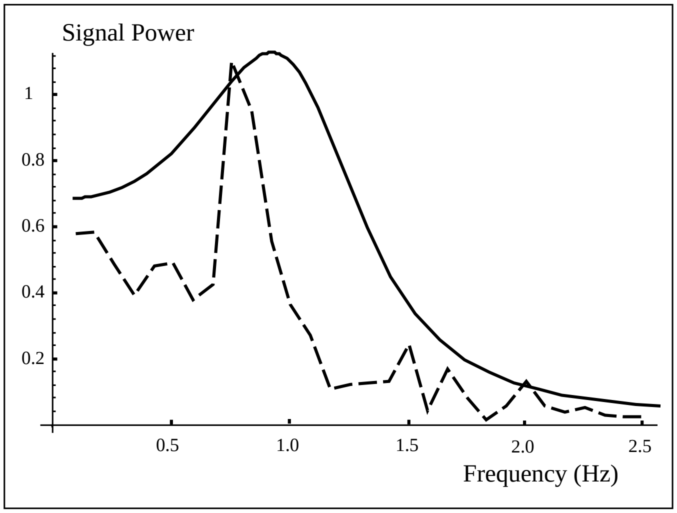


Figure 11.3: Power spectrum of learned lip motion *The power spectrum of the learned model (solid) fits that of the lip-motion training data (dashed) quite well.*

account in the hand-fitted model. In fact, the learned deterministic parameters are

$$f = 0.95\,\mathrm{Hz}, \quad \beta = 2.9\,\mathrm{s}^{-1},$$

giving the coherence time-constant $1/\beta = 0.35\,\mathrm{s}$ for the process, short enough to indicate that successive cycles in the signal are largely uncorrelated.

As in chapter 9, the learned model can be simulated and compared with the data. The result (figure 11.4) appears, to the naked eye, to be as plausible a replica of the data as the simulation of the hand-fitted model was (figure 9.12 on page 202).

Beating heart

It is instructive to see how the learning algorithm deals with training data that is highly periodic and coherent. Of course, perfectly coherent, periodic data of frequency f_0 can be represented by a second-order ARP with $f = f_0$, $\beta = 0$ and with $b_0 = 0$ — zero driving noise, so that the process is entirely deterministic. The behaviour of such a process is solely determined by initial conditions: once launched, the process follows an entirely predictable trajectory. Such a model, if used as a predictor for tracking, would have the strange effect that the gain of the Kalman filter would fall

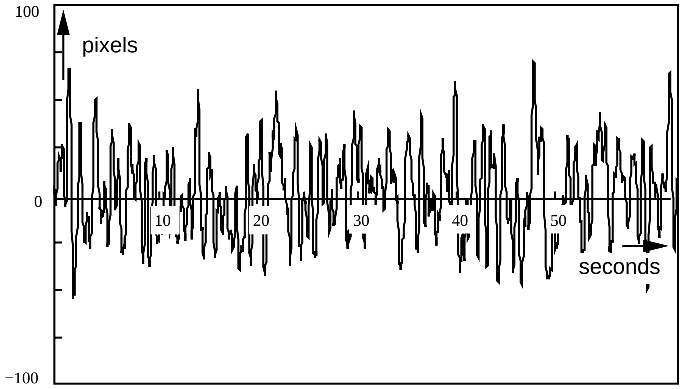


Figure 11.4: Simulation of learned lip motion *A sample path for the learned process is shown here and appears plausibly comparable with the original signal in figure 9.8 on page 198.*

to 0 in the steady state, because the strength of the predictions would overwhelm the observations. Consequently observations would be ignored.

What is more realistic is *almost* periodic training data such as that from the beating heart in figure 9.7 on page 198. It appears to be a regular, periodic process with some additive noise, attributable perhaps to noise in observations. As noted earlier, the learning algorithm neglects to allow for observation noise and it is with highly periodic data that the discrepancy introduced by this erroneous assumption is most evident. Figure 11.5 shows a simulation of a learned dynamical model, exhibiting the right kind of motion over short time-scales (of the order of half a period), but over longer time-scales the phase coherence of the original signal is lost. The deterministic parameters of the learned model are

$$f = 1.12\,\mathrm{Hz}, \quad \beta = 3.5\,\mathrm{s}^{-1},$$

representing an oscillation of period $1/f = 0.90\,\mathrm{s}$ which matches closely the average period 0.87 s of the data (based on zero crossings). However, the damping represented by the factor β is strong, representing a decay in amplitude by a factor of 5 over one half period. This corresponds to the loss of coherence apparent in the simulation over times greater than one half period or so. In fact the learned model, despite its imperfection, is still useful for tracking because it works well as a predictor over shorter time-scales, and visual observations carry the signal over longer time-scales. Moreover, the imperfection can be remedied by recourse to a more elaborate learning

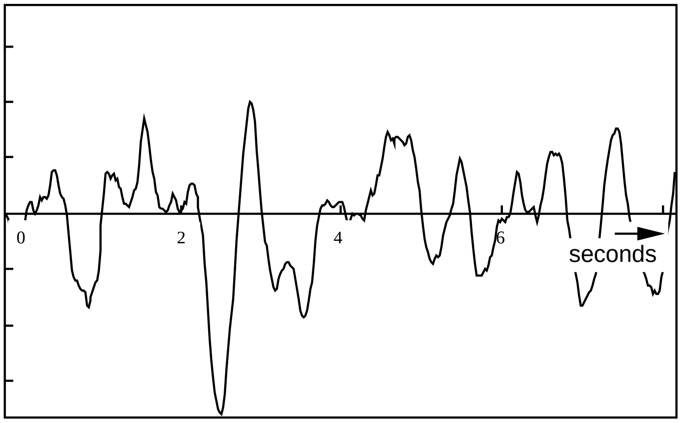


Figure 11.5: Simulating the learned motion of a beating heart. *A dynamical model learned from the training data of figure 9.7 on page 198 is simulated here. Limitations of the learning algorithm (see text) means that although the periodic nature of the training data is represented (each half-cycle is roughly the right size) its phase coherence is lost.*

procedure using "Expectation–maximisation" or EM (see bibliographic notes) which takes explicit account of observation noise.

Learning accuracy and training-sequence duration

Intuitively, a longer training sequence should give more reliable learned dynamical parameters. This is indeed the case, and the nature of the influence of the number of frames M on the accuracy of learned parameters a_1, a_2 and b_0 can be made precise. In fact what is most relevant is to quantify the effect of training-set duration $T = (M-2)\tau$ on the parameters β, f and $\overline{\rho}$ of the underlying continuous process. In this way, a characterisation of learning performance is obtained that is independent of sampling rate τ. The characterisation is valid under the assumptions that $\beta\tau \ll 1$ and that the process is observed in its steady state. A derivation is given in appendix B.

The principal result is that the proportional error in the parameters β and $\overline{\rho}$ is of order $1/\sqrt{\beta T}$, and that the proportional error in f relative to β is also of order $1/\sqrt{\beta T}$. As expected, accuracy varies as the square root of training-set size, the usual statistical phenomenon of error averaging. What may be more surprising at first sight is that the error depends not on M directly, but on βT which can be thought of as the number of *independent chunks* in the training signal. This is because $1/\beta$ is a "coherence time," the duration over which parts of the signal are correlated. In the lip-motion example above, for instance, the coherence duration is $1/\beta = 0.35\,\mathrm{s}$ and the total duration of the training set (figure 9.8 on page 198) is

60 seconds. In that case $\beta T = 171$ so that the proportional error in the continuous dynamical parameters should be

$$100\sqrt{\frac{60}{0.35}}\% \approx 7.6\%.$$

For the beating heart the coherence duration is 0.29 s but the training sequence lasts only 8 seconds, giving a bound on error in dynamical parameters of

$$100\sqrt{\frac{8.0}{0.29}}\% \approx 19\%.$$

11.2 Learning AR process dynamics in shape-space

The general multi-variate learning algorithm follows broadly the line of the univariate one, but the separability of the estimation of deterministic and stochastic parameters, although it still holds, is no longer so obvious. Furthermore, it will no longer be assumed that the mean $\overline{\mathbf{X}}$ is known, so that it also must be learned. The log-likelihood function for the multi-variate normal distribution is then, up to a constant:

$$\begin{aligned} &L(\mathbf{X}_1, \ldots \mathbf{X}_M | A_1, A_2, C, \overline{\mathbf{X}}) = \\ &\quad -\frac{1}{2}\sum_{k=3}^{M} \left| B_0^{-1} \left(\mathbf{X'}_k - A_2 \mathbf{X'}_{k-2} - A_1 \mathbf{X'}_{k-1} \right) \right|^2 - (M-2) \log \det B_0 \end{aligned} \tag{11.3}$$

where

$$\mathbf{X'}_k = \mathbf{X}_k - \overline{\mathbf{X}}, \tag{11.4}$$

and

$$C = B_0 B_0^T. \tag{11.5}$$

The problem is to estimate A_1, A_2, $\overline{\mathbf{X}}$ and C by maximising the log-likelihood L. This is a non-linear problem because, unlike the simple case considered earlier in which the mean was fixed at $\mathbf{0}$, the mean $\overline{\mathbf{X}}$ now has to be estimated. This means that the likelihood is quartic (rather than quadratic) in the unknowns, owing to the product terms $A_2\overline{\mathbf{X}}$ and $A_1\overline{\mathbf{X}}$ that appear inside the $|\cdot|^2$ term in (11.3). The non-linearity can be removed by using the alternative form for the AR process from chapter 9 (equation (9.6) on page 193) in which

$$\mathbf{D} = (I - A_2 - A_1)\overline{\mathbf{X}}$$

so that the likelihood becomes

$$L(\mathbf{X}_1 \ldots \mathbf{X}_M | A_1, A_2, C, \mathbf{D}) = \tag{11.6}$$
$$-\frac{1}{2} \sum_{k=3}^{M} \left| B_0^{-1} \left(\mathbf{X}_k - A_2 \mathbf{X}_{k-2} - A_1 \mathbf{X}_{k-1} - \mathbf{D} \right) \right|^2 - (M-2) \log \det B_0$$

which is then quadratic in A_2, A_1 and $\mathbf{D}$.

Minimising the log-likelihood L leads to the learning algorithm of figure 11.6 which estimates the dynamical parameters A_2, A_1, $\mathbf{D}$ and C. It is clearly a generalisation of the univariate algorithm (figure 11.2). A derivation (optional) is given later.

Example: learning the dynamics of writing

As an illustration of multi-variable learning, the written name of figure 9.2 on page 187 is used as training data, to learn a dynamical model for finger-writing. A good test for the plausibility of a learned model is to simulate it randomly, as was done in chapter 9 for synthesised dynamical models. A random simulation of the model for finger-writing is demonstrated in figure 11.7. The resulting scribble has characteristics consistent with the training set in terms of direction and the size, shape and frequency of excursions. Note that the dynamical model was actually learned in the affine space for the finger outline and that the simulation therefore contains (minor) changes of finger shape. The figure shows just the translational components, which account for most of the motion in this case. (The learned model specifies dynamics in the steady state, but not initial conditions; the *speed* at which the hand drifts across the page is left indeterminate by the model. In this demonstration, the handwriting simulation was initialised with zero velocity, and then superimposed on a constant velocity drift roughly matching that in the training set.)

Modular learning: aggregation of training sets

It is often convenient to collect several training sets and to construct a dynamical model which explains the motion in all of the training sets taken jointly. For example, it might be desired to broaden the finger-writing model to cover writing in various directions, not just the single direction in the training set illustrated. Alternatively, when learning lip dynamics (see below) it might be convenient to construct separate training sets for rigid motion of the head, and deformation during speech. In either

Dynamical learning problem

Given a training set $\{\mathbf{X}_1, \ldots, \mathbf{X}_M\}$ of shapes from an image sequence, learn the parameters A_1, A_2, B_0 and $\overline{\mathbf{X}}$ for a second-order AR process that describes the dynamics of the moving shape.

Algorithm

1. First, sums $R_i,\ i = 0, 1, 2$ and auto-correlation coefficients R_{ij} and $R'_{ij},\ i, j = 0, 1, 2$ are computed:

$$R_i = \sum_{k=3}^{M} \mathbf{X}_{k-i}, \quad R_{ij} = \sum_{k=3}^{M} \mathbf{X}_{k-i}\mathbf{X}_{k-j}^T, \quad R'{}_{ij} = R_{ij} - \frac{1}{M-2} R_i R_j^T.$$

2. Estimated parameters $\hat{A}_1$, $\hat{A}_2$ and $\hat{\mathbf{D}}$ are given by

$$\begin{aligned}
\hat{A}_2 &= \left(R'{}_{02} - R'{}_{01} R'{}_{11}^{-1} R'{}_{12}\right) \left(R'{}_{22} - R'{}_{21} R'{}_{11}^{-1} R'{}_{12}\right)^{-1} \\
\hat{A}_1 &= \left(R'{}_{01} - \hat{A}_2 R'{}_{21}\right) R'{}_{11}^{-1} \\
\hat{\mathbf{D}} &= \frac{1}{M-2}\left(R_0 - \hat{A}_2 R_2 - \hat{A}_1 R_1\right).
\end{aligned}$$

3. If required for the standard form of the AR process, the mean $\overline{\mathbf{X}}$ is estimated from

$$\hat{\overline{\mathbf{X}}} = (I - \hat{A}_2 - \hat{A}_1)^{-1} \hat{\mathbf{D}}.$$

4. The covariance coefficient B_0 is estimated as a matrix square root $\hat{B}_0 = \sqrt{\hat{C}}$ where

$$\hat{C} = \frac{1}{M-2}\left(R_{00} - \hat{A}_2 R_{20} - \hat{A}_1 R_{10} - \hat{\mathbf{D}} R_0^T\right).$$

Figure 11.6: Algorithm for learning multi-variate dynamics.

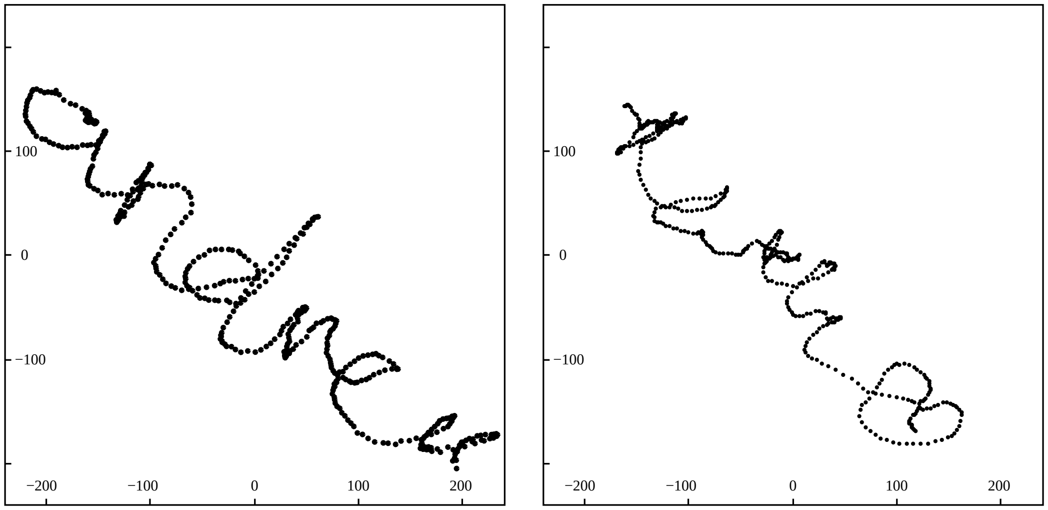


Figure 11.7: Scribbling: simulating a learned model for finger-writing. *A training set (left) consisting of six handwritten letters is used to learn a dynamical model for finger motion. A random simulation from the model (right) exhibits reasonable gross characteristics.*

case it is *incorrect* simply to concatenate the training sets and treat them as one long training set because the abrupt transition at the join of the sets would be treated as genuine data, whereas in fact it is spurious. The solution is to compute autocorrelations individually for the training sets and to combine them in a linear fashion. Details of the combination method are omitted here, but see the bibliographic notes.

Derivation of the learning algorithm of figure 11.6.

Maximising likelihood L (11.6) first with respect to A_1, A_2 and $\mathbf{D}$, it will be shown that separability holds — the resulting estimates $\hat{A}_2$, $\hat{A}_1$ and $\mathbf{D}$ are independent of the value of C. To show this, a lemma is needed.

Lemma: Given a scalar function $f(Y)$ of a matrix Y of the form

$$f = \mathrm{tr}(KZ) \quad \text{with} \quad Z(Y) = YSY^T - Y{S'}^T - S'Y^T,$$

in which K, S and S' are constant matrix coefficients and S and K are non-singular and symmetric, the value $Y = \hat{Y}$ at which $\partial f / \partial Y = 0$ is independent of K.
Proof: $Z(Y)$ can be rewritten

$$Z(Y) = (Y - \hat{Y})S(Y - \hat{Y})^T + \text{ const} \quad \text{where} \quad \hat{Y} = S'S^{-1}$$

and then

$$dZ = dY\, S(Y-\hat{Y})^T + (Y-\hat{Y})S\, dY^T$$

and, since K is symmetric,

$$df = 2\,\mathrm{tr}\left(K\, dY\, S(Y-\hat{Y})^T\right)$$

so that $df = 0$ for all dY iff $Y = \hat{Y}$, independent of the value of K.
□

Now we can proceed with maximising L, which is equivalent to minimising

$$f(A_1, A_2, \mathbf{D}) = \sum_{k=3}^{M} \left|B_0^{-1}\left(\mathbf{X}_k - A_2\mathbf{X}_{k-2} - A_1\mathbf{X}_{k-1} - \mathbf{D}\right)\right|^2 \tag{11.7}$$

with respect to A_1 and A_2. This function can be expressed as

$$f(A_1, A_2, \mathbf{D}) = \mathrm{tr}(ZC^{-1})$$

where

$$Z = \sum_{k=3}^{M}\left(\mathbf{X}_k - A_2\mathbf{X}_{k-2} - A_1\mathbf{X}_{k-1} - \mathbf{D}\right)\left(\mathbf{X}_k - A_2\mathbf{X}_{k-2} - A_1\mathbf{X}_{k-1} - \mathbf{D}\right)^T$$

and $C = B_0 B_0^T$. Invoking the lemma three times, with Y as each of A_0, A_1 and $\mathbf{D}$ in turn (and with the other two treated as constant), indicates that we can effectively set $B_0 = I$, for the purposes of determining $\hat{A}_1$, $\hat{A}_2$ and $\mathbf{D}$ by finding the minimum of

$$\mathrm{tr}(Z) = \sum_{k=3}^{M}\left|\mathbf{X}_k - A_2\mathbf{X}_{k-2} - A_1\mathbf{X}_{k-1} - \mathbf{D}\right|^2.$$

Setting to 0 the derivatives of $\mathrm{tr}(Z)$ with respect to A_0, A_1 and $\mathbf{D}$ respectively shows that the minimum must satisfy the simultaneous equations

$$\begin{aligned} R_{02} - \hat{A}_2 R_{22} - \hat{A}_1 R_{12} - \hat{\mathbf{D}} R_2 &= 0 \\ R_{01} - \hat{A}_2 R_{21} - \hat{A}_1 R_{11} - \hat{\mathbf{D}} R_1 &= 0 \\ R_0 - \hat{A}_2 R_2 - \hat{A}_1 R_1 - \hat{\mathbf{D}}(M-2) &= 0, \end{aligned} \tag{11.8}$$

where R_i and R_{ij} are defined as in step 1 of the algorithm. Note that a minimum of f must exist because it is quadratic and bounded below by 0.

Lastly, the simultaneous equations can be simplified by subtracting multiples of the third from the first and second to give

$$\begin{aligned} R'_{02} - \hat{A}_2 R'_{22} - \hat{A}_1 R'_{12} &= 0 \\ R'_{01} - \hat{A}_2 R'_{21} - \hat{A}_1 R'_{11} &= 0 \\ R_0 - \hat{A}_2 R_2 - \hat{A}_1 R_1 - \hat{\mathbf{D}}(M-2) &= 0 \end{aligned}$$

where ${R'}_{ij}$ is defined as in step 1 of the algorithm. Now the first two equations can be solved directly for $\hat{A}_1$ and $\hat{A}_2$ which can then be substituted into the last one to give $\hat{D}$, all as in step 2 of the algorithm.

It remains to estimate B_0 which is obtained as the square root of $C = B_0 B_0^T$. Rewriting (11.6) as

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

fixing $A_2 = \hat{A}_2$, $A_1 = \hat{A}_1$ and $\mathbf{D} = \hat{\mathbf{D}}$, and extremising with respect to C^{-1} (using the identity $\partial(\det M)/\partial M \equiv (\det M)M^{-T}$) gives

$$\hat{C} = \frac{1}{M-2} Z(\hat{A}_2, \hat{A}_1, \hat{\mathbf{D}}), \tag{11.9}$$

which simplifies, using the equations of step 2 of the algorithm, to give the formula for $\hat{C}$ in step 4.

11.3 Dynamical modes

One of the advantages of the "state-space" form for the second-order AR process is that the characteristics of the underlying continuous process can readily be identified. The continuous-time interpretation consists of a number of "modes." Some are damped oscillations with the impulse response $\exp -\beta t \cos 2\pi f t$ that is characteristic of second-order motion. Others are simple decays, with impulse responses of the form $\exp -\beta t$, associated with first-order motion. Each mode exhibits a particular pattern of motion. For example, for a finger tracing out letters (figure 9.2 on page 187) one would expect a slow exponential associated with the gradual, lateral drift, a faster oscillatory mode associated with letter strokes, and various other "minor" modes with relatively rapid decay. Modes characterise the deterministic component of dynamics, indicative of the behaviour of a tracker when the observation process suddenly fails and tracking is left to rely on prediction. That leaves the stochastic component which is relevant under normal tracking conditions, and in particular when observations suddenly resume. The Gaussian envelope following an extended period of prediction is given by the steady-state covariance P_∞, which can be computed from A and B. In fact P_∞ represents the static prior for the dynamical model and as such is similar to the covariance $\overline{P}$ for the training set treated as a static set of shapes, as analysed by PCA in chapter 8.

Modal analysis

Modes of the underlying dynamical process are characterised as follows. First, eigenvalues λ_m and eigenvectors $\mathcal{X}_m^A$ for $m = 1, \ldots, 2N_X$ of the matrix A are computed. Any real, positive eigenvalue with $\lambda_m < 1$ corresponds to a exponentially decaying mode, as for first-order processes, of the form $\exp -\beta_m t$ where

$$\beta_m = \frac{1}{\tau} \log \frac{1}{\lambda_m}. \tag{11.10}$$

Note that if $\lambda_m > 1$ the mode is unstable because $\beta_m < 0$. The mode's pattern of motion in shape-space is given by $\mathbf{X}_m^A$ which is the upper half of the eigenvector

$$\mathcal{X}_m^A = \begin{pmatrix} \mathbf{X}_m^A \\ \mathbf{Y}_m^A \end{pmatrix}. \tag{11.11}$$

(Note that the upper and lower halves are related by $\mathbf{Y}_m^A = \lambda_m \mathbf{X}_m^A$.) Suppose, for an object in Euclidean similarity shape-space, and assuming a sampling interval $\tau = 1/50\,s$, that $\lambda_1 = 0.99$ and $\mathcal{X}_m^A = (1, 1, 0, 0, 0.99, 0.99, 0, 0)^T$. This implies a decaying exponential impulse response with

$$\beta_1 = 50 \log(1/0.99) = 0.50\,\mathrm{s}^{-1}$$

so that the characteristic decay time of the mode is $1/0.50 = 2.0\,\mathrm{s}$. The pattern of displacement for the mode is given by $\mathbf{X}_1^A = (1, 1, 0, 0)^T$ representing translation up and to the right.

Otherwise, eigenvalues can be negative or complex and represent harmonic motion of the form $\exp -\beta_m t \cos 2\pi f_m t$ where

$$\beta_m = \frac{1}{\tau} \log \frac{1}{|\lambda_m|} \tag{11.12}$$

$$f_m = \frac{1}{2\pi\tau} \arg \lambda_m \tag{11.13}$$

(where arg denotes the canonical argument of a complex number). Again, the mode-shape is conveyed by the upper half $\mathbf{X}_m^A$ of the corresponding eigenvector. In general, the elements of $\mathbf{X}_m^A$ are complex, their moduli indicating the amplitude of oscillation in each shape-space component while their complex arguments indicate their relative phases of oscillation. Note that complex eigenvalues must come in conjugate pairs, so each oscillatory mode is expressed in terms of *two* of the eigenvalues out of the $2N_X$ eigenvalues of the matrix A. This means that A can have $2N_X$ exponential modes, or just N_X oscillatory ones, or some combination of the two kinds.

As an example of an oscillatory mode, considering again the object in Euclidean similarity shape-space, $\lambda_2 = 0.95 + 0.24i$ and $\mathbf{X}_2^A = (-1, 1, 0, 0)^T$ imply a damped oscillatory impulse response for which

$$\begin{aligned} \beta_2 &= 50 \log \frac{1}{\sqrt{0.95^2 + 0.24^2}} = 1.02\,\mathrm{s}^{-1} \\ f_2 &= \frac{50}{2\pi} \arctan(\frac{0.24}{0.95}) = 1.97\,\mathrm{Hz} \end{aligned}$$

representing an oscillatory period of $1/1.97 = 0.51\,\mathrm{s}$. The decay time-constant is $1/1.02 = 0.98\,\mathrm{s}$, indicating that the oscillation is coherent over approximately two periods; over intervals longer than this, the phase of oscillation would be expected to drift. The corresponding pattern of motion given by $\mathbf{X}^A$ is translational, up and to the left. If instead $\mathbf{X}_2^A = (1, i, 0, 0)$, the horizontal and vertical components are 90^o out of phase, indicating circular motion. Note that there must be a conjugate eigenvalue, say $\lambda_3 = \lambda_2^*$, representing the same frequency (up to a change of sign) and damping constant, and with $\mathbf{X}_3^A = (\mathbf{X}_2^A)^*$.

Example: analysis of finger-writing

Earlier, in figure 11.7, the learned dynamical model for finger-writing was displayed by a simulation which depicted a plausible scribble. Modal analysis as above reveals that all the modes are stable ($\beta_m < 0$) and that the two slowest modes have time constants

$$\frac{1}{\beta_m} = 75\,\mathrm{s},\ 1.15\,\mathrm{s}$$

respectively. The first of these has frequency $f_m = 0$ so that, given the long decay time, this is effectively a constant-velocity mode. It has an (upper) eigenvector

$$\mathbf{X}^A = (1.0, -1.18, -0.10, 0.05, -0.05, -0.05)^T$$

in affine space which represents predominantly translational motion along the upper-left to lower-right diagonal, following the gross flow of finger-writing in the training set. (Note that affine space has been set up as described in chapter 4 so that components have comparable units, and the neglect of the non-translational components for this mode is therefore valid.) The next mode is oscillatory, with frequency 1.01 Hz. This is consistent with the formation of letters, involving one or two strokes each, so that with around 12 strokes in “andrew,” written over a duration of 10 seconds, 1 Hz is a very plausible frequency. The decay time of 1.15 s is almost exactly one period of oscillation, suggesting that the stroke sequence is not coherent — successive strokes are sufficiently independent that their phases do not match. The eigenvector for the mode is

$$\mathbf{X}^A = (1, 0.38 - 0.28i, -0.74 + 0.13i, 0.47 - 0.28i, -0.51 + 0.53i, -0.43 + 0.23i)^T$$

which contains translation and also other affine components. The translational part, being complex, represents an elliptical motion at the 1 Hz frequency.

11.4 Performance of trained trackers

Preceding sections have shown, using modal analysis, that trained models capture the oscillatory behaviour of typical motions. This section demonstrates that, as expected, tracking performance is enhanced by replacing default dynamics in the predictor with specific dynamics learned from a training set.

Tuning for lip gestures

First an example is given of tracking the motion of talking lips. Lips may be tracked either in a frontal or in a side-on view. The side-on view, whilst arguably less informative in speech analysis applications, has the advantage that the mouth outline is silhouetted and therefore offers high contrast. Deformations of the lips for two sounds are shown in figure 11.8. Individual dynamical models are learned for each of the sounds “Ooh” and “Pah.” For example the “Pah” model is learned from a training sequence in which the “Pah” motion is repeated several times. The resulting selectivity is shown in figure 11.9. It is clear from these results that the tuning effect for individual sounds is strong. Actually to render assistance for speech analysis, it is necessary to learn the repertoire of lip motions that occurs in typical connected speech, and this is addressed next.

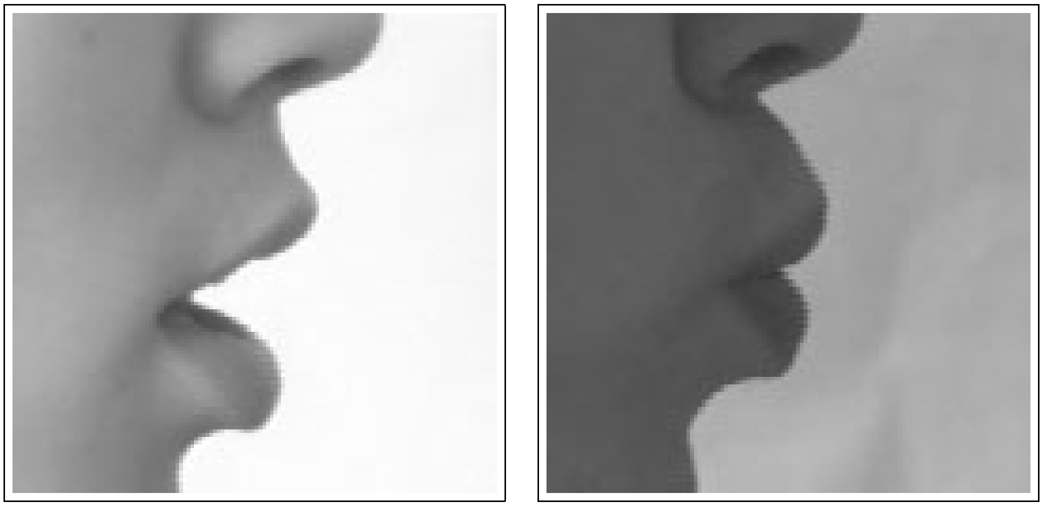


Figure 11.8: Single-syllable training. *Deformations of the mouth are shown corresponding to the sounds a) "Pah" and b) "Ooh."*

Connected speech

Two-stage training is used to learn a dynamical model for connected speech. In the bootstrap stage, the default tracker follows a slow-speech training sequence which is then used, via the learning algorithm, to generate a trained tracker. This "bootstrapped" tracker is capable of following speech of medium speed and is used to follow a medium-speed training sequence, from which dynamics for a full-speed tracker are obtained. The trained tracker is then compared with the default tracker, using a test sequence entirely different from the training sequences. Two deformation components of lip motion are extracted, at 50 Hz, as "lip-reading" signals. The more significant one, in the sense that it accounts for the greater part of the lip motion, corresponds approximately to the degree to which the lips are parted. This component is plotted for the default tracker and the partly and fully trained ones in figure 11.10. It is clear from the figure that the trained filter is considerably more agile. In a demonstration in which the signal was used to animate a head, the untrained filter was clearly able to follow only very slow speech, whereas the trained filter successfully follows speech delivered at normal speed.

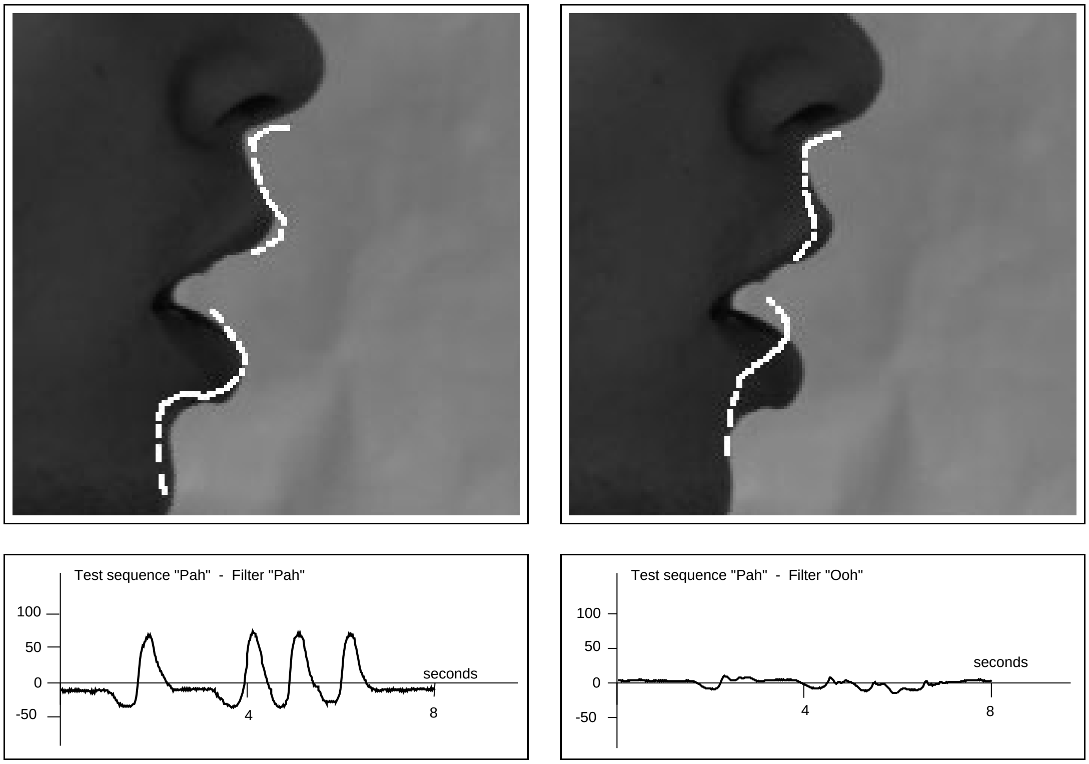


Figure 11.9: Filter selectivity for lip gestures. *The sound "pah" is repeated and tracked by filters trained on the sounds "pah" (left) and "ooh" (right) respectively. Plotted signals refer to the Principal Component of motion in the "Pah" training sequence, which corresponds roughly to the degree to which the mouth is open. Corresponding tracked contours, approximately 4.1 s after the start of the signal, are shown in the snapshots (top left and right). Clearly the filter trained on "Pah" relays the test sequence faithfully, whereas the filter trained on "Ooh" suppresses the test signal almost entirely.*

Bibliographic notes

The dynamical learning algorithm presented here is based on the "Maximum Likelihood" principle (Rao, 1973; Kendall and Stuart, 1979). Maximum likelihood algorithms for estimating dynamics are based on the "Yule-Walker" equations for estimation of the parameters of auto-regressive models (Gelb, 1974; Goodwin and Sin, 1984; Ljung, 1987). A multi-dimensional version of the algorithm which estimates

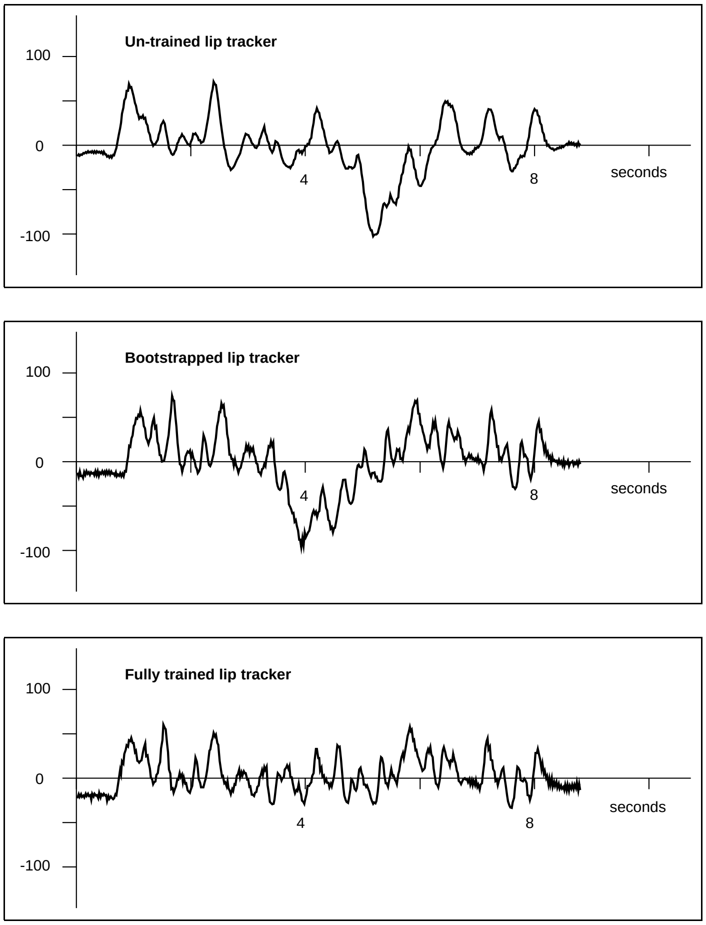


Figure 11.10: Trained lip tracker. *Training a tracker for side-on viewing of speaking lips greatly enhances tracking performance. The graphs show plots from the untrained, default filter, the bootstrapped filter after one training cycle and lastly the filter after a second training cycle. One component of deformation of the lips is shown, corresponding to the degree to which the mouth is open — the space of deformations spanned by the first two templates in figure 4.12 on page 91. Note the considerable loss of detail in the default filter and the overshoots in both default and bootstrapped filters, compared with the fully trained filter. (The sentence spoken here was "In these cases one would like to reduce the dependence of a sensory information processing algorithm on these constraints if possible.")*

not only deterministic parameters A but also the stochastic parameters B is given in (Blake and Isard, 1994; Blake et al., 1995). The particular compact form of the estimator for C in the learning algorithm here is due to Wildenberg (Wildenberg, 1997). A related algorithm for learning deterministic parameters A only is described by Baumberg and Hogg (Baumberg and Hogg, 1995b), who also address the issue of orthogonality constraints on A. The extension of the algorithm to estimate the shape-mean $\overline{\mathbf{X}}$ originated from (Reynard et al., 1996).

An extension of the basic algorithm for *classes* of objects, dealing independently with motion and with variability of mean shape/position over the class, is described in (Reynard et al., 1996). The same algorithm is also used for modular learning — the aggregation of training sets for which a joint dynamical model is to be constructed.

The result that the steady-state covariance P_∞ in a learned dynamic model approximates to the sample covariance $\overline{P}$ for the training set treated as a static set of shapes is described by (Wildenberg, 1997).

The learning algorithm treats the training set as exact whereas in fact it is inferred from noisy observations. Dynamics can be learned directly from the observations using Expectation–maximisation (EM) (Dempster et al., 1977). Learning dynamics by EM is suggested by Ljung (Ljung, 1987) and the detailed algorithm is given in (North and Blake, 1998). It is related to the Baum-Welch algorithm used to learn speech models (Huang et al., 1990; Rabiner and Bing-Hwang, 1993) but with additional complexity because the state-space is continuous rather than discrete.

A number of alternative approaches have been proposed for learning dynamics, with a view to gesture recognition rather than tracking — see for instance (Mardia et al., 1993; Campbell and Bobick, 1995; Bobick and Wilson, 1995).

Chapter 12

Non-Gaussian models and random sampling algorithms

This chapter describes in detail a powerful algorithm for contour tracking that uses random sampling — the CONDENSATION algorithm. It applies to cases where there is substantial clutter in the background. Clutter presents a particular challenge because elements in the background may mimic parts of foreground features. In the most severe case of camouflage, the background may consist of objects similar to the foreground object, for instance when a person is moving past a crowd. The probability density for $\mathcal{X}$ at time t_k is multi-modal and therefore not even approximately Gaussian. The Kalman filter is not suited to this task, being based on pure Gaussian distributions.

The Kalman filter as a recursive linear estimator is a special case, applying only to Gaussian densities, of a more general probability density propagation process. In continuous time the process would be described in terms of diffusion, governed by a "Fokker-Planck" equation, in which the density for $\mathcal{X}(t)$ drifts and spreads under the action of a stochastic model of its dynamics. In the simple Gaussian case, the diffusion is purely linear and the density function evolves as a Gaussian pulse that translates, spreads and is reinforced. It remains Gaussian throughout, as in figure 10.1 on page 214, and its evolution is described analytically and exactly by the Kalman filter. The random component of the dynamical model leads to spreading — increasing uncertainty — while the deterministic component causes the density function to drift bodily. The effect of an external observation $\mathbf{Z}(t)$ is to superimpose a reactive effect on the diffusion in which the density tends to peak in the vicinity of observations. In

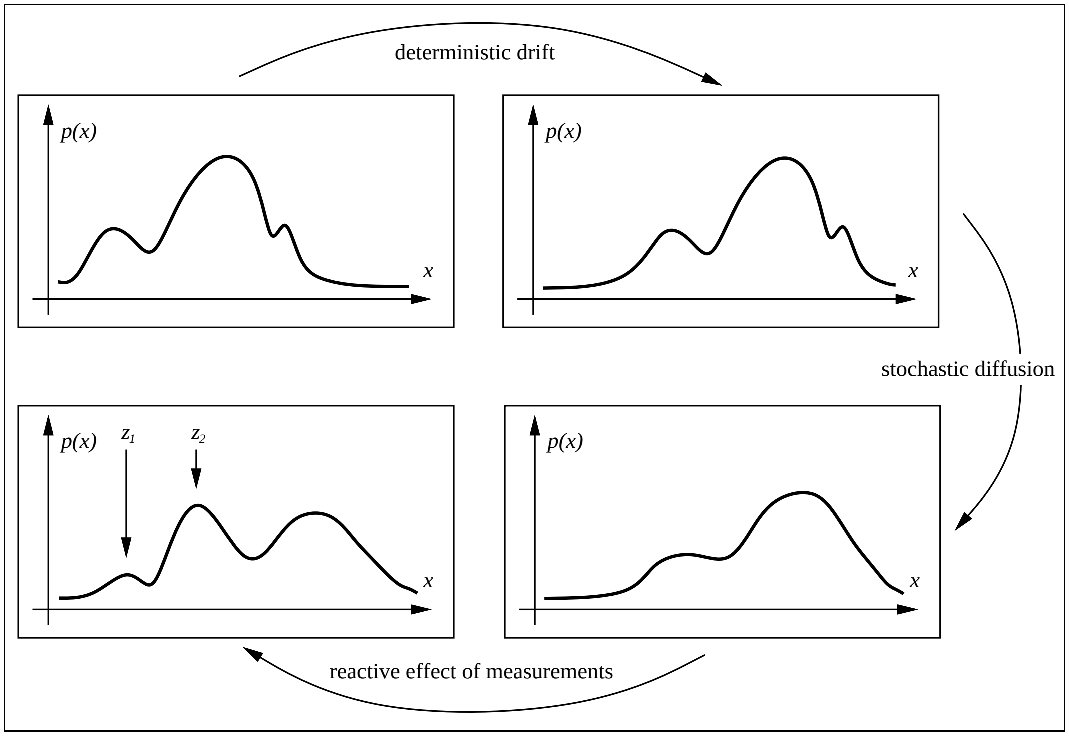


Figure 12.1: Probability density propagation. *In general, the state density describing an object is multi-modal (compare with the Gaussian model used by the Kalman filter in figure 10.1 on page 214). Propagation occurs in three phases: drift due to the deterministic component of object dynamics; diffusion due to the random component; reactive reinforcement due to observations.*

clutter, there are typically several competing observations and these tend to encourage a non-Gaussian state density (figure 12.1).

The Condensation algorithm (CONDitional DENSity propagATION) is designed to address this more general situation. It has the striking property that, generality notwithstanding, it is a considerably simpler algorithm than the Kalman filter. Moreover, despite its use of random sampling which is often thought to be computationally inefficient, the Condensation algorithm runs in near real time. This is because tracking over time maintains relatively tight distributions for shape at successive time-steps, and particularly so given the availability of accurate, learned models of shape and motion.

12.1 Factored sampling

This section describes the factored sampling algorithm, which can be used to search single still images for an object when observations are non-Gaussian. Then, in the following section, the CONDENSATION algorithm is presented as an extension of factored sampling which handles temporal image sequences.

Chapter 8 introduced the notion of the posterior distribution for an object, and the task of curve-fitting was cast as the problem of maximising $p(\mathbf{X}|\mathbf{Z})$ ((8.19) on page 171), where $\mathbf{Z}$ denoted the "aggregated observation" from least-squares fitting in chapter 6. By restricting the prior and observation densities to be Gaussian, the posterior was also constrained to be Gaussian, described completely by its mean and covariance matrix which were evaluated in closed form using the recursive fitting algorithm on pages 127 and 174.

In clutter, $\mathbf{Z}$ has to incorporate all of the information in an image. It is no longer valid to assume that the features consist of one measurement on each normal of a single curve, and details of the form of $\mathbf{Z}$ to be used will be given in section 12.3. In the general case the prior $p(\mathbf{X})$ and the observation density $p(\mathbf{Z}|\mathbf{X})$ are non-Gaussian, and there is no closed-form algorithm to evaluate the posterior. Factored sampling provides a way of approximating the posterior, using a random number generator to sample from a prior for curve-shape. Random sampling methods were used in chapter 8 as a way of visualising distributions; here they form an integral part of the algorithm.

The factored sampling algorithm generates a random variate $\tilde{\mathbf{X}}$ from a distribution $\tilde{p}(\tilde{\mathbf{X}})$ that approximates the posterior $p(\mathbf{X}|\mathbf{Z})$. First a sample set $\{\mathbf{s}^{(1)}, \ldots, \mathbf{s}^{(N)}\}$ is generated[1] from the prior density $p(\mathbf{X})$ and then an index $n \in \{1, \ldots, N\}$ is chosen with probability $\pi^{(n)}$, where

$$\pi^{(n)} = \frac{p_z(\mathbf{s}^{(n)})}{\sum_{j=1}^{N} p_z(\mathbf{s}^{(j)})}$$

and

$$p_z(\mathbf{s}) = p(\mathbf{Z}|\mathbf{X} = \mathbf{s}),$$

---

[1] This can be done for example using (8.8) on page 164. Note that the presence of clutter causes $p(\mathbf{Z}|\mathbf{X})$ to be non-Gaussian, but the prior $p(\mathbf{X})$ may still happily be Gaussian, and that is what is assumed in later examples.

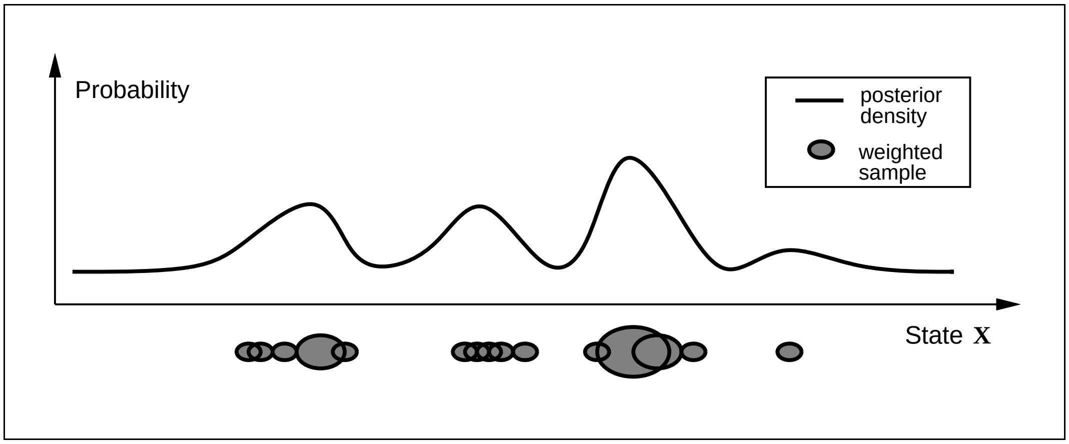


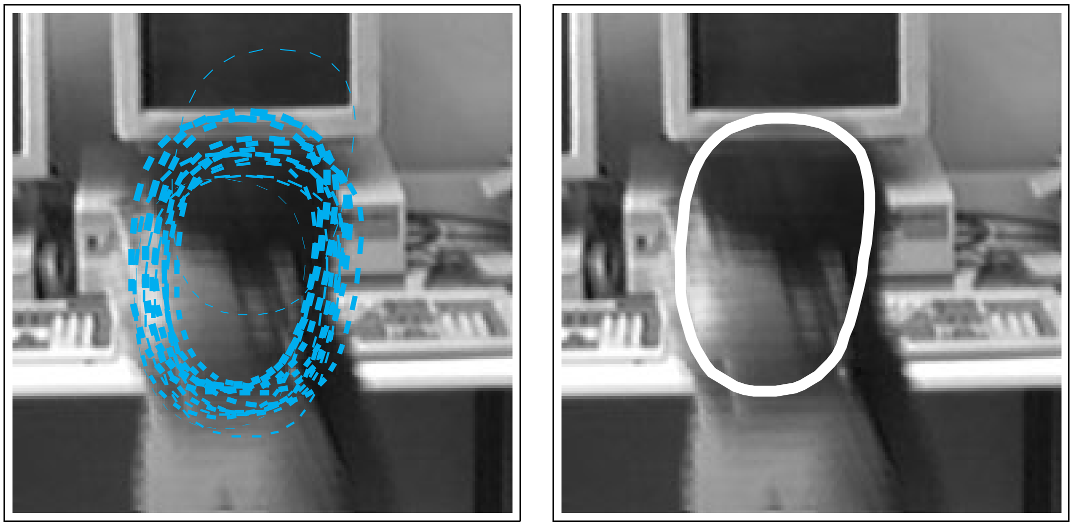
Figure 12.2: Factored sampling. *A set of points* $\mathbf{s}^{(n)}$*, the centres of the blobs in the figure, is sampled randomly from a prior density* $p(\mathbf{X})$*. Each sample is assigned a weight* $\pi^{(n)}$ *(depicted by blob area) in proportion to the value of the observation density* $p(\mathbf{Z}|\mathbf{X} = \mathbf{s}^{(n)})$*. The weighted point set then serves as a representation of the posterior density* $p(\mathbf{X}|\mathbf{Z})$*, suitable for sampling. The one-dimensional case illustrated here extends naturally to the practical case that the density is defined over several position and shape variables.*

the conditional observation density. The value $\tilde{\mathbf{X}} = \mathbf{s}^{(n)}$ chosen in this fashion has a distribution which approximates the posterior $p(\mathbf{X}|\mathbf{Z})$ increasingly accurately as N increases (figure 12.2).

Note that arbitrary posterior expectations $\mathcal{E}[g(\mathbf{X})|\mathbf{Z}]$ can be generated directly from the samples $\{\mathbf{s}^{(n)}\}$ by weighting with $p_z(\mathbf{s})$ to give:

$$\mathcal{E}[g(\mathbf{X})|\mathbf{Z}] \approx \frac{\sum_{n=1}^{N} g(\mathbf{s}^{(n)}) p_z(\mathbf{s}^{(n)})}{\sum_{j=1}^{N} p_z(\mathbf{s}^{(j)})}. \tag{12.1}$$

For example, the mean can be estimated using $g(\mathbf{X}) = \mathbf{X}$ (illustrated in figure 12.3) and the second moment using $g(\mathbf{X}) = \mathbf{X}\mathbf{X}^T$. In the case that the density $p(\mathbf{Z}|\mathbf{X})$ is normal, the mean obtained by factored sampling is consistent with an estimate obtained more conventionally, and efficiently, from linear least-squares estimation. An example of the application of the factored sampling algorithm to locate objects is given in figure 12.4. A prior distribution is used which is quite diffuse, covering most of the image area, and allowing a wide range of variation of orientation and shape. Given a suitable observation density, the posterior distribution shown is strongly peaked at



(a) (b)

Figure 12.3: Sample-set representation of shape distributions. *The sample-set representation of probability distributions, illustrated in one dimension in figure 12.2, is illustrated here (a) as it applies to the distribution of a multi-dimensional curve parameter* $\mathbf{X}$*. Each sample* $\mathbf{s}^{(n)}$ *is shown as a curve (of varying position and shape) with a thickness proportional to the weight* $\pi^{(n)}$*. The weighted mean of the sample set (b) serves as an estimator of the distribution mean.*

the locations of several genuine objects and also a few fairly convincing frauds.

12.2 The CONDENSATION algorithm

The CONDENSATION algorithm is based on factored sampling but extended to apply iteratively to images in a sequence, taken at successive times t_k. In chapter 10 the Kalman filter was used to incorporate prediction, using a dynamical model, into the curve-fitting process. CONDENSATION is the analogous extension to factored sampling, and in the examples described later the dynamical model used is exactly that developed in chapters 9 and 11. In fact much more general classes of dynamical models can be used within the algorithm and these are discussed in the bibliographic notes.

The state of the modelled object at time t_k, denoted $\mathcal{X}(t_k)$ earlier, will now be

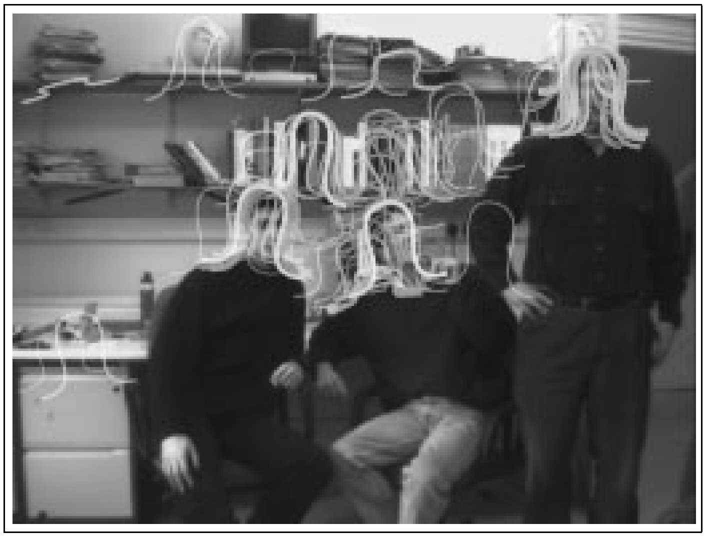


Figure 12.4: Sample-set representation of posterior shape distribution *for a curve with parameters* $\mathbf{X}$*, modelling a head outline. Each sample* $\mathbf{s}^{(n)}$ *is shown as a curve (of varying position and shape) with a thickness and intensity proportional to the weight* $\pi^{(n)}$*. The prior is uniform over translation, and a constrained Gaussian in the remainder of its affine shape-space. (Figure taken from (MacCormick and Blake, 1998).)*

denoted as $\mathcal{X}_k$, for compactness. Its "history" is $\underline{\mathcal{X}}_k = \{\mathcal{X}_1, \ldots, \mathcal{X}_k\}$. Similarly the image observation at time t_k, previously denoted $\mathbf{Z}(t_k)$, will be denoted $\mathbf{Z}_k$ with history $\underline{\mathbf{Z}}_k = \{\mathbf{Z}_1, \ldots, \mathbf{Z}_k\}$.

The process at each time-step is a self-contained iteration of factored sampling, so the output of an iteration will be a weighted, time-stamped sample set, denoted $\{\mathbf{s}_k^{(n)},\ n = 1, \ldots, N\}$ with weights $\pi_k^{(n)}$, representing approximately the conditional state density $p(\mathcal{X}_k|\underline{\mathbf{Z}}_k)$ at time t_k. How is this sample set obtained? The process must begin with a prior density and the effective prior for time-step t_k should be $p(\mathcal{X}_k|\underline{\mathbf{Z}}_{k-1})$. This prior is multi-modal in general and no functional representation of it is available. It is derived from the sample-set representation $\{(\mathbf{s}_{k-1}^{(n)}, \pi_{k-1}^{(n)}),\ n = 1, \ldots, N\}$ of $p(\mathcal{X}_{k-1}|\underline{\mathbf{Z}}_{k-1})$, the output from the previous time-step, to which prediction must then be applied.

The iterative process as applied to sample sets, depicted in figure 12.5, mirrors the continuous diffusion process in figure 12.1. At the top of the diagram, the output

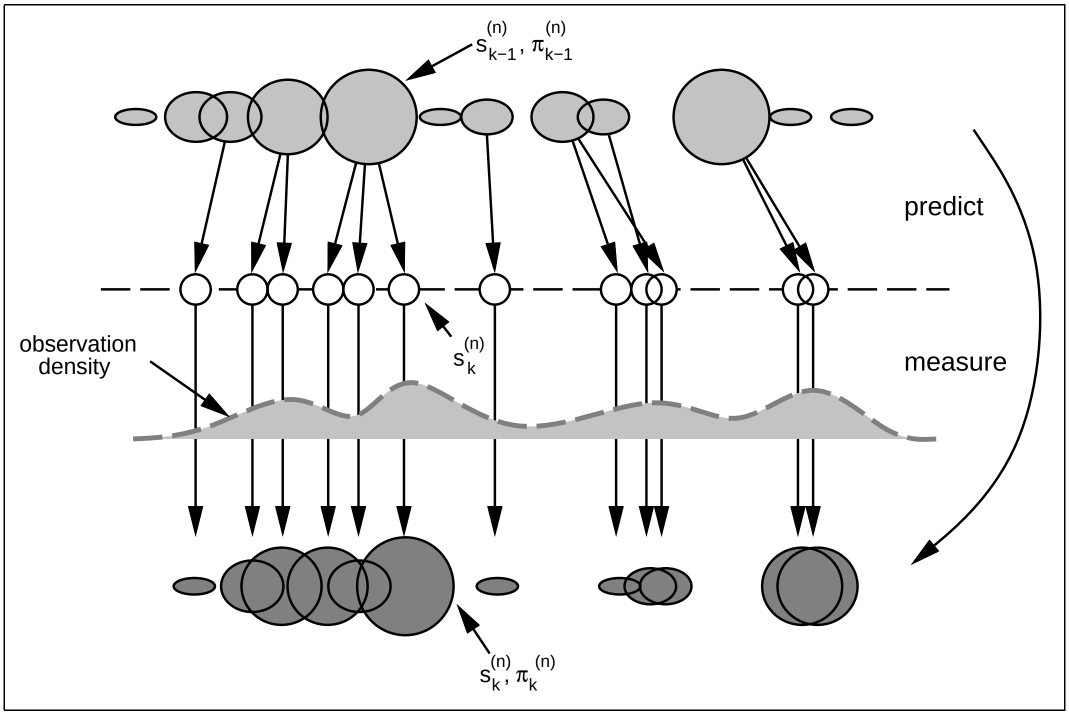


Figure 12.5: One time-step in the CONDENSATION algorithm. *The drift and diffusion steps of the probabilistic propagation process of figure 12.1 are combined into a single prediction stage.*

from time-step t_{k-1} is the weighted sample set $\{(\mathbf{s}_{k-1}^{(n)}, \pi_{k-1}^{(n)}),\ n = 1, \ldots, N\}$. The aim is to maintain, at successive time-steps, sample sets of fixed size N, so that the algorithm can be guaranteed to run within a given computational resource. The first operation therefore is to sample (with replacement) N times from the set $\{\mathbf{s}_{k-1}^{(n)}\}$, choosing a given element with probability $\pi_{k-1}^{(n)}$. Some elements, especially those with high weights, may be chosen several times, leading to identical copies of elements in the new set. Others with relatively low weights may not be chosen at all.

Each element chosen from the new set is now subjected to a predictive step. This corresponds to sampling from the distribution $p(\mathcal{X}_k|\mathcal{X}_{k-1})$, and for the second-order AR models described in chapter 9 the sampling equation is given by (9.17) on page 204. At this stage, the sample set $\{\mathbf{s}_k^{(n)}\}$ for the new time-step has been generated but, as yet, without its weights; it is approximately a fair random sample from the effective prior density $p(\mathcal{X}_k|\underline{\mathbf{Z}}_{k-1})$ for time-step t_k. Finally, the observation step from factored sampling is applied, generating weights from the observation density $p(\mathbf{Z}_k|\mathcal{X}_k)$ to obtain the sample-set representation $\{(\mathbf{s}_k^{(n)}, \pi_k^{(n)})\}$ of state density for time t_k.

Figure 12.6 gives a synopsis of the algorithm. Note the use of *cumulative* weights $c_{k-1}^{(j)}$ (constructed in step 3) to achieve efficient sampling in step 1. After any time-step, it is possible to "report" on the current state, for example by evaluating some moment of the state density:

$$\mathcal{E}[g(\mathcal{X}_k)] = \sum_{n=1}^{N} \pi_k^{(n)} g\left(\mathbf{s}_k^{(n)}\right).$$

In later examples the mean position is displayed using $g(\mathcal{X}) = \mathcal{X}$.

One of the striking properties of the Condensation algorithm is its simplicity, compared with the Kalman filter, despite its generality. Largely this is because it is not necessary in the Condensation framework to propagate covariance explicitly.

12.3 An observation model

The observation process defined by $p(\mathbf{Z}_k|\mathcal{X}_k)$ is assumed here to be stationary in time (though the Condensation algorithm does not necessarily demand this) so a static function $p(\mathbf{Z}|\mathcal{X})$ needs to be specified. It is also assumed here that observations depend only on position and shape, not on velocities, so that

$$p(\mathbf{Z}|\mathcal{X}) = p(\mathbf{Z}|\mathbf{X}).$$

One-dimensional observations in clutter

The observation density $p(\mathbf{Z}|\mathbf{X})$ for curves in clutter is quite different to the Gaussian approximation used in chapter 8, so for clarity a simplified one-dimensional form is described first. In one dimension, observations reduce to a set of scalar positions $\{\mathbf{z} = (z_1, z_2, \ldots, z_M)\}$ and the observation density has the form $p(\mathbf{z}|x)$ where x is one-dimensional position. The multiplicity of measurements reflects the presence of clutter so either one of the events

$$\phi_m = \{\text{true measurement is } z_m\},\ m = 1, \ldots, M$$

Iterate

From the "old" sample set $\{\mathbf{s}_{k-1}^{(n)}, \pi_{k-1}^{(n)}, c_{k-1}^{(n)}, n = 1, \ldots, N\}$ at time-step t_{k-1}, construct a "new" sample set $\{\mathbf{s}_k^{(n)}, \pi_k^{(n)}, c_k^{(n)}, n = 1, \ldots, N\}$ for time t_k.

Construct the n^{th} of N new samples as follows:

1. **Select** a sample ${\mathbf{s}'}_k^{(n)}$ as follows:
    (a) generate a random number $r \in [0, 1]$, uniformly distributed.
    (b) find, by binary subdivision, the smallest j for which $c_{k-1}^{(j)} \geq r$
    (c) set ${\mathbf{s}'}_k^{(n)} = \mathbf{s}_{k-1}^{(j)}$
2. **Predict** by sampling from

$$p(\mathcal{X}_k | \mathcal{X}_{k-1} = {\mathbf{s}'}_k^{(n)})$$

   to choose each $\mathbf{s}_k^{(n)}$. For instance, in the case that the dynamics are governed by a linear AR process, the new sample value may be generated as: $\mathbf{s}_k^{(n)} = A\,{\mathbf{s}'}_k^{(n)} + (I - A)\overline{\mathcal{X}} + B\mathbf{w}_k^{(n)}$ where $\mathbf{w}_k^{(n)}$ is a vector of standard normal random variates, and BB^T is the process noise covariance.

3. **Measure** and weight the new position in terms of the measured features $\mathbf{Z}_k$:

$$\pi_k^{(n)} = p(\mathbf{Z}_k | \mathcal{X}_k = \mathbf{s}_k^{(n)})$$

   then normalise so that $\sum_n \pi_k^{(n)} = 1$ and store together with cumulative probability as $(\mathbf{s}_k^{(n)}, \pi_k^{(n)}, c_k^{(n)})$ where

$$\begin{aligned} c_k^{(0)} &= 0, \\ c_k^{(n)} &= c_k^{(n-1)} + \pi_k^{(n)} \quad \text{for} \quad n = 1, \ldots, N. \end{aligned}$$

Figure 12.6: The CONDENSATION algorithm.

occurs, or else the target object is not visible with probability $q = 1 - \sum_m P(\phi_m)$. Now the observation density can be expressed as

$$p(\mathbf{z}|x) = q\, p(\mathbf{z}|\text{clutter}) + \sum_{m=1}^{M} p(\mathbf{z}|x, \phi_m) P(\phi_m).$$

A reasonable functional form for this can be obtained by making some specific assumptions: that $P(\phi_m) = p$ for all m features[2], that the clutter is a Poisson process along the line with spatial density λ and that any true target measurement is unbiased and normally distributed with standard deviation σ. This leads to

$$p(\mathbf{z}|x) \propto 1 + \frac{1}{\sqrt{2\pi}\sigma\alpha} \sum_m \exp -\frac{\nu_m^2}{2\sigma^2} \tag{12.2}$$

where $\alpha = q\lambda$ and $\nu_m = z_m - x$, and is illustrated in figure 12.7. Peaks in the density function correspond to measured features and the state density will tend to be reinforced in the CONDENSATION algorithm at such points. The background level reflects the possibility that the true target has not been detected at all, and allows a good hypothesis to survive a transitory failure of observations due, for example, to occlusion of the tracked object. The parameters σ (units of distance) and α (units of inverse distance) must be chosen, though in principle they could be estimated from data by observing measurement error σ and both the density of clutter λ and probability of non-detection q.

Considerable economy can be applied, in practice, in the evaluation of the observation density. Given a hypothesised position x in the "observation" step (figure 12.6) it is not necessary to attend to all features $z_1, \ldots, z_M$. Any ν_m for which

$$\frac{1}{\sqrt{2\pi}\sigma\alpha} \exp -\frac{\nu_m^2}{2\sigma^2} \ll 1$$

can be neglected and this sets a search window around the position x outside which measurements can be ignored. For practical values of the constants the search window will have a width of a few σ.

Note that the density $p(\mathbf{z}|x)$ represents the information about x given a fixed number M of measurements. Potentially, the *event* ψ_M that there are M measurements,

---

[2] There could be some benefit in allowing the $P(\phi_m)$ to vary with m to reflect varying degrees of feature affinity, based on contrast, colour or orientation.

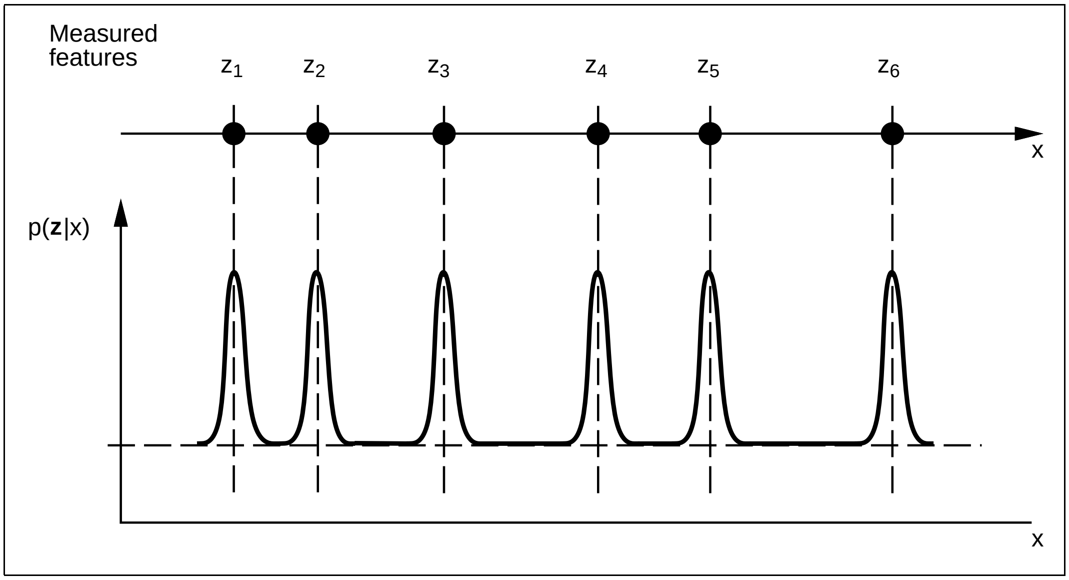


Figure 12.7: One-dimensional observation model. *A probabilistic observation model allowing for clutter and the possibility of missing the target altogether is specified here as a conditional density $p(\mathbf{z}|x)$.*

regardless of the actual *values* of those measurements, also constitutes information about x. However, we can reasonably assume that

$$P(\psi_M|x) = P(\psi_M),$$

for instance because x is assumed to lie always within the image window. In that case, by Bayes' theorem,

$$p(x|\psi_M) = p(x)$$

— the event ψ_M provides no additional information about the position x. (If x is allowed also to fall outside the image window then the event ψ_M *is* informative: a value of M well above the mean value for the background clutter enhances the probability that x lies within the window.)

Two-dimensional observations

In a two-dimensional image, the set of observations $\mathbf{Z}$ is, in principle, the entire set of features visible in the image. However, an important aspect of achieving real-time

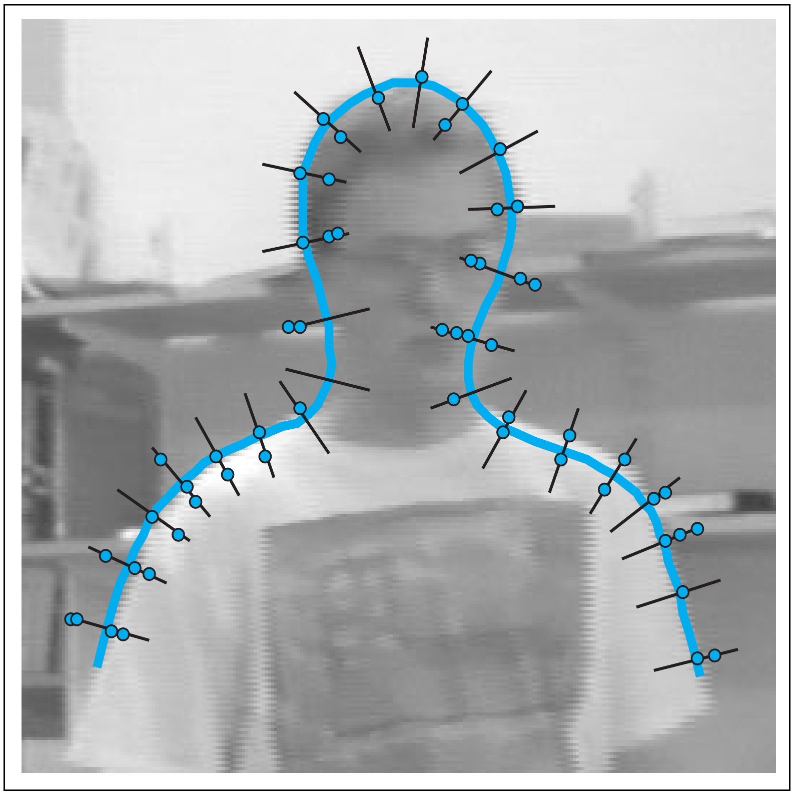


Figure 12.8: Observation process. *The thick line is a hypothesised shape, represented as a parametric spline curve. The spines are curve normals along which high-contrast features (dots) are sought.*

performance has been the restriction of measurement to a sparse set of lines normal to the tracked curve. These two apparently conflicting ideas are hard to resolve and some discussion is given elsewhere (see bibliographic notes). One good choice is simply to construct the two-dimensional observation density as the product of one-dimensional densities (12.2), evaluated independently along M curve normals as in figure 12.8. Note that *locally* $p(\mathbf{Z}|\mathbf{X})$ is approximately Gaussian, especially if constants are chosen so that σ is small and only one feature typically falls within each search region. As $\mathbf{X}$ varies across the entire image, however, the multi-modality of the observation density is apparent.

12.4 Applications of the Condensation algorithm

Four examples are shown here of the practical efficacy of the Condensation algorithm. MPEG versions of some results are available on the web page for the book.

Tracking a multi-modal distribution

The ability of the Condensation algorithm to represent multi-modal distributions is demonstrated in a sequence of a cluttered room containing three people each facing the camera (figure 12.9). One of the people moves from right to left, in front of the other two. The shape-space for tracking is built from a hand-drawn template of head and shoulders (figure 12.8) which is then allowed to deform via planar affine transformations. A Kalman filter contour-tracker with default motion parameters is able to track a single moving person just well enough to obtain a sequence of outline curves that is usable as training data. Given the high level of clutter, adequate performance with the Kalman filter is obtained here by means of statistical background

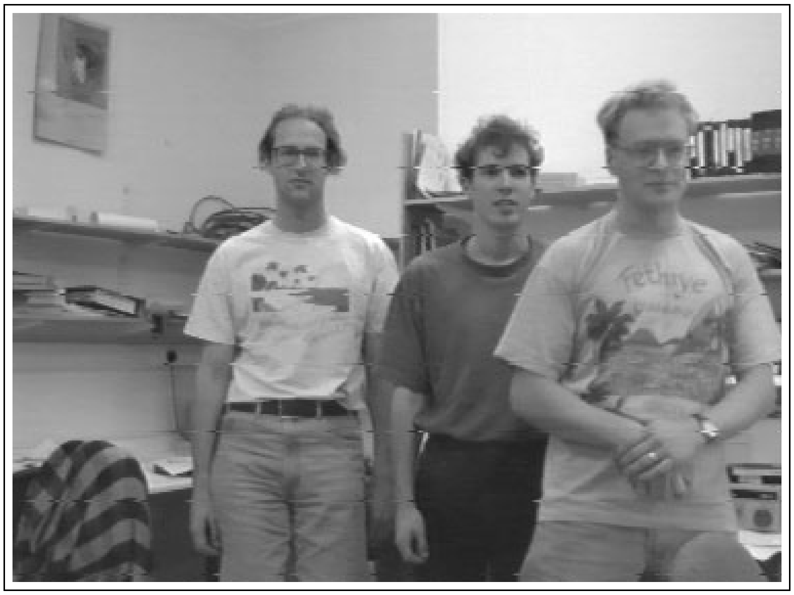


Figure 12.9: Tracking three people in a cluttered room. *The first frame of a sequence in which one figure moves from right to left in front of two stationary figures.*

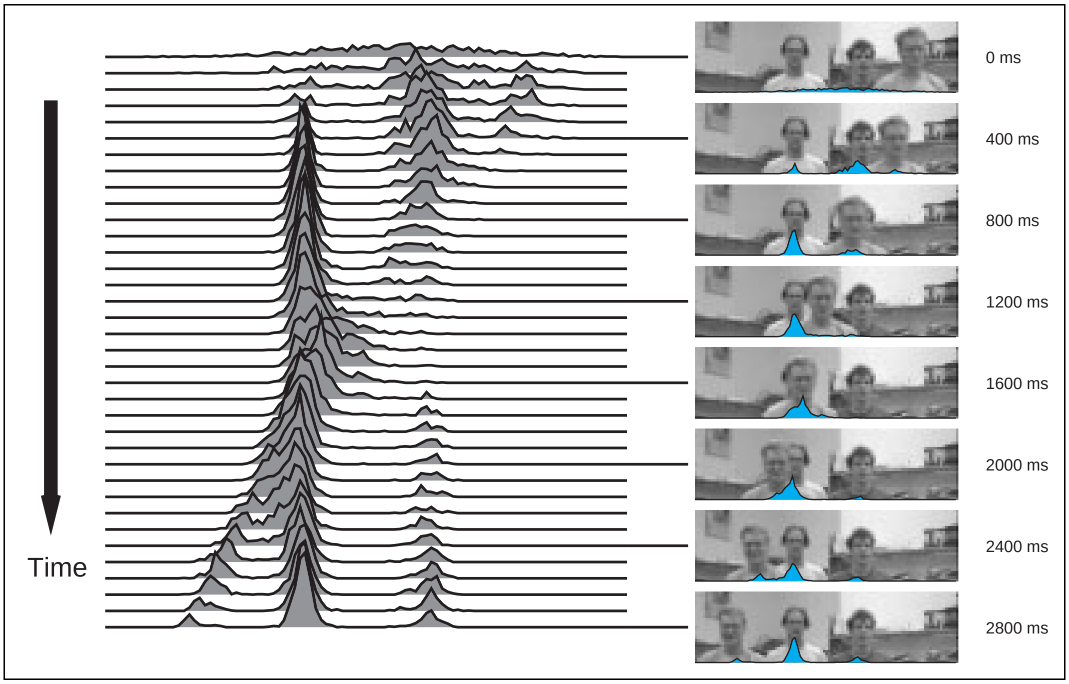


Figure 12.10: Tracking with a multi-modal state density. *An approximate depiction of the state density is shown, computed by smoothing the distribution of point masses* $\mathbf{s}_k^{(1)}, \mathbf{s}_k^{(2)}, \ldots$ *in the* CONDENSATION *algorithm. The density is, of course, multi-dimensional; its projection onto the horizontal translation axis is shown here. The initial distribution is roughly Gaussian but this rapidly evolves to acquire peaks corresponding to each of the three people in the scene. The rightmost peak drifts leftwards, following the moving person, coalescing with and separating from the other two peaks as it moves. Having specified a tracker for* one *person we effectively have, for free, a multi-person tracker, owing to the innate ability of the* CONDENSATION *algorithm to maintain multiple hypotheses.*

subtraction. It transpires, for this particular training set, that the learned motions comprise primarily horizontal translation, with vertical translation and horizontal and vertical shear present to a lesser degree.

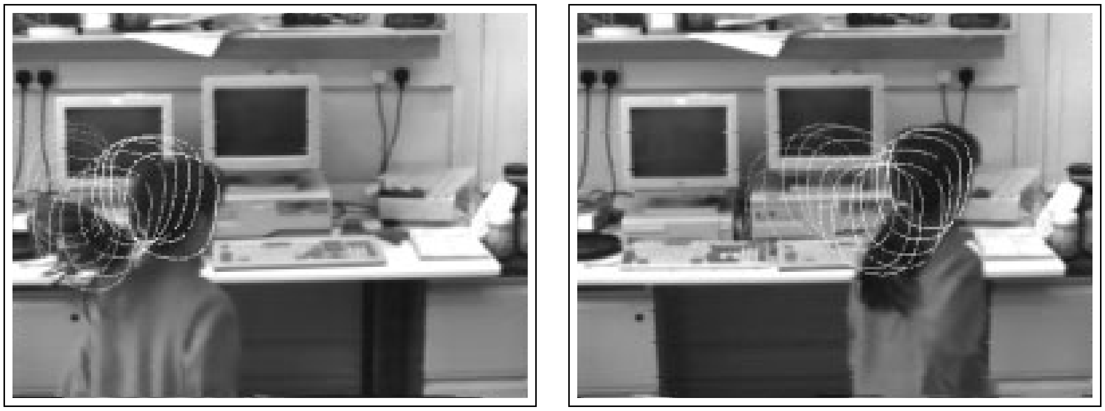
The learned shape and motion model can be installed as $p(\mathcal{X}_k|\mathcal{X}_{k-1})$ in the CONDENSATION algorithm which is now run on a test sequence but *without* the benefit of background modelling, so that the background clutter is now visible to the tracker. Figure 12.10 shows how the state density evolves as tracking progresses. Initialisation is performed simply by iterating the stochastic model, in the absence of measurements, to its steady state and it can be seen that this corresponds, at time t_0, to a roughly Gaussian distribution, as expected. The distribution rapidly collapses down to three peaks which are then maintained appropriately even during temporary occlusion. Although the tracker was designed to track just one person, the CONDENSATION algorithm takes account of the motion of all three; the ability to represent multi-modal distributions effectively provides multiple-hypothesis capability. Tracking is based on frame rate (40 ms) sampling in this demonstration and distributions are plotted in the figure for alternate frames. A distribution of $N = 1000$ samples per time-step is used.

Tracking rapid motions through clutter

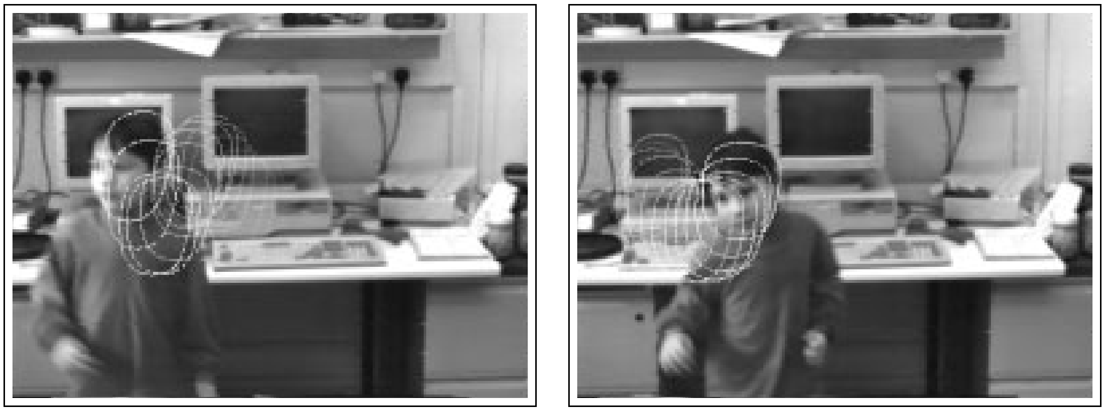
The ability to track more agile motion, still against clutter, is demonstrated by a sequence of a girl dancing vigorously to a Scottish reel. The shape-space for tracking is planar affine, based on a hand-drawn template curve for the head outline. The training sequence consists of dancing against a largely uncluttered background, tracked by a Kalman filter contour-tracker with default dynamics to record 140 fields (2.8 seconds) of tracked head positions, the most that can be tracked before losing lock. Those 140 fields are sufficient to learn a bootstrap motion model which then allows the Kalman filter to track the training data for 800 fields (16 seconds) before loss of lock. The motion model obtained from these 800 fields can now be applied to test data that includes clutter.

Figure 12.11 shows some stills from the test sequence, with a trail of preceding head positions to indicate motion. The motion is primarily translation, with some horizontal shear apparent as the dancer turns her head. Representing the state density with $N = 100$ samples at each time-step proves just sufficient for successful tracking. As in the previous example, a prior density can be computed as the steady state of the motion model and, in this case, that yields a prior for position that spreads across most of the image area, as might be expected given the range of the dance. Such a broad distribution cannot effectively be represented by just $N = 100$ samples. One alternative is to increase N in the early stages of tracking, and this is demonstrated later. Alternatively, the prior can be based on a narrower distribution whose centre is positioned by hand over the object at time t_0, and that is what has been done here. (Observation parameters were $\alpha = 0.005$, $\sigma = 7$ with 18 normals.)

Figure 12.12 shows the motion of the centroid of the estimates head position as tracked both by the CONDENSATION algorithm and by a Kalman filter using the



field 91 (1820 ms) field 121 (2420 ms)



field 221 (4420 ms) field 265 (5300 ms)

Figure 12.11: Tracking agile motion in clutter. *The test sequence consists of 500 fields (10 seconds) of agile dance against a cluttered background. The dancer's head is tracked through the sequence. Several representative fields are shown here, each with a trail of successive mean tracked head positions at intervals of 40 ms. The* CONDENSATION *algorithm used* $N = 100$ *samples per time-step to obtain these results.*

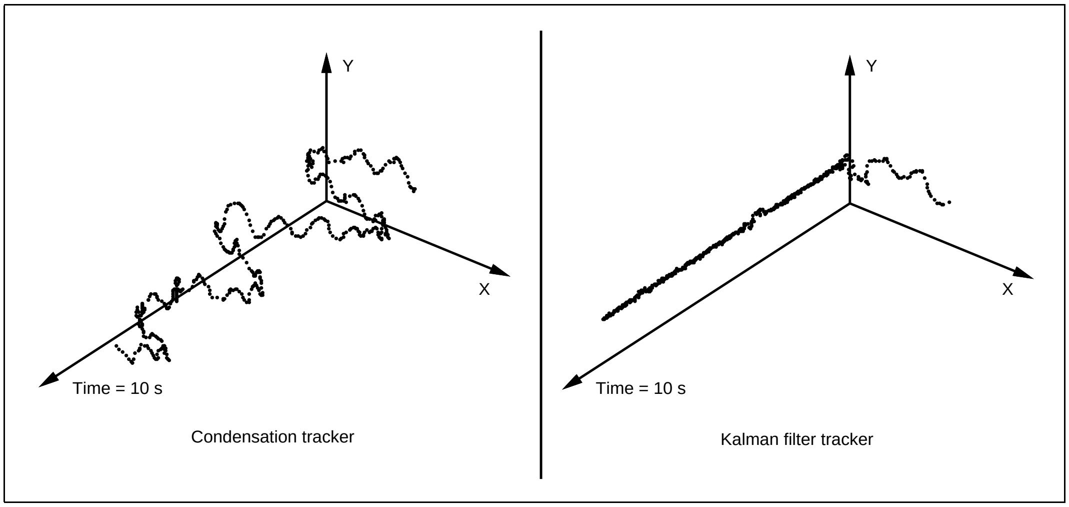
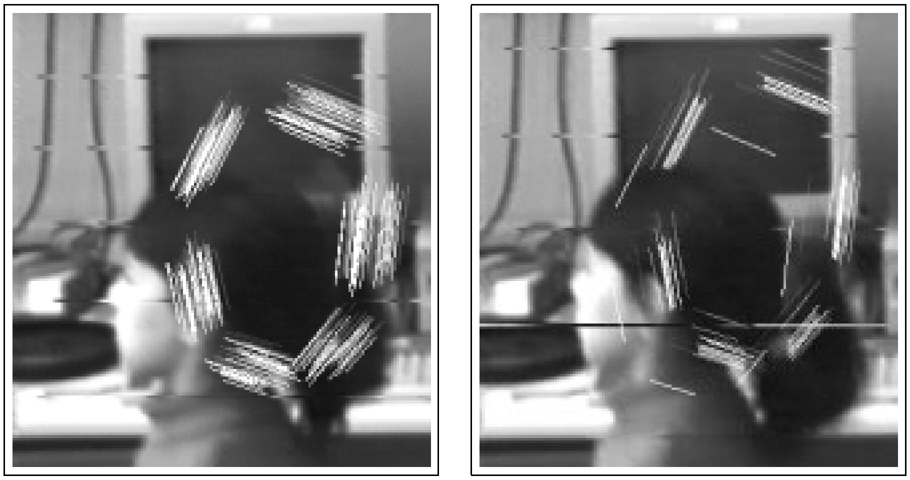


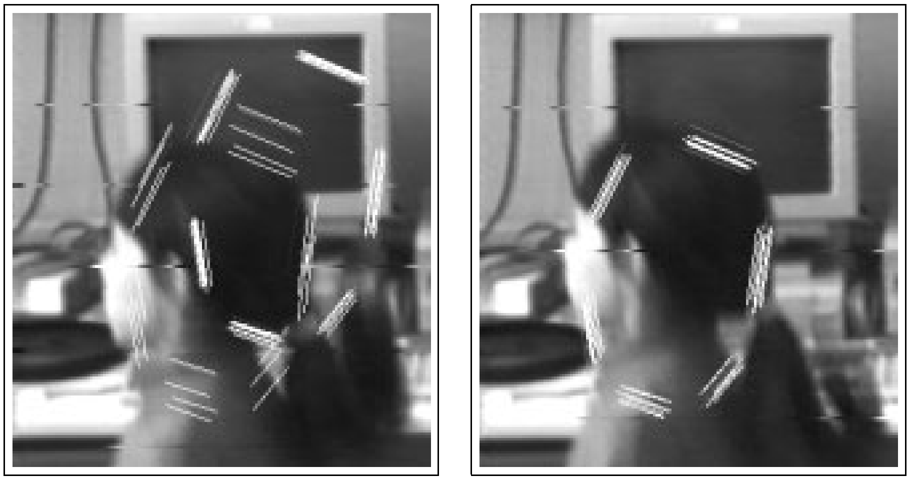
Figure 12.12: The Condensation tracker succeeds where a Kalman filter fails. *The estimated centroid for the sequence shown in figure 12.11 is plotted against time for the entire 500 field sequence, as tracked first by the* CONDENSATION *tracker, then by a comparable Kalman filter tracker. The* CONDENSATION *algorithm correctly estimates the head position throughout the sequence. The Kalman filter tracks briefly, but is soon distracted by clutter.*

same motion model. The CONDENSATION tracker correctly estimated head position throughout the sequence, but after about 40 fields (0.80 s), the Kalman filter is distracted by clutter, never to recover.

Given that there is only one moving person now, unlike the previous example in which there were three, it might seem that a uni-modal representation of the state density would suffice. This is emphatically not the case. The facility to represent multiple modes is crucial to robustness as figure 12.13 illustrates. The figure shows how the distribution becomes misaligned (at 900 ms), reacting to the distracting form of the computer screen. After 20 ms the distribution splits into two distinct peaks, one corresponding to clutter (the screen), one to the dancer's head. At this point the clutter peak actually has the higher posterior probability — a uni-modal tracker, for instance a Kalman filter, would almost certainly discard the lower peak, rendering it unable to recover. The CONDENSATION algorithm however, capable as it is of carrying several hypotheses simultaneously, does recover rapidly as the clutter peak decays for lack of confirmatory observation, leaving just one peak corresponding to the dancer after 60 ms.



field 45 (900 ms) field 46 (920 ms)



field 47 (940 ms) field 48 (960 ms)

Figure 12.13: Recovering from tracking failure. *Detail from 4 consecutive fields of the sequence illustrated in figure 12.11. Each sample from the distribution is plotted on the image, with intensity scaled to indicate its posterior probability. (Most of the $N = 100$ samples have too low a probability to be visible in this display.) At field 45 the distribution is misaligned, and has begun to diverge. At fields 46 and 47 it has split into two distinct peaks, the larger attracted to background clutter, but converges back onto the dancer at field 48.*

Tracking an articulated object

The preceding sequences show motion taking place in affine shape-spaces of just 6 dimensions. High dimensionality is one of the factors, in addition to agility and clutter, that makes tracking hard. In order to demonstrate tracking performance in higher dimensions, we use a test sequence of a hand translating, rotating, and flexing its fingers independently, over a highly cluttered desk scene (figure 12.14). Figure 12.15 shows just how severe the clutter problem is — the hand is immersed in a dense field of edges.

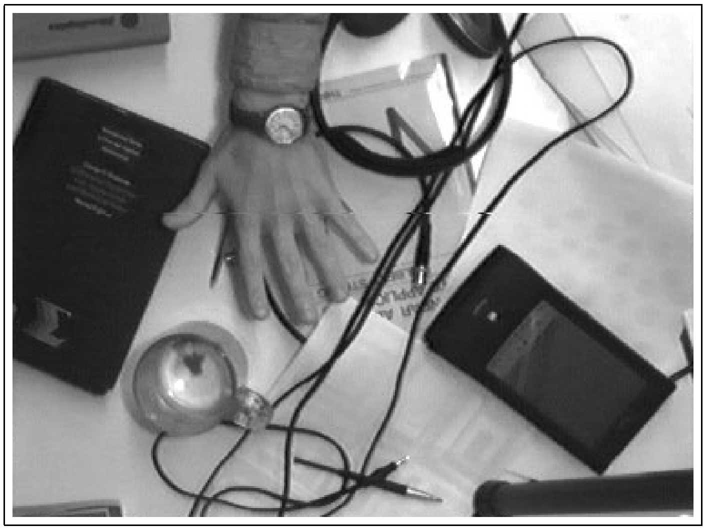


Figure 12.14: A hand moving over a cluttered desk. *Field 0 of a 500 field (10 second) sequence in which the hand translates, rotates, and the fingers and thumb flex independently.*

A model of shape and motion has been learned from training sequences of hand motion against a plain background, tracked by Kalman filter (using signed edges to help to disambiguate finger boundaries). The procedure comprised several stages, a creative assembly of methods from the available toolkit for generating a shape-space and learning dynamics.

1. **Shape-space** is constructed from 6 templates drawn around the hand with the palm in a fixed orientation and with the fingers and thumb in various configurations. The 6 templates combine linearly to form a 5-dimensional space

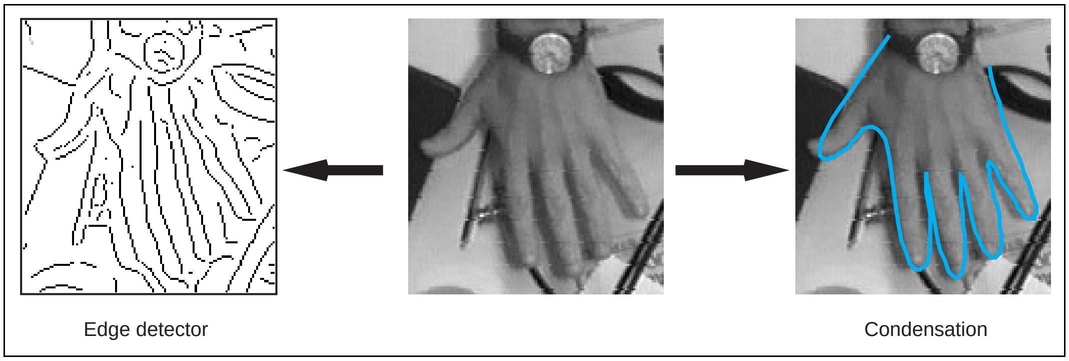


Figure 12.15: Severe clutter. *Detail of one field from the test sequence shows the high level of potential ambiguity. Output from a directional Gaussian edge detector shows that there are many clutter edges present as potential distractors.*

of deformations which are then added to the space of translations to form a 7-dimensional shape-space.

2. **Default dynamics** in the shape-space above are adequate to track a clutter-free training sequence of 600 frames in which the palm of the hand maintains an approximately fixed attitude.

3. **Principal components analysis:** the sequence of 600 hand outlines is replicated with each hand contour rotated through 90 degrees, and the sequences concatenated to give a sequence of 1200 deformations. Projecting out the translational component of motion, the application of Principal Component Analysis (PCA) to the sequence of residual deformations of the 1200 contours establishes a 10-dimensional space that accounted almost entirely for deformation. This is then combined with the translational space to form a 12-dimensional shape-space that accounts both for the flexing of fingers and thumb and also for rotations of the palm.

4. **Bootstrapping:** a Kalman filter with default dynamics in the 12-dimensional shape-space is sufficient to track a training sequence of 800 fields of the hand translating, rotating, and flexing fingers and thumb slowly. This is used to learn a model of motion.

5. **Re-learning:** that motion model is installed in a Kalman filter and used to track

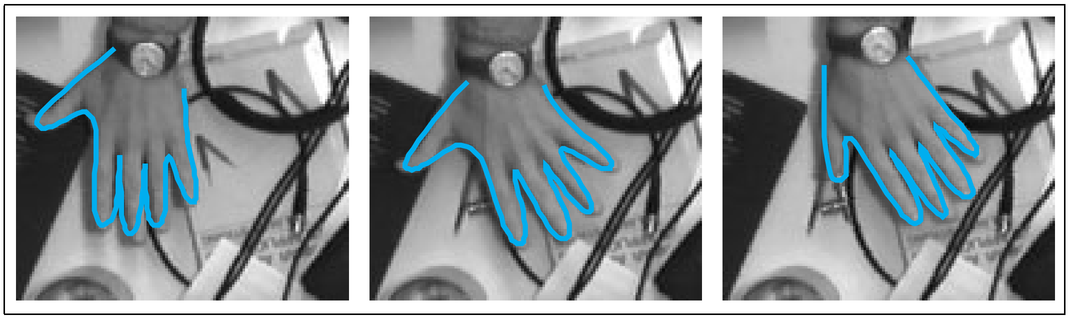


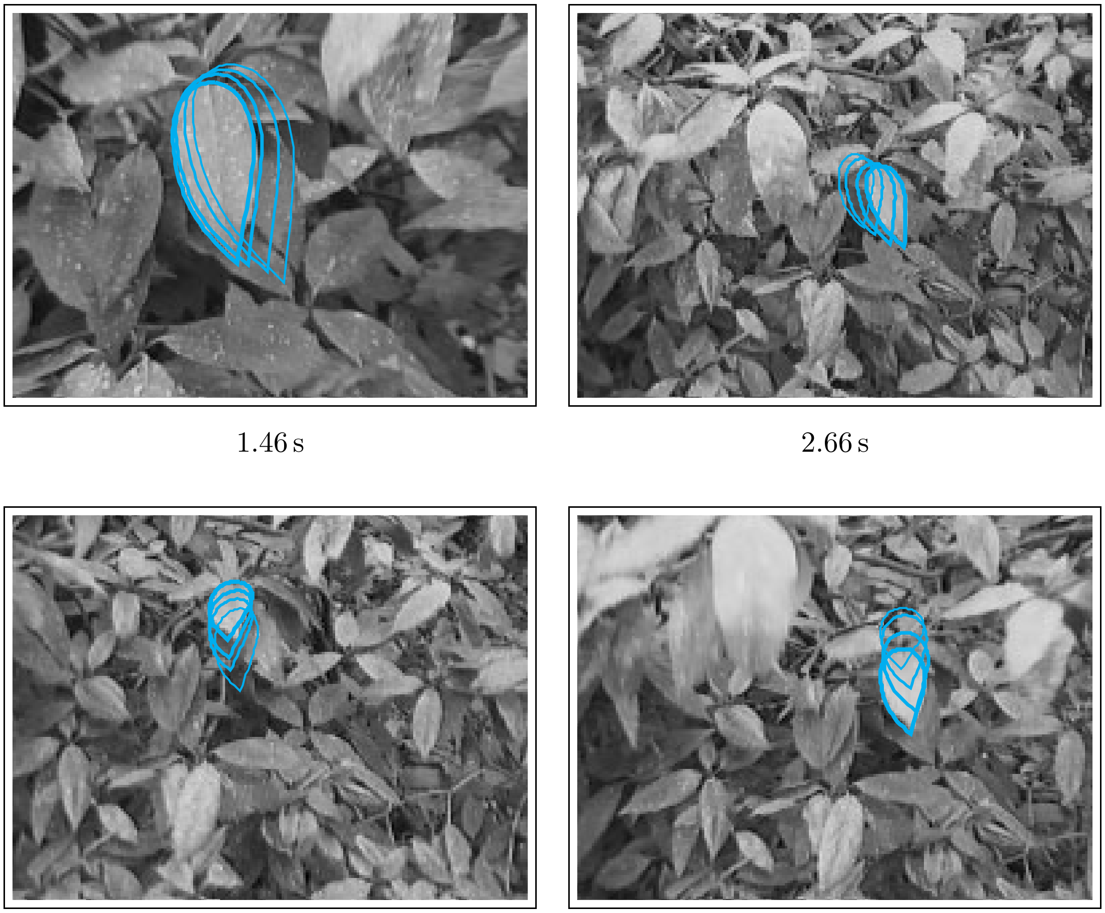
Figure 12.16: Tracking a flexing hand across a cluttered desk. *Representative stills from a 500 field (10 second) sequence show a hand moving over a highly cluttered desk scene. The fingers and thumb flex independently, and the hand translates and rotates. Here the* Condensation *algorithm uses $N = 1500$ samples per time-step initially, dropping over 4 fields to $N = 500$ for the tracking of the remainder of the sequence. The mean configuration of the contour is displayed.*

> another, faster training sequence of 800 fields. This allows a model for more agile motion to be learned, which is then used in a high-performance Condensation tracker.

Figure 12.16 shows detail of a series of images from a tracked, 500 field test sequence. The initial state density is simply the steady state of the motion model, obtained by allowing the filter to iterate in the absence of observations. Tracker initialisation is facilitated by using more samples per time-step ($N = 1500$) at time t_0, falling to 500 over the first 4 fields. The rest of the sequence is tracked using $N = 500$. As with the previous example of the dancer, clutter may distract the tracker but the ability to represent multi-modal state density means that tracking can recover.

Tracking a camouflaged object

Finally, the ability of the algorithm to track rapid motion against background distraction is demonstrated in an extreme case: that background objects actually mimic the tracked object. A 12 second (600 field) sequence shows a bush blowing in the wind, the task being to track one particular leaf. A template is drawn by hand around a still of one chosen leaf and allowed to undergo affine deformations during tracking. Given that a clutter-free training sequence cannot be obtained in this case, the motion model is again learned by means of a bootstrap procedure. A tracker with hand-specified



1.46 s

2.66 s

5.54 s

7.30 s

Figure 12.17: Tracking with camouflage. *The aim is to track a single camouflaged moving leaf in this 12 second sequence of a bush blowing in the wind. Despite the heavy clutter of distractors which actually mimic the foreground object, and occasional violent gusts of wind, the chosen foreground leaf is successfully tracked throughout the sequence. Representative stills depict mean contour configurations, with preceding tracked leaf positions plotted at 40 ms intervals to indicate motion.*

dynamics proves capable of tracking the first 150 fields of a training sequence before losing the leaf, and those tracked positions allow a first approximation to the model to be learned. Installing that in a CONDENSATION tracker, the entire sequence can be tracked, though with occasional misalignments. Finally, a second learned model is capable of tracking accurately the entire 12 second training sequence. Despite occasional violent gusts of wind and temporary obscuration by another leaf, the CONDENSATION algorithm successfully follows the object, as figure 12.17 shows. In fact, tracking is accurate enough using $N = 1200$ samples to separate the foreground leaf from the background reliably, an effect which can otherwise only be achieved using "blue-screening." Having obtained the model iteratively as above, further sequences can be tracked without further training. With $N = 1200$ samples per time-step the tracker runs at 6.5 Hz on a SGI Indy SC4400 200 MHz workstation. Reducing this to $N = 100$ increases processing speed to video field rate (50 Hz), at the cost of occasional misalignments in the mean configuration of the contour. (Observation parameters are $\alpha = 0.022$, $\sigma = 3$ with 21 normals.)

Bibliographic notes

There has been much written about non-linear filtering algorithms, for handling non-Gaussian probability densities. The Extended Kalman filter (Gelb, 1974; Bar-Shalom and Fortmann, 1988; Jacobs, 1993), is a linearised approximation designed to deal with non-linearities in dynamics and/or sensors. It effectively approximates the non-Gaussian state density as a Gaussian. Bayesian multiple-hypothesis filters and approximate variants include the PDAF and JPDAF (Bar-Shalom and Fortmann, 1988) and can be applied to motion correspondence (Cox, 1993) and visual tracking of discrete features (Rao et al., 1993). The RANSAC algorithm is an alternative mechanism for dealing with ambiguous association (Fischler and Bolles, 1981), in which hypotheses are generated bottom-up, from subsets of image features. Bucy's numerical integration of Bayes' rule for one-dimensional state (Bucy, 1969) is very general but feasible only in one or two dimensions, inapplicable to the state-spaces used in contour tracking whose dimensionality is typically 8–30. Additive Gaussian mixtures as used for representation of non-Gaussian densities in pattern recognition theory (Duda and Hart, 1973; Bishop, 1995) can be used in temporal filtering by dynamically re-weighting the mixture (Sorenson and Alspach, 1971). For approximate methods using additive Gaussian mixtures see also (Anderson and Moore, 1979).

The Fokker-Planck equation (Astrom, 1970) governs the evolution of the probability distribution for point particles following a random walk. The equation describes a diffusing probability density and its coefficients depend on the deterministic and stochastic components of the random walk, as described by the A and B coefficients in the case of an AR process.

One of the best known uses of iterative sampling in image processing is in the statistical restoration algorithms of Geman and Geman (Geman and Geman, 1984) in which a "Gibbs sampler" is used to sample fairly from the posterior distribution for the restored image. Their idea has been generalised and named "Markov Chain Monte-Carlo" (MCMC) (Gelfand and Smith, 1990). Random sampling as a means of image construction has also been applied to curves rather than image intensities (Ripley and Sutherland, 1990; Grenander et al., 1991), to sweep out a posterior distribution for an object outline, and also extended using simulated annealing (Storvik, 1994) to converge to a particular curve estimate. The term "factored sampling" is due to Grenander *et al.* (1991).

Importance sampling is a general technique (Ripley, 1987) for Monte-Carlo methods to bias generation of variates which would otherwise be generated from the stated prior. The purpose of the bias is to increase efficiency by concentrating on areas in which the observation density is likely to be non-negligible. The method includes a correction factor for the bias so that generated samples continue to be drawn fairly from the posterior density for the problem.

Fuller details of the CONDENSATION algorithm are given in (Isard and Blake, 1998a), including a proof of the (asymptotic) correctness of the algorithm. A similar extension of sampling methods to work over time has been reported also by (Gordon et al., 1993) who refer to the "bootstrap" algorithm and derive it via MCMC, rather than factored sampling. Another account (Kitagawa, 1996) elegantly extends the idea to a "smoothing" algorithm which is applicable to off-line applications. It is a two-pass method in which the estimate at each time-step is derived not only from preceding observations but also takes into account all following observations. This is analogous to the smoothing algorithm for Gaussians (Gelb, 1974) which is a two-pass extension of the Kalman filter.

Reasoning about clutter and false alarms of the sort given in section 12.3 is commonly used in target tracking (Bar-Shalom and Fortmann, 1988). Some discussion of the problem of extending the observation model from a single line to a curve of normals is given in (Isard and Blake, 1998a). An extension to account for object opacity is given in (MacCormick and Blake, 1998).

Some progress has been made recently in extending the scope of the motion models used in the CONDENSATION algorithm to include mixed continuous/discrete states (Isard and Blake, 1998b). This is related to Hidden Markov Modelling which is the dominant paradigm for speech recognition algorithms, and a comprehensive introduction is given in (Rabiner and Bing-Hwang, 1993).