A smoothing filter for CONDENSATION

Michael Isard and Andrew Blake

Department of Engineering Science,
University of Oxford, Oxford OX1 3PJ, UK,
misard,ab@robots.ox.ac.uk,
WWW home page: http://robots.ox.ac.uk/~ab

Abstract. CONDENSATION, recently introduced in the computer vision literature, is a particle filtering algorithm which represents a tracked object's state using an entire probability distribution. Clutter can cause the distribution to split temporarily into multiple peaks, each representing a different hypothesis about the object configuration. When measurements become unambiguous again, all but one peak, corresponding to the true object position, die out. While several peaks persist estimating the object position is problematic. "Smoothing" in this context is the statistical technique of conditioning the state distribution on both past and future measurements once tracking is complete. After smoothing, peaks corresponding to clutter are reduced, since their trajectories eventually die out. The result can be a much improved state-estimate during ambiguous time-steps. This paper implements two algorithms to smooth the output of a CONDENSATION filter. The techniques are derived from the work of Kitagawa, reinterpreted in the CONDENSATION framework, and considerably simplified.

1 Introduction

The CONDENSATION algorithm was recently introduced in the context of computer vision, originally to allow contour-tracking through heavy clutter [5], and more recently as the engine for exploring more complex non-linear dynamical models than have been traditionally used in vision [6, 4]. The algorithm is attractive because it is both simple and very general, and thus has potential application to a wide range of estimation problems beyond those contour-based tracking applications for which is has so far been used in computer vision. In fact, the CONDENSATION algorithm is functionally identical to algorithms developed in the target-tracking [2] and statistical literature [7]. In his formulation of the algorithm, Kitagawa [7] also presented two smoothing algorithms which allow the state at time t to be estimated in the light of all of the measurement data in a sequence, rather than just the data up until time t. Note that in this paper "smoothing" refers to the statistical technique of conditioning the state density on both past and future measurements. It has nothing to do with the standard computer vision definition involving convolution with a smoothing kernel, either spatially or temporally.

This paper implements Kitagawa's smoothing algorithms in the CONDENSATION framework, and in doing so incorporates a significant simplification of one of them which extends its use to a wider class of dynamical model. Smoothing highlights an aspect of CONDENSATION which has not so far been much studied. One of the distinguishing characteristics of the CONDENSATION algorithm is that it represents multiple hypotheses about object state in the form of a multi-modal state density. All of the known information about the object is contained in the state density, and this information must be processed in some way if a single estimated object position is required at each time-step. Existing implementations calculate simple moments of the state density, for example the mean, for display purposes. This approach breaks down when the density has several peaks, and one advantage of a smoothing filter is that it tends to eliminate hypotheses which become unlikely with hindsight. The result is that the smoothed density better approximates a uni-modal density, and simple mean-estimation produces a more accurate representation of the density. The next section briefly describes the CONDENSATION algorithm, and smoothing extensions are presented in following sections.

2 The CONDENSATION algorithm

The CONDENSATION algorithm [5, 6] was developed to address the problem of tracking contour outlines through heavy image clutter. The filter's output at a given time-step, rather than being a single estimate of position and covariance as in a Kalman filter, is an approximation of an entire probability distribution of likely object positions. This allows the filter to maintain multiple hypotheses and thus be robust to distracting clutter.

The object's position, shape and velocity are encoded in a state vector $\mathbf{X} \in \mathbb{R}^{N_X}$ (which may, for example, represent the outline of a curve using a low-dimensional parameterisation), and the observed image at time t is denoted $\mathbf{Z}_t$, with measurement history $\mathcal{Z}_t = (\mathbf{Z}_1, \ldots, \mathbf{Z}_t)$. The representation used for probability distributions is derived from factored sampling [3, 9], where it was applied to static images. Factored sampling is a Bayesian technique to approximate a distribution $p(\mathbf{X}|\mathbf{Z})$ which applies when $p(\mathbf{X}|\mathbf{Z})$ is too complicated to sample directly, but when the prior $p(\mathbf{X})$ can be sampled, and the measurement density $p(\mathbf{Z}|\mathbf{X})$ can be evaluated. The algorithm proceeds by generating a set of N samples $\{\mathbf{s}^{(n)}\}$ from the prior $p(\mathbf{X})$ and then assigning to each sample a weight $\pi^{(n)} = p(\mathbf{Z}|\mathbf{X} = \mathbf{s}^{(n)})$ corresponding to the measurement density. The $\pi^{(n)}$ are normalised to sum to 1 and then the weighted set $\{(\mathbf{s}^{(n)}, \pi^{(n)})\}$ is an approximation $\tilde{p}(\mathbf{X}|\mathbf{Z})$ to the desired posterior $p(\mathbf{X}|\mathbf{Z})$, where a sample is drawn from $\tilde{p}(\mathbf{X}|\mathbf{Z})$ by choosing one of the $\mathbf{s}^{(n)}$ with probability $\pi^{(n)}$. As $N \to \infty$ samples from $\tilde{p}(\mathbf{X}|\mathbf{Z})$ arbitrarily closely approximate fair samples from $p(\mathbf{X}|\mathbf{Z})$. Moments of the posterior can also be estimated as

$$\mathcal{E}[\phi(\mathbf{X})] \approx \sum_{n=1}^{N} \pi^{(n)} \phi\left(\mathbf{s}^{(n)}\right).$$

The CONDENSATION algorithm is a generalisation of factored sampling to temporal sequences, where the conditional state density $p(\mathbf{X}_t|\mathcal{Z}_t)$ at time t is approximated by a weighted, time-stamped sample set $\{(\mathbf{s}_t^{(n)}, \pi_t^{(n)})\}$. Each iteration of the algorithm is a self-contained application of factored sampling in which the prior $p(\mathbf{X}_t)$ is replaced by a prediction density $p(\mathbf{X}_t|\mathcal{Z}_{t-1})$. This density is approximated by taking the sample set $\{(\mathbf{s}_{t-1}^{(n)}, \pi_{t-1}^{(n)})\}$ from the previous time-step and applying a prediction from a dynamical model. The iterative process applied to the sample-sets is depicted in figure 1. At the top of the diagram, the output

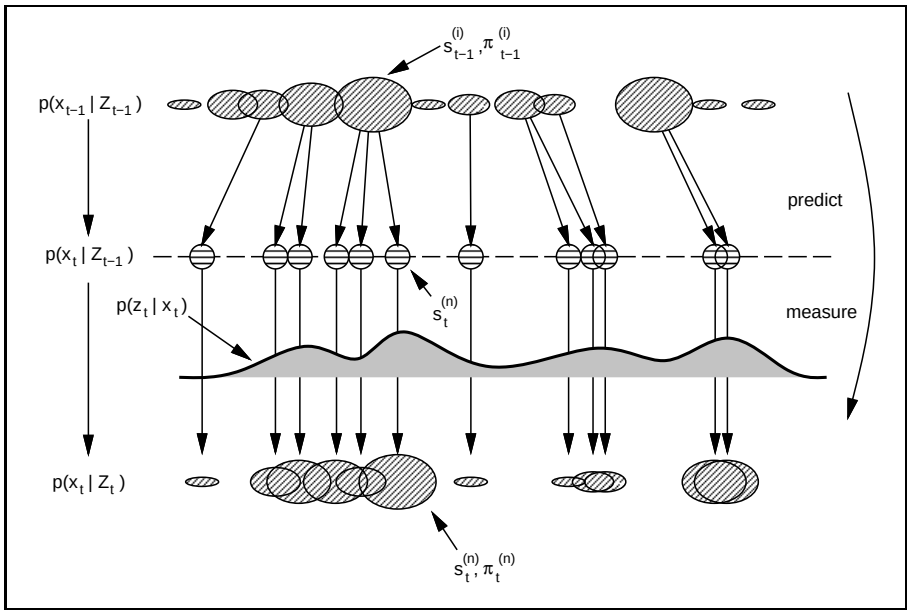

Fig. 1. One time-step in the CONDENSATION algorithm. *Blob centres represent sample values and sizes depict sample weights.*

from time-step $t-1$ is the weighted sample-set $\{(\mathbf{s}_{t-1}^{(n)}, \pi_{t-1}^{(n)}),\ n = 1, \ldots, N\}$. 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 procedure for a single time-step consists of N iterations to generate the N elements of the new sample set. The first operation of iteration n is to choose a "base sample" $\mathbf{s}_{t-1}^{(i)}$ from the sample-set at time $t-1$. This is done by sampling (with replacement), choosing a given element $\mathbf{s}_{t-1}^{(i)}$ with probability $\pi_{t-1}^{(i)}$. Some elements, especially those with high weights, may be chosen several times as n goes from 1 to N, while others with relatively low weights may not be chosen at all. The second step is to subject the chosen element to a

prediction corresponding to the dynamical model. This is a stochastic model, and the prediction of a new sample $\mathbf{s}_t^{(n)}$ from a base sample $\mathbf{s}_{t-1}^{(i)}$ corresponds to sampling from the process density $p(\mathbf{X}_t|\mathbf{X}_{t-1} = \mathbf{s}_{t-1}^{(i)})$, so the predictions from identical base samples will in general be different. Any dynamical model can be used efficiently within the algorithm provided that it is straightforward to sample from this process density. At this point the $\mathbf{s}_t^{(n)}$ are approximately a fair sample from the distribution $p(\mathbf{X}_t|\mathcal{Z}_{t-1})$. Finally, the observation step is applied, calculating a weight $\pi_t^{(n)}$ for $\mathbf{s}_t^{(n)}$ by evaluating the observation density $p(\mathbf{Z}_t|\mathbf{X}_t = \mathbf{s}_t^{(n)})$. After iterating over n the $\pi_t^{(n)}$ are normalised and the sample-set representation $\{(\mathbf{s}_t^{(n)}, \pi_t^{(n)})\}$ of the state-density for time t, an approximation to $p(\mathbf{X}_t|\mathcal{Z}_t)$, has been obtained. As with factored sampling, at any time-step it is possible to "report" on the current state, for example by evaluating some moment of the state density as

$$\mathcal{E}[\phi(\mathbf{X}_t)|\mathcal{Z}_t] \approx \sum_{n=1}^{N} \pi_t^{(n)} \phi\left(\mathbf{s}_t^{(n)}\right), \tag{1}$$

where typically $\phi(\mathbf{X}) = \mathbf{X}$ is used to estimate the mean of the distribution.

3 Smoothing the output of CONDENSATION

The conditional state density $p(\mathbf{X}_t|\mathcal{Z}_t)$ encodes all of the known information about the object state given the current measurement history $\mathcal{Z}_t \equiv (\mathbf{Z}_1, \ldots, \mathbf{Z}_t)$. Once tracking has completed it may be desirable to return, in batch-mode, to calculate $p(\mathbf{X}_t|\mathcal{Z}_T)$, the state density for each time-step given the *entire* measurement history. This is particularly valuable in the case of temporary distraction, when the state density splits for a few time-steps into several distinct trajectories. During real-time[1] tracking, it is impossible to reliably determine which of these competing hypotheses corresponds to the true object trajectory, however all but one of the trajectories will "die out" eventually when it becomes apparent that they correspond to clutter, distractions or mis-estimation.

Kitagawa [7] presents two algorithms to smooth a time-series of sample-set state estimates, which we reproduce here in the CONDENSATION framework. The first is very straightforward. Rather than storing the set $\{(\mathbf{s}_t^{(n)}, \pi_t^{(n)})\}$ at each time t, the sample position $\mathbf{s}_t^{(n)}$ is replaced by an entire trajectory $\mathcal{S}_t^{(n)} = (\mathbf{s}_t^{(n,1)}, \ldots, \mathbf{s}_t^{(n,t)})$. The history $(\mathbf{s}_t^{(n,1)}, \ldots, \mathbf{s}_t^{(n,t-1)})$ is taken to be the trajectory of the base sample which is chosen in the first step of the CONDENSATION algorithm, and the moments of the smoothed density $p(\mathbf{X}_\tau|\mathcal{Z}_t)$ can be

[1] Real time is used here to distinguish the standard CONDENSATION tracking algorithm from any batch-mode post-processing. It does not imply the standard computer vision meaning, that tracking is effected in the time between acquisition of consecutive images.

estimated for $1 \leq \tau \leq t$ by computing the expectation

$$\mathcal{E}[\phi(\mathbf{X}_\tau)|\mathcal{Z}_t] \approx \sum_{n=1}^{N} \pi_t^{(n)} \phi\left(\mathbf{s}_t^{(n,\tau)}\right).$$

The sequence-based smoothing algorithm is shown in figure 2. Note that it is reminiscent of the Viterbi dynamic programming algorithm used, for example, to estimate the most likely path through a Hidden Markov Model (HMM) [8]. This algorithm has the disadvantage that in practice, the variance of the samples

Iterate

From the "old" sample-sequence set $\{(\mathcal{S}_{t-1}^{(n)}, \pi_{t-1}^{(n)}),\ n = 1, \ldots, N\}$ at time-step $t-1$, construct a "new" sample-sequence set $\{(\mathcal{S}_t^{(n)}, \pi_t^{(n)}),\ n = 1, \ldots, N\}$ for time t, where $\mathcal{S}_t^{(n)} = (\mathbf{s}_t^{(n,1)}, \ldots, \mathbf{s}_t^{(n,t)})$.

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

1. **Select** a base sequence $\mathcal{S}_{t-1}^{(i)} = (\mathbf{s}_{t-1}^{(i,1)}, \ldots, \mathbf{s}_{t-1}^{(i,t-1)})$ by sampling with probability $\pi_{t-1}^{(i)}$. This can be done efficiently, for example using cumulative probabilities.
2. **Predict** by sampling from $p(\mathbf{X}_t|\mathbf{X}_{t-1} = \mathbf{s}_{t-1}^{(i,t-1)})$ to choose $\mathbf{s}_t^{(n,t)}$.
3. **Measure** and weight the new position in terms of the image data $\mathbf{Z}_t$, setting $\pi_t^{(n)} = p(\mathbf{Z}_t|\mathbf{X}_t = \mathbf{s}_t^{(n,t)})$, then set $\mathcal{S}_t^{(n)} = \mathcal{S}_{t-1}^{(i)} \cup \mathbf{s}_t^{(n,t)}$.

Finally normalise so that $\sum_n \pi_t^{(n)} = 1$ to find the new sample-sequence set $(\mathcal{S}_t^{(n)}, \pi_t^{(n)})$. Moments ϕ of the smoothed density $p(\mathbf{X}_\tau|\mathcal{Z}_t)$ for $1 \leq \tau \leq t$ can be found from

$$\mathcal{E}[\phi(\mathbf{X}_\tau)|\mathcal{Z}_t] \approx \sum_{n=1}^{N} \pi_t^{(n)} \phi\left(\mathbf{s}_t^{(n,\tau)}\right)$$

Fig. 2. The sequence-based smoothing algorithm for CONDENSATION. *The algorithm is identical to standard* CONDENSATION *filtering, except that entire trajectories* $\mathcal{S}_t^{(n)}$ *are stored instead of sample positions* $\mathbf{s}_t^{(n)}$.

$\{\mathbf{s}_t^{(n,\tau)}\}$ for $\tau \ll t$ is very small. In fact, for large $t-\tau$ it is typical to find that all of the $\{\mathbf{s}_t^{(n,\tau)},\ 1 \ldots N\}$ are identical, meaning that all of the sample-sequences share a common ancestor trajectory. (In later results this is typically true for $t - \tau > 10$.) This may be acceptable if the only required output is a single estimated position for each time-step, but in some circumstances it is preferable to maintain more detailed information as long as possible, and so a more complex algorithm follows. Note that the collapse of the trajectories $\mathcal{S}_t^{(n)}$ into common

histories permits pruning, thus allowing a significant economy of storage, which is otherwise $O(Nt)$.

The second smoothing algorithm presented in [7] is a forward–backward algorithm, analogous to the smoothing algorithm for Gaussians [1] which is a two-pass extension of the Kalman filter, and also related to the Baum-Welch forward–backward algorithm for HMMs [8]. The forward pass consists of a standard application of the CONDENSATION tracker, during which all the sets $\{(\mathbf{s}_t^{(n)}, \pi_t^{(n)})\}$ for $t = 1 \dots T$ are stored. Now smoothing is done purely by reweighting the $\pi_t^{(n)}$ — all of the $\mathbf{s}_t^{(n)}$ remain fixed. The algorithm presented in [7] contains a backward filtering step which requires access to the measurements $\mathbf{Z}_t$ during the second pass, and also means that the density $p(\mathbf{X}_{t-1}|\mathbf{X}_t)$ must be available for sampling, a condition which is not true for the standard CONDENSATION algorithm. We believe this backward filtering step is unnecessary and so do not include it, however the mathematical treatment and the basic structure of our algorithm are both derived from [7]. Note that our algorithm, like that in [7], does require the evaluation of $p(\mathbf{X}_t|\mathbf{X}_{t-1})$ which imposes some restriction on the form of dynamical model used.

Defining $\mathcal{Z}_t^T = (\mathcal{Z}_t, \dots, \mathcal{Z}_T)$ we have $\mathcal{Z}_T = \mathcal{Z}_{t-1} \cup \mathcal{Z}_t^T$. Therefore,

$$\begin{aligned} p(\mathbf{X}_t|\mathcal{Z}_T) &= p(\mathbf{X}_t|\mathcal{Z}_{t-1}, \mathcal{Z}_t^T) \\ &\propto p(\mathbf{X}_t, \mathcal{Z}_t^T|\mathcal{Z}_{t-1}) \\ &= p(\mathcal{Z}_t^T|\mathbf{X}_t) p(\mathbf{X}_t|\mathcal{Z}_{t-1}) \quad \text{by the independence of the } \mathbf{Z}_t. \end{aligned}$$

It is this rearrangement which allows the sample positions $\mathbf{s}_t^{(n)}$ to remain fixed after the smoothing step. Recall that the set $\{\mathbf{s}_t^{(n)}\}$ is approximately a fair sample from $p(\mathbf{X}_t|\mathcal{Z}_{t-1})$, so by replacing the original $\pi_t^{(n)}$ by smoothing weights

$$\psi_t^{(n)} = p(\mathcal{Z}_t^T|\mathbf{X}_t = \mathbf{s}_t^{(n)}),$$

the set $\{(\mathbf{s}_t^{(n)}, \psi_t^{(n)})\}$, when normalised, will approximate $p(\mathbf{X}_t|\mathcal{Z}_T)$ as required. It is therefore the weights $\psi_t^{(n)}$ which the backward smoothing pass will calculate.

A recursive algorithm to calculate the densities $p(\mathcal{Z}_t^T|\mathbf{X}_t)$ can be specified mathematically as follows:

$$\begin{aligned} p(\mathcal{Z}_T^T|\mathbf{X}_T) &= p(\mathbf{Z}_T|\mathbf{X}_T) \\ p(\mathcal{Z}_{t+1}^T|\mathbf{X}_t) &= \int p(\mathcal{Z}_{t+1}^T|\mathbf{X}_{t+1}) p(\mathbf{X}_{t+1}|\mathbf{X}_t)\, d\mathbf{X}_{t+1} \\ p(\mathcal{Z}_t^T|\mathbf{X}_t) &= p(\mathbf{Z}_t|\mathbf{X}_t) p(\mathcal{Z}_{t+1}^T|\mathbf{X}_t) \end{aligned}$$

A concrete implementation requires the derivation of an approximation $\delta_t^{(n)}$ to $p(\mathcal{Z}_{t+1}^T|\mathbf{X}_t = \mathbf{s}_t^{(n)})$. The integral is approximated as a sum:

$$p(\mathcal{Z}_{t+1}^T|\mathbf{X}_t = \mathbf{s}_t^{(n)}) \approx \delta_t^{(n)} = \sum_{m=1}^{N} p(\mathcal{Z}_{t+1}^T|\mathbf{X}_{t+1} = \mathbf{s}_{t+1}^{(m)}) \frac{p(\mathbf{X}_{t+1} = \mathbf{s}_{t+1}^{(m)}|\mathbf{X}_t = \mathbf{s}_t^{(n)})}{p(\mathbf{X}_{t+1} = \mathbf{s}_{t+1}^{(m)}|\mathcal{Z}_t)}$$

where the correction

$$p(\mathbf{X}_{t+1} = \mathbf{s}_{t+1}^{(m)} | \mathcal{Z}_t) = \gamma_t^{(m)} = \sum_{k=1}^{N} \pi_t^{(k)} p(\mathbf{X}_{t+1} = \mathbf{s}_{t+1}^{(m)} | \mathbf{X}_t = \mathbf{s}_t^{(k)}).$$

It is introduced because the $\mathbf{s}_{t+1}^{(m)}$ are not distributed uniformly but are a sam-

At this stage a forward pass of standard CONDENSATION has been performed, storing a weighted sample-set $\{(\mathbf{s}_t^{(n)}, \pi_t^{(n)})\}$ for each $t = 1 \dots T$.

1. **Initialise** smoothing weights $\psi_T^{(n)}$:

$$\psi_T^{(n)} = \pi_T^{(n)} \quad \text{for } n = 1 \dots N.$$

2. **Iterate** backwards over the sequence for $t = T - 1 \dots 1$:
   (a) **Calculate** prediction probabilities:

$$\alpha_t^{(m,n)} = p(\mathbf{X}_{t+1} = \mathbf{s}_{t+1}^{(m)} | \mathbf{X}_t = \mathbf{s}_t^{(n)}) \quad \text{for } m, n = 1 \dots N.$$

   (b) **Calculate** correction factors:

$$\gamma_t^{(m)} = \sum_{k=1}^{N} \pi_t^{(k)} \alpha_t^{(m,k)} \quad \text{for } m = 1 \dots N.$$

   (c) **Approximate** backward variables:

$$\delta_t^{(n)} = \sum_{m=1}^{N} \psi_{t+1}^{(m)} \frac{\alpha_t^{(m,n)}}{\gamma_t^{(m)}} \quad \text{for } n = 1 \dots N.$$

   (d) **Evaluate** smoothing weights

$$\psi_t^{(n)} = \pi_t^{(n)} \delta_t^{(n)} \quad \text{for } n = 1 \dots N$$

   then normalise multiplicatively so $\sum \psi_t^{(n)} = 1$, and store with sample positions as

$$\{(\mathbf{s}_t^{(n)}, \psi_t^{(n)}),\ n = 1 \dots N\}$$

Fig. 3. The backward stage of the two-pass smoothing algorithm for CONDENSATION.

ple from $p(\mathbf{X}_{t+1} | \mathcal{Z}_t)$, and without the correction this could bias the sum. This method of correcting the estimate of an integral over sample-sets is borrowed from the technique of importance sampling [9]. The backward pass of the two-pass smoothing algorithm is shown in figure 3. Note that the complexity of the

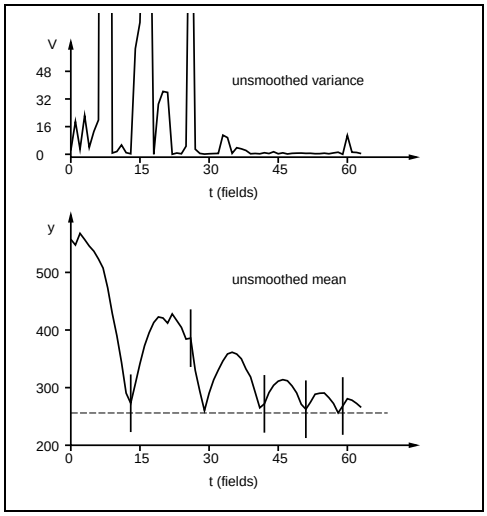

Fig. 4. The unsmoothed output of a mixed-state CONDENSATION algorithm contains estimation errors. *The mean of the y coordinate of the distribution is shown in pixels, along with the mean-square variance around the curve in pixels*^{2}*. Vertical bars correspond to time-steps which are estimated to contain bounce events. The variance is high when several hypotheses have high probabilities (see figure 5).*

algorithm is $O(TN^2)$ and that $O(TN)$ storage is required for the sample-sets, and $O(N^2)$ for the $\alpha_t^{(m,n)}$. This latter storage requirement can be avoided by eliminating the $\alpha_t^{(m,n)}$ from the algorithm and instead calculating each of the $p(\mathbf{X}_{t+1} = \mathbf{s}_{t+1}^{(m)} | \mathbf{X}_t = \mathbf{s}_t^{(n)})$ twice. Since this calculation is typically the most computationally expensive step of the algorithm, this tradeoff must be carefully considered.

4 Applying the smoothing algorithms

First, the sequence-based smoothing algorithm was applied to a test sequence from [6] which shows a ball bouncing against a backdrop of heavy clutter. The ball moves under the action of a two-mode motion model, where the first mode is constant acceleration due to gravity and the second mode corresponds to an instantaneous bounce event during which the ball's vertical velocity is reversed. The state vector $\mathbf{X}_t$ now includes a discrete variable labelling which of the two transition modes the model has just executed. The unsmoothed output of a mixed-state CONDENSATION tracker is depicted in figure 4. At each time-step, an MAP estimate is computed to determine which of the two modes the tracker has executed, and the mean and variance of the y translation coordinate within that mode are shown, along with an indication of which time-steps were estimated to contain bounce events. The unsmoothed output is rather jittery due

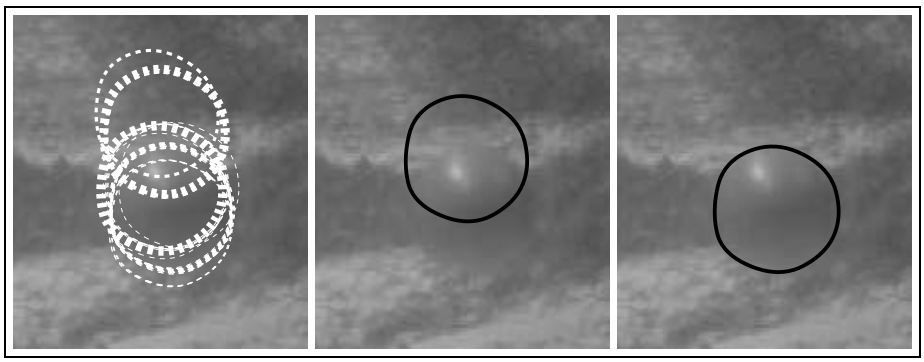

Fig. 5. Smoothing eliminates false hypotheses. *Before smoothing, multiple hypotheses can increase the variance of the distribution (top left) and shift the mean away from the object position (top right). After running the sequence-based smoothing algorithm the estimated variance has dropped to zero (see text) but the mean is now correctly positioned (bottom). Detail from field 26 of the sequence is shown (see figures 4 and 6). The solid black line is the distribution mean, and the dotted white lines are high-scoring samples, where the width of the sample outline is proportional to its sample weight. A ball is being tracked against heavy clutter, and it is difficult to distinguish in a single still image. The ball is located under the contour in the right-hand image.*

to the clutter, and the bounce events are not always accurately found. Figure 5 demonstrates the mis-estimation problem; the distribution has split into several peaks, and although one peak is present at the true ball position, the other peaks pull the distribution mean away from the desired value. After running the sequence-based smoothing algorithm (figure 6) most of the jitter has been eliminated and the trajectory shows smooth parabolas between bounces. One field has still been incorrectly estimated to contain a bounce. As discussed in the previous section, the variance is estimated to be zero except over the last few time-steps, since all the samples in the final distribution share the same history until $t = 60$ fields. Of course, this must be an under-estimate. Figure 5 shows detail from field 26 of the sequence before and after smoothing.

The two-pass algorithm also successfully smooths the raw tracked output (figure 7), and now correctly determines the bounce events. Variance information is also preserved by the two-pass filter, and a small spread of samples in the distribution can be seen in figure 8. Note that neither smoothing algorithm incorporates any separate machinery to estimate the mixed-state transitions. These transition labels, forming part of the state-vector $\mathbf{X}_t$, are automatically estimated along with the continuous state variables. Of course, the values of the transition labels of $\mathbf{s}_t^{(n)}$ and $\mathbf{s}_{t-1}^{(m)}$ play a large part in determining the density $p(\mathbf{X}_{t+1} = \mathbf{s}_{t+1}^{(n)} | \mathbf{X}_t = \mathbf{s}_t^{(m)})$ for the two-pass algorithm.

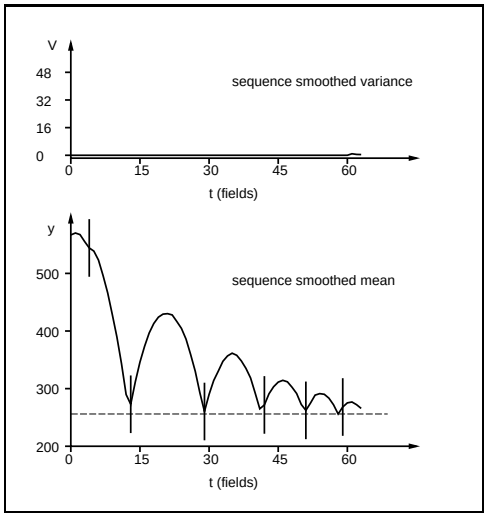

Fig. 6. The sequence-based algorithm smooths away jitter. *The mean of the y coordinate of the distribution is shown in pixels, along with the mean-square variance around the curve in pixels*^{2}*. Vertical bars correspond to time-steps which are estimated to contain bounce events, and a bounce is incorrectly estimated at field 4. The sequence-based smoothing algorithm collapses the variance to zero for all but the last few time-steps (see text).*

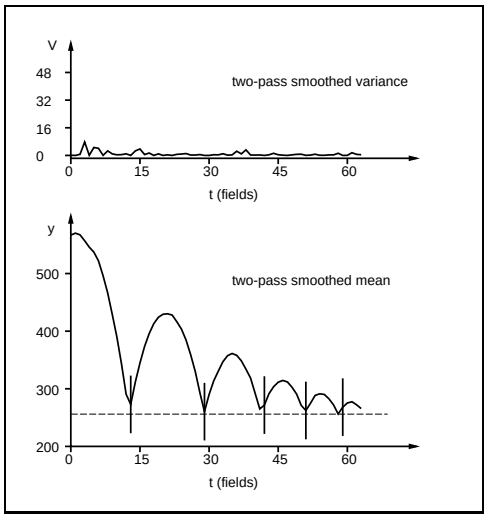

Fig. 7. The two-pass algorithm preserves sample variance while smoothing. *The mean of the y coordinate of the distribution is shown in pixels, along with the mean-square variance around the curve in pixels*^{2}*. Vertical bars correspond to modes which are estimated to contain bounce events. The bounce events are correctly identified.*

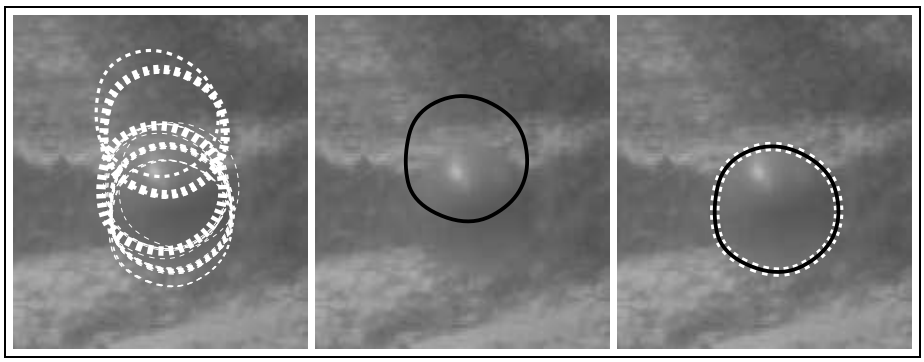

Fig. 8. Smoothing eliminates false hypotheses. *The two-pass smoothing algorithm collapses the distribution down to a single peak (bottom), since the other peaks in the raw data (top left) which cause a mis-estimation of the object position (top right), die out in subsequent time-steps due to lack of support. Within the peak, however, variance information is preserved, and so a small spread of samples is present around the mean (figure 7 indicates that the estimated mean-square variance around the curve is only a few pixels). Detail from field 26 of the sequence is shown. The solid black line is the distribution mean, and the dotted white lines are high-scoring samples, where the width of the sample outline is proportional to its sample weight. A ball is being tracked against heavy clutter, and it is difficult to distinguish in a single still image. The ball is located under the contour in the right-hand image.*

Finally, the algorithms were applied to another test sequence showing a hand moving over a cluttered desk. The hand translates and deforms in a 12-dimensional linear shape-space. After approximately 30 fields, the distribution splits into two peaks (figure 9), one of which is caused by clutter. The clutter peak dominates for 10 fields, causing a serious error in the estimated state, although the true position is maintained as a smaller peak in the distribution throughout, and the tracker recovers eventually. Figure 10 shows a graphs of the y coordinate of the estimated mean of the distribution along with the variance of the sample-set. The hand moves up smoothly from field 20 to field 40, but the unsmoothed estimate is distracted between fields 30–40, before rapidly regaining the correct position at field 42. Note the very high variances, especially just before the tracker recovers.

Figure 11 shows the result of applying the two-pass smoothing algorithm to the hand sequence (the sequence-based algorithm provides similar state estimates and lower variance as before). When the entire sequence is taken into account, it is apparent that the lower peak in figure 9 corresponds to clutter, and so only the trajectory corresponding to the actual hand position survives. Figure 12 graphs the estimated y coordinate and the mean-square curve variances for the output of the two-pass smoother. As in the case of the ball, the jitter on the state estimates is reduced, and the hand position is significantly more accurately determined compared with the raw CONDENSATION algorithm. The

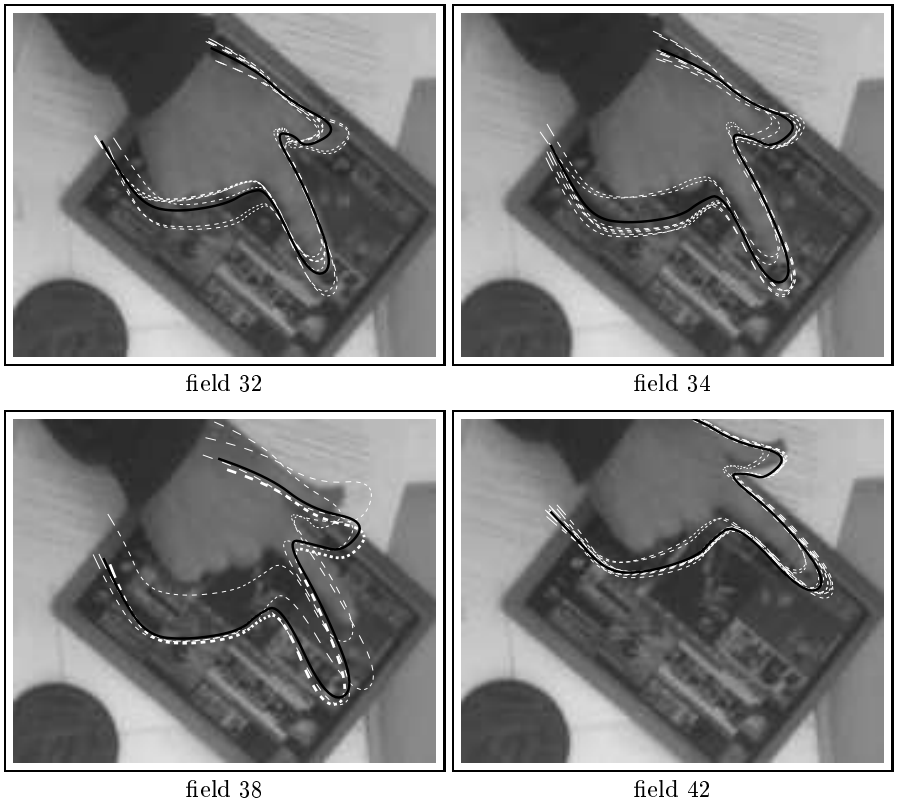

Fig. 9. Clutter causes temporary mis-estimation from unsmoothed data. *The unsmoothed state distribution has begun to diverge in field 32, and by field 34 the clutter peak dominates. The multi-modality persists until field 38, after which the clutter peak rapidly dies away, leaving a single peak around the object again by field 42. The solid line is the distribution mean, and the dotted lines are high-scoring samples, where the width of the sample outline is proportional to its sample weight.*

variance of the sample-sets is also much reduced, although clearly towards the end of the sequence the variance must increase to match that of the raw data.

5 Conclusions and future work

Both of the smoothing algorithms presented here significantly aid the interpretation of the output of a CONDENSATION tracker. One of the major benefits of the CONDENSATION algorithm is that it allows the state density to split into several peaks to transiently represent multiple hypotheses about object configuration. This facility enables the tracker to follow the object while measurements are ambiguous, keeping track of several possible trajectories until the true ob-

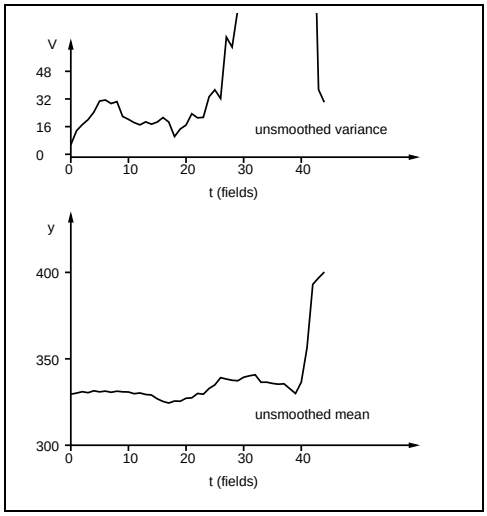

Fig. 10. The unsmoothed hand data leads to estimation errors. *The mean of the y coordinate of the distribution is shown in pixels, along with the mean-square variance around the curve in pixels*^{2}*. Figure 9 shows the distribution splitting into two peaks between fields 32–34, and this is apparent from the variances. The clutter peak is stronger, and causes the position to be mis-estimated by shifting the distribution mean. Although the hand moves up steadily during fields 30–40, the estimated position moves down before suddenly recovering at field 42.*

ject position can once more be confidently established. During the period that the distribution contains multiple peaks, however, existing implementations of CONDENSATION may report grossly misleading state estimates even though they ultimately recover. This is because the state estimates are based on the mean of the distribution, and thus implicitly assume a single peak. The application of a smoothing algorithm concentrates the distribution into those areas which are most likely given the entire tracking sequence, and the result is that peaks caused by temporary clutter distractions tend to be greatly reduced in size. The distribution is then more approximately uni-modal, and its mean is a good estimator for the object configuration. The forward–backward algorithm may also prove very useful for learning mixed-state motion models. The algorithm can be used as the basis of an E–M procedure analagous to the Baum-Welch algorithm for learning Hidden Markov Model coefficients, and this is the subject of current research.

The two algorithms were tested on sequences where the state distribution periodically diverges to form several hypotheses and all but one of these competing hypotheses ultimately dies out. Both algorithms successfully smoothed the test sequences, with slightly improved accuracy from the two-pass algorithm, and this suggests that for tasks of this kind the sequence-based algorithm should be used, given its greater conceptual and computational simplicity. It is anticipated

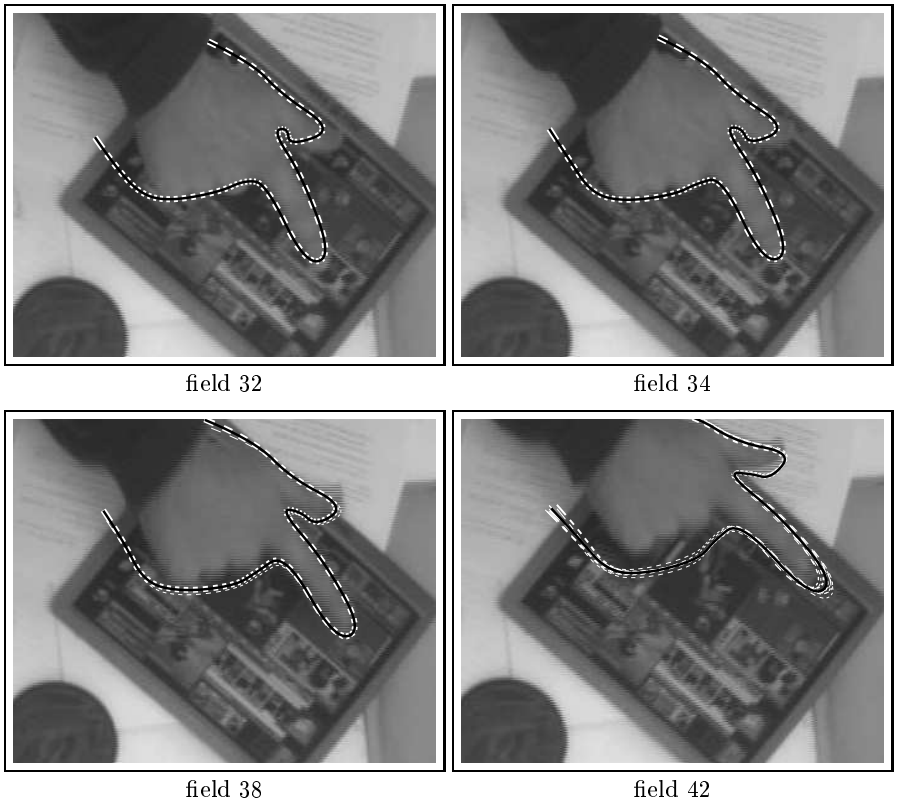

Fig. 11. The smoothing algorithm correctly follows the hand. *Compare with figure 9; after smoothing using the two-pass algorithm, all of the visible distribution is concentrated on the correct peak. The solid line is the distribution mean after smoothing, and the dotted lines are high-scoring samples, where the width of the sample outline is proportional to its weight.*

that the two-pass algorithm will come into its own as more complex distributions come to be used while tracking, and more complex state estimates are required than a single configuration at each time-step. A situation could arise where the state density repeatedly split into competing hypotheses and then merged again, for example if two similar objects were moving in front of one another. The sequence-based algorithm would be very unlikely to preserve the structure of the trajectories; instead it would tend to choose the most likely single path. The two-pass algorithm, on the other hand, by computing a richer representation of the past history, is more likely to keep all the likely hypotheses and only reject genuine clutter. It may also be desirable to estimate sample-set variances to detect periods of uncertainty, for example due to partial occlusion of an object, and the two-pass algorithm is much better suited to this task.

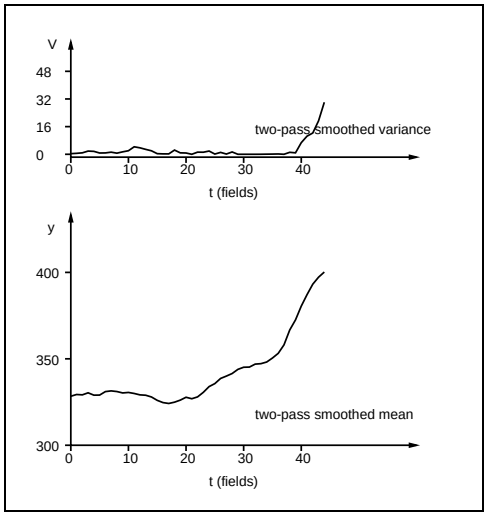

Fig. 12. The two-pass smoothing algorithm corrects estimation errors. *Figure 11 shows that the two-pass algorithm eliminates the clutter peak which distracted the standard tracker. Now the estimated state corresponds to the true hand position as it moves steadily up the image from field 20 (compare with figure 10). The variance of the sample-set is also greatly reduced, although clearly at the end of the sequence the variance increases to match that of the raw output. The mean of the y coordinate of the distribution is shown in pixels, along with the mean-square variance around the curve in pixels*^{2}*.*

References

1. A. Gelb, editor. *Applied Optimal Estimation.* MIT Press, Cambridge, MA, 1974.
2. N. Gordon, D. Salmond, and A.F.M. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proc. F*, 140(2):107–113, 1993.
3. U. Grenander, Y. Chow, and D.M. Keenan. *HANDS. A Pattern Theoretical Study of Biological Shapes.* Springer-Verlag. New York, 1991.
4. T. Heap and D. Hogg. Wormholes in shape space: Tracking through discontinuous changes in shape. In *Proc. 6th Int. Conf. on Computer Vision*, 1998.
5. M.A. Isard and A. Blake. Visual tracking by stochastic propagation of conditional density. In *Proc. 4th European Conf. Computer Vision*, 343–356, Cambridge, England, Apr 1996.
6. M.A. Isard and A. Blake. A mixed-state Condensation tracker with automatic model switching. In *Proc. 6th Int. Conf. on Computer Vision*, 107–112, 1998.
7. G. Kitagawa. Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *Journal of Computational and Graphical Statistics*, 5(1):1–25, 1996.
8. L. Rabiner and J. Bing-Hwang. *Fundamentals of speech recognition.* Prentice-Hall, 1993.
9. B.D. Ripley. *Stochastic simulation.* New York: Wiley, 1987.