ICONDENSATION: Unifying Low-Level and High-Level Tracking in a Stochastic Framework

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. Tracking research has diverged into two camps; low-level approaches which are typically fast and robust but provide little fine-scale information, and high-level approaches which track complex deformations in high-dimensional spaces but must trade off speed against robustness. Real-time high-level systems perform poorly in clutter and initialisation for most high-level systems is either performed manually or by a separate module. This paper presents a new technique to combine low- and high-level information in a consistent probabilistic framework, using the statistical technique of importance sampling combined with the CONDENSATION algorithm. The general framework, which we term ICONDENSATION, is described, and a hand tracker is demonstrated which combines colour blob-tracking with a contour model. The resulting tracker is robust to rapid motion, heavy clutter and hand-coloured distractors, and re-initialises automatically. The system runs comfortably in real time on an entry-level desktop workstation.

1 Introduction

Current research into tracking has somewhat diverged into two camps — informally these can be distinguished as low-level vs. high-level. Low-level approaches include "blob trackers" [24, 17] and systems which track sets of point features [22, 6]. Blob trackers perform low-level processing, for example colour segmentation, usually on low-resolution images, and are fast and robust but convey little information other than object centroid. Rigid object deformations can be tracked by matching point correspondences frame-to-frame [21], but this relies on a rich set of point features on the object of interest, and segmenting the sets of points into coherent objects is challenging. An alternative is to use higher-level information, whether by modelling objects with specific grey-level templates [2] which may be allowed to deform [12], or with more abstract templates such as curved outlines [4, 5]. By including high-level motion models [4, 1] these trackers can follow complex deformations in high-dimensional spaces, but there tends to be a trade-off between speed and robustness. Kalman-filter based contour trackers which run in real time are very susceptible to distraction by clutter, and correlation-based systems are vulnerable to changes in object appearance and lighting, and

rapidly slow down as the space of deformations increases in complexity. Contour trackers have been constructed which are highly robust to clutter [14, 18] but only by sacrificing real-time performance. The high-level approaches also tend to economise on processing time by searching only those regions of the image where the object is predicted to be. This diminishes robustness, and also precludes natural extensions of the trackers to perform initialisation when the object could be anywhere in the image. The difficulty of initialisation is compounded when the dimension of the tracking space increases, since it is rapidly impractical to perform an exhaustive search for the object.

This paper presents a framework, ICONDENSATION, to bridge the gap between low-level and high-level tracking approaches. An implementation is demonstrated which uses colour segmentation to find skin-coloured blobs in a subsampled image, and feeds this information to a contour tracker specialised for hands. The techniques used, however, apply to the general sensor fusion problem of augmenting a tracker operating with one measurement modality to use information from an auxiliary measurement source. Tracking is achieved using a CONDENSATION filter [14], extended to incorporate the statistical technique of "Importance Sampling." [20] Importance sampling offers a mathematically principled way of directing search, combining prediction information based on the previous object position and motion with any additional knowledge which may be available from auxiliary sensors. This *combination* confers robustness to temporary failures in one of the measurement processes, and allows the tracker to take advantage of the distinct qualities of different information sources. In the hand-tracking system presented here, for example, colour segmentation allows rapid initialisation and robust tracking of gross motions, while the contour tracker gives fine-scale position and shape information as well as maintaining lock on the object when colour blobs merge or momentarily disappear. Previous demonstrations of the CONDENSATION algorithm have been slower than real time, sometimes by a significant margin [14, 15]. The hand-tracker presented here operates comfortably in real time (30 or 60 Hz) on an entry-level desktop workstation (SGI O2 R5000 180SC). The speed improvement is due partly to a reduction in the required number of samples as a result of using importance sampling, and partly to a careful implementation which is discussed in section 4.

2 Shape representation and sample sets

Suppose that an 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 images observed at time t are denoted $\mathbf{Z}_t$, with measurement history $\mathcal{Z}_t = (\mathbf{Z}_1, \ldots, \mathbf{Z}_t)$. The Bayesian technique of factored sampling [20, 11] is a random-sampling method 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. Factored sampling 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 $\{(\mathbf{s}^{(n)}, \pi^{(n)})\}$ represents 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

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

Factored sampling has been generalised to deal with temporal sequences [14, 9, 16] under a variety of names. The CONDENSATION algorithm [14] was introduced in the context of computer vision, and it incorporates learned motion models to track deforming objects, represented by image-contours, through cluttered image sequences. This paper extends that work, however the importance sampling framework applies equally well to the algorithms of [9, 16]. In the case of temporal sequences, each time-step consists of an application of factored sampling, where the prior $p(\mathbf{X})$ is replaced by a prediction density $p(\mathbf{X}_t|\mathcal{Z}_{t-1})$ and the sample-set obtained is time-stamped and denoted $\{(\mathbf{s}_t^{(n)}, \pi_t^{(n)})\}$. The prediction density $p(\mathbf{X}_t|\mathcal{Z}_{t-1})$ is obtained by applying a dynamical model $p(\mathbf{X}_t|\mathbf{X}_{t-1})$ to the output $\{(\mathbf{s}_{t-1}^{(n)}, \pi_{t-1}^{(n)})\}$ of the previous time-step.

3 Importance sampling

In the standard formulation of the CONDENSATION algorithm, positions of samples $\mathbf{s}_t^{(n)}$ are fixed in the prediction stage using only the previous approximation to the state density $\{(\mathbf{s}_{t-1}^{(n)}, \pi_{t-1}^{(n)})\}$ and the motion model $p(\mathbf{X}_t|\mathbf{X}_{t-1})$. The portions of state-space (and thus the image $\mathbf{Z}_t$) which are to be examined in the measurement stage are therefore determined before any measurements are made. This is appropriate when the sample-set approximation to the state density is accurate. In principle, as the state density evolves over time, the random nature of the motion model induces some non-zero probability everywhere in state-space that the object is present at that point. With a sufficiently good sample-set approximation this would tend to cause all areas of state-space to lie near some samples, so even motions which were extremely unlikely given the model would be detected, and could therefore be tracked. In practice, however, the finite nature of the sample-set approximation means that all of the samples will be concentrated near the most likely object positions. There may be several such clusters corresponding to multiple hypotheses, but in general each cluster will be fairly localised, which in fact is precisely the behaviour which permits an efficient discrete representation of high-dimensional state spaces. The result is that large areas of state-space contain no samples at all. In order to robustly track sudden movements the process noise of the motion model must be artificially high, thus increasing the extent of each predicted cluster in state-space. To populate these larger clusters with enough samples to permit effective tracking, the sample-set

size must be increased, and the algorithm therefore runs more slowly. Various techniques have been proposed to improve the efficiency of the representation in random sampling filters [9,8] but to our knowledge none have been advanced which draw on information available from alternative sensors.

Importance sampling [20] is a technique developed to improve the efficiency of factored sampling. It applies when auxiliary knowledge is available in the form of an importance function $g(\mathbf{X})$ describing which areas of state-space contain most information about the posterior. The idea is then to concentrate samples in those areas of state-space by generating sample positions $\mathbf{s}^{(n)}$ from $g(\mathbf{X})$ rather than sampling from the prior $p(\mathbf{X})$. The desired effect is to avoid as far as possible generating any samples which have low weights, since they are "wasted" in the factored sampling representation as they provide a negligible contribution to the posterior. A correction term f/g must be added to the sample weights giving

$$\pi^{(n)} = \frac{f(\mathbf{s}^{(n)})}{g(\mathbf{s}^{(n)})} p(\mathbf{Z}|\mathbf{X} = \mathbf{s}^{(n)}) \quad \text{where} \quad f(\mathbf{s}^{(n)}) \equiv p(\mathbf{X} = \mathbf{s}^{(n)})$$

to compensate for the uneven distribution of sample positions. This correction term ensures that, for large N, importance sampling has *no effect* on the consistency of the approximation which $\tilde{p}(\mathbf{X}|\mathbf{Z})$ makes to the posterior. Any importance function could be chosen (subject to some weak constraints) and if N is sufficiently large then $\tilde{p}(\mathbf{X}|\mathbf{Z})$ will be a good approximation to $p(\mathbf{X}|\mathbf{Z})$. The purpose of importance sampling is to reduce the variance of the estimates for a given N and so improve the accuracy of $\tilde{p}(\mathbf{X}|\mathbf{Z})$ when N is small. Since samples are drawn from $g(\mathbf{X})$ it plays the part of a probability density, but note that it does not necessarily correspond to the distribution of any particular random variable.

A typical Bayesian approach to sensor fusion would be to combine measurements from the various sensors in the representation of $\mathbf{Z}_t$, weighted according to their inverse variances. This is only possible, however, when the statistical dependencies between the measurements are understood, and in practice it is often assumed that sensors produce independent measurements. This paper is concerned with combining measurements made with different modalities but from the same underlying image, so it is expected that such an independence assumption would be invalid. Instead of the traditional sensor fusion approach, therefore, importance sampling allows measurements to be combined in a Bayesian framework even when no knowledge at all is available about their dependence. The tradeoff is that the symmetry between sensors is broken, since the measurements used to define the importance function are not included in the overall model, and this may result in some genuine independent information being discarded.

Importance sampling can be applied in the context of CONDENSATION sampling, and we denote this extension ICONDENSATION. Now the importance function at time t is denoted $g_t(\mathbf{X}_t)$. In CONDENSATION, sample positions are drawn from the density

$$f_t(\mathbf{s}_t^{(n)}) \equiv \tilde{p}(\mathbf{X}_t = \mathbf{s}_t^{(n)} | \mathcal{Z}_{t-1})$$
$$= \sum_{j=1}^{N} \pi_{t-1}^{(j)} p(\mathbf{X}_t = \mathbf{s}_t^{(n)} | \mathbf{X}_{t-1} = \mathbf{s}_{t-1}^{(j)}). \quad (2)$$

Note that while the set $\{(\mathbf{s}_t^{(n)}, \pi_t^{(n)})\}$ provides a discrete point-representation of a distribution, the prediction density $\tilde{p}(\mathbf{X}_t|\mathcal{Z}_{t-1})$ is a mixture of continuous density kernels shaped by $p(\mathbf{X}_t|\mathbf{X}_{t-1})$, representing the dynamical model. Instead of sampling from $\tilde{p}(\mathbf{X}_t|\mathcal{Z}_{t-1})$, samples $\mathbf{s}_t^{(n)}$ can instead be drawn from some $g_t(\mathbf{X}_t)$ and then the weights need to be redefined as

$$\pi_t^{(n)} = \frac{f_t(\mathbf{s}_t^{(n)})}{g_t(\mathbf{s}_t^{(n)})} p(\mathbf{Z}_t|\mathbf{X}_t = \mathbf{s}_t^{(n)}). \quad (3)$$

The effect of the correction ratio is to preserve the information about motion coherence which is present in the dynamical model. Although the samples are positioned according to g_t, the distribution approximated by $\{(\mathbf{s}_t^{(n)}, \pi_t^{(n)})\}$ still generates $p(\mathbf{X}_t|\mathcal{Z}_t)$. Importance sampling is again intended to improve the efficiency of the sample-set representation, but does not change the probabilistic model. It should be noted that (2) imposes a restriction on the form of dynamical model which can be used; for CONDENSATION it is enough to be able to sample from $p(\mathbf{X}_t|\mathbf{X}_{t-1})$ but in the ICONDENSATION algorithm this density must also be evaluated. Existing implementations of CONDENSATION [14, 15, 13] use Gaussians or mixtures of Gaussians for this process density, and so evaluation is straightforward. The sum in (2) must be evaluated in (3) for each $n = 1, \ldots, N$, which changes the complexity of the algorithm from $O(N)^\star$ to $O(N^2)$. While this is a theoretical disadvantage, it has little effect in practice, since the time expended on the calculation of (2) in an efficient implementation, for practical values of N, is dwarfed by other stages of the computation.

In practice the importance function g_t will derive from an imperfect measurement process, so it may omit some likely peaks of $p(\mathbf{Z}_t|\mathbf{X}_t)$. It is therefore prudent to generate some samples using standard factored sampling and some by importance sampling using g_t. As long as $\tilde{p}(\mathbf{X}_t|\mathcal{Z}_{t-1})$ and g_t do not simultaneously fail to predict the object state, tracking will succeed.

It may also be advantageous to augment the dynamical model to include some probability q of reinitialisation — repositioning the object according to a prior which is independent of past history $\mathcal{Z}_t$. This allows a tracker to lock on to an object entering the scene, or rediscover an object which has been lost due to gross failures of measurements, perhaps because the object moved while it was entirely occluded. The amended model is of the form

$$\tilde{p}'(\mathbf{X}_t|\mathcal{Z}_{t-1}) = (1-q)\tilde{p}(\mathbf{X}_t|\mathcal{Z}_{t-1}) + qp(\mathbf{X}_t)$$

* It was claimed in [14] that CONDENSATION was $O(N \log N)$, however by choosing base samples as described in section 4.4 of this paper the complexity is reduced to $O(N)$.

where $p(\mathbf{X}_t)$ is the required initialisation prior. This is an application of mixed-state CONDENSATION [15] with two discrete states, and in later sections the model will be augmented to include a further two states. A mixed-state model can be included in the factored sampling scheme by choosing with probability $1-q$ to generate samples as before (using importance sampling with probability r and standard factored sampling with probability $1-q-r$) and with probability q to generate $\mathbf{s}_t^{(n)}$ by sampling directly from $p(\mathbf{X}_t)$. In the absence of another initialisation prior, the importance function, suitably normalised, can be used, so $p(\mathbf{X}_t) \propto g_t$. A complete sampling algorithm is shown in figure 1.

Iterate

From the "old" sample set $\{(\mathbf{s}_{t-1}^{(n)}, \pi_{t-1}^{(n)}),\ n = 1, \ldots, N\}$ at time-step $t-1$, construct a "new" sample set $\{(\mathbf{s}_t^{(n)}, \pi_t^{(n)}),\ n = 1, \ldots, N\}$ for time t. The importance function $g_t(\mathbf{X}_t)$ for time t is assumed to be known at this stage.

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

1. **Choose** the sampling method by generating a random number $\alpha \in [0, 1)$, uniformly distributed.
2. **Sample** from the prediction density $\bar{p}'(\mathbf{X}_t|\mathcal{Z}_{t-1})$ as follows:
   (a) **If** $\alpha < q$ use the initialisation prior. Choose $\mathbf{s}_t^{(n)}$ by sampling from $g_t(\mathbf{X}_t)$ and set the importance correction factor $\lambda_t^{(n)} = 1$.
   (b) **If** $q \leq \alpha < q + r$ use importance sampling. Choose $\mathbf{s}_t^{(n)}$ by sampling from $g_t(\mathbf{X}_t)$ and set $\lambda_t^{(n)} = f_t(\mathbf{s}_t^{(n)})/g_t(\mathbf{s}_t^{(n)})$, where

$$f_t(\mathbf{s}_t^{(n)}) = \sum_{j=1}^{N} \pi_{t-1}^{(j)} p(\mathbf{X}_t = \mathbf{s}_t^{(n)} | \mathbf{X}_{t-1} = \mathbf{s}_{t-1}^{(j)}).$$

   (c) **If** $\alpha \geq q + r$ use standard CONDENSATION sampling. Choose a base sample $\mathbf{s}_{t-1}^{(i)}$ with probability $\pi_{t-1}^{(i)}$, then choose $\mathbf{s}_t^{(n)}$ by sampling from $p(\mathbf{X}_t|\mathbf{X}_{t-1} = \mathbf{s}_{t-1}^{(i)})$ and set $\lambda_t^{(n)} = 1$.
3. **Measure** and weight the new position in terms of the image data $\mathbf{Z}_t$ and the importance sampling correction term:

$$\pi_t^{(n)} = \lambda_t^{(n)} p(\mathbf{Z}_t|\mathbf{X}_t = \mathbf{s}_t^{(n)})$$

then normalise multiplicatively so $\sum_n \pi_t^{(n)} = 1$ and store as $\{(\mathbf{s}_t^{(n)}, \pi_t^{(n)})\}$.

Fig. 1. ICONDENSATION: CONDENSATION **with importance sampling and reinitialisation.**

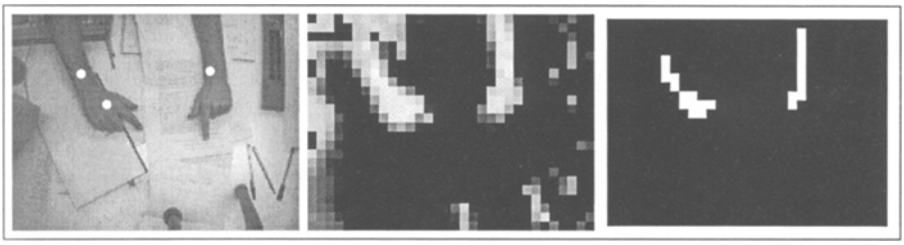

Fig. 2. Colour segmentation allows the detection of skin-coloured blobs. *Circles on the input frame (left) show the centres of blobs detected using colour-segmentation and a flood-fill. The output of the colour discriminant on a* 32×24 *pixel subsampled field of the original image is shown scaled (middle) so that white corresponds to a high probability of skin-colour. The output after convolution and blob-detection (right) shows white areas belonging to blobs.*

4 Experiments with a real-time hand-tracker

The framework of ICONDENSATION applies to any parametric representation of objects and their motion, and any form of importance function g_t. The remainder of this paper presents a specific implementation of a real-time hand tracker, combining blob-tracking with a contour model. First, a crude colour-segmentation technique is described which is used to construct the importance function $g_t(\mathbf{X}_t)$.

4.1 Finding skin-coloured blobs

Previous researchers [17, 10] have noted that human skin can be effectively distinguished in a typical office scene using colour segmentation, and so this method is adopted here. Training images of hands are used to construct a colour discriminant based on a Gaussian prior in (r, g, b) space which expresses the probability that a given pixel is skin-coloured. The prior is clipped with an intensity threshold to ensure that very dark pixels are not classified as skin. Blob-detection is performed by taking the input image and subsampling to give 32×24 pixels per field, then evaluating the colour discriminant for each pixel. A 2×2 moving block average is applied to the image to reduce noise, and then pixels are grouped using a flood-fill with hysteresis [7]. An example image and the segmented output is shown in figure 2. The technique has been found to be very effective in separating skin colour from background, and works over the variation in lighting conditions from day to night in our office. Let B be the number of blobs detected, then the mean of each blob is computed as a coordinate $\mathbf{b}'_k$ in the original image, and a two-dimensional importance function $\tilde{g}_t$ is defined to be a mixture of Gaussians over $\mathbb{R}^2$

$$\tilde{g}_t(\mathbf{x}_{trans}) = \sum_{k=1}^{B} \delta_k N(\mathbf{b}_k, \Sigma_B)$$

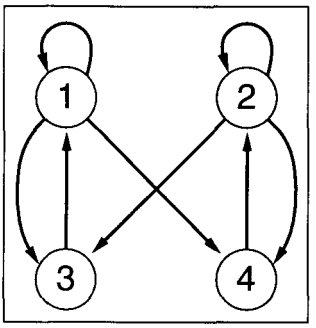

Fig. 3. The hand-tracker uses a four-state discrete/continuous motion model. *States 1 and 2 denote the left and right hand respectively, and states 3 and 4 correspond to initialising a sample with either the left or the right model.*

where $\mathbf{b}_k = \mathbf{b}'_k + \bar{\mathbf{x}}_B$, and $\bar{\mathbf{x}}_B$ and Σ_B are the mean and covariance respectively of the offset from the blob position to the centroid of the contour describing the hand. These are learned by following a user's hand using a contour tracker and comparing the output of the blob segmentation with the centroid of the tracked contour. The mixture weights δ_k will be discussed in the next section. A *hybrid* sampling scheme is now required since the importance function is defined only over translations, and this is outlined below.

4.2 A contour tracker for hands

A discrete-time second-order motion model is used to describe the hand, giving an augmented state-space

$$\mathbf{X}_t = \begin{pmatrix} \mathbf{x}_t \\ \mathbf{x}_{t-1} \end{pmatrix} \quad \text{where} \quad \mathbf{x}_t = (x_t, y_t, \theta_t, r_t, \chi_t)^T$$

and $\mathbf{x}_t$ is a non-linear representation of a Euclidean similarity transform applied to a template $\bar{\mathbf{x}}$. The extra parameter χ_t is a discrete label which determines whether the template is left- or right-handed — $\bar{\mathbf{x}}$ is reflected about the y-axis for the right hand. The left–right parameter χ_t is constant according to the motion model, and can only change as a result of a reinitialisation, and so the tracker effectively uses a four-state model (tracking or reinitialising either the left or the right hand) shown in figure 3. Note that the contour-based object representation allows accurate tracking of hand rotation and scaling as θ_t and r_t vary, which can be problematic for traditional correlation-based techniques. By the nature of the second-order representation, the lower half of $\mathbf{X}_t$ is found deterministically from $\mathbf{X}_{t-1}$, so we will refer interchangeably to $p(\mathbf{X}_t)$ and $p(\mathbf{x}_t)$ with slight abuse of notation.

The motion model is a second-order auto-regressive process (ARP) [3] where each of the four dimensions of the similarity is modelled by an independent

one-dimensional oscillator. The oscillators are specified by parameters defining a damping constant β, a natural frequency f and a root-mean-square average displacement ρ. These parameters can be used to determine the ARP model in one dimension $x_t = a_2 x_{t-2} + a_1 x_{t-1} + b\omega_t$ where ω_t is Gaussian noise drawn from $N(0,1)$, a_1, a_2 and b are given by [3]

$$a_2 = -\exp(-2\beta\tau), \quad a_1 = 2\exp(-\beta\tau)\cos(2\pi f\tau)$$
$$b = \rho\sqrt{1 - a_2^2 - a_1^2 - 2\frac{a_2 a_1^2}{1 - a_2}}$$

and τ is the time-step length in seconds (so $\tau = 1/30\,s$ for NTSC frame-rate). Sensible default parameters for the oscillators are chosen by hand.

As explained in section 4.1, the importance function $\tilde{g}_t$ is defined only over the space of x–y translations, being the output of a crude blob-tracker. The state-space decomposes into a translation subspace $\mathbf{x}_t^{\mathcal{T}} = (x_t, y_t)^T$ and a deformation subspace $\mathbf{x}_t^{\mathcal{D}} = (\theta_t, r_t, \chi_t)^T$. Modifications to steps 2(a) and 2(b) of figure 1 can now be made to implement a hybrid sampling scheme. For initialisation in step 2(a), the translation component $\mathbf{s}_t^{(n)\mathcal{T}}$ is sampled from $\tilde{g}_t$ and the deformation component $\mathbf{s}_t^{(n)\mathcal{D}}$ is sampled from a fixed prior $p^{\mathcal{D}}(\mathbf{x}^{\mathcal{D}})$ which is taken to be Gaussian in θ_t and r_t and assigns equal probabilities for left and right to χ_t, then $\mathbf{s}_t^{(n)} = \mathbf{s}_t^{(n)\mathcal{T}} \oplus \mathbf{s}_t^{(n)\mathcal{D}}$. The hybrid importance sampling step 2(b) proceeds as follows. First generate a sample $\mathbf{s}_t'^{(n)} = \mathbf{s}_t'^{(n)\mathcal{T}} \oplus \mathbf{s}_t'^{(n)\mathcal{D}}$ using standard CONDENSATION sampling as in step 2(c). Then choose $\mathbf{s}_t^{(n)\mathcal{T}}$ by sampling from $\tilde{g}_t$ and set $\mathbf{s}_t^{(n)} = \mathbf{s}_t^{(n)\mathcal{T}} \oplus \mathbf{s}_t'^{(n)\mathcal{D}}$. Finally the importance correction factor $\lambda_t^{(n)}$ is replaced by $\lambda_t^{(n)\mathcal{T}} = f_t^{\mathcal{T}}(\mathbf{s}_t^{(n)})/\tilde{g}_t(\mathbf{s}_t^{(n)})$, where

$$f_t^{\mathcal{T}}(\mathbf{s}_t^{(n)}) = \sum_{j=1}^{N} \pi_{t-1}^{(j)} p(\mathbf{X}_t^{\mathcal{T}} = \mathbf{s}_t^{(n)\mathcal{T}} | \mathbf{X}_{t-1} = \mathbf{s}_{t-1}^{(j)}).$$

It remains to specify the mixture weights δ_k for g_t. One reasonable choice is to set $\delta_k = 1/B$, and apportion samples equally in the vicinity of each blob. Since the motivation for importance sampling is to avoid generating samples with low weights, it may be preferable to increase δ_k for blobs which are near to many predicted sample positions. This can be done approximately by setting $\delta_k \propto f_t(\mathbf{b}_k)$, and later results are produced using these weights.

The prior distribution over translation for reinitialisation is chosen to be the distribution obtained by sampling from g_t with $\delta_k = 1/B$ for all k. The parameters θ and r are chosen from a suitable Gaussian prior density, with parameters set by hand, and χ has an equal chance of being left- or right-handed. When $B = 0$ no importance or reinitialisation samples are generated, and all of the computing time is spent on standard CONDENSATION samples.

4.3 The measurement process

Having detailed the dynamical model it remains to specify the measurement density $p(\mathbf{Z}_t|\mathbf{X}_t)$. This is assumed to be constant over time and dependent only

on the current configuration, so

$$p(\mathbf{Z}_t|\mathbf{X}_t) \equiv p(\mathbf{Z}|\mathbf{x}).$$

Recall that $\mathbf{x}$ specifies the outline in the image of a B-spline curve. The measurement density is approximated by examining a set of points $\mathbf{z}_m$ for $m = 1 \ldots M$ which lie on the curve outline, where the normal to the curve at $\mathbf{z}_m$ is $\mathbf{n}_m$. First of all, edge-operator convolutions are taken at $\mathbf{z}_m$ in the x and y directions, and the dot product of these is taken with the normal direction $\mathbf{n}_m$ to find a directed edge strength which is scaled and interpreted directly as a log probability p_m. When the edge strength is above a certain threshold an additional colour calculation is made to examine the pixels just inside the contour. This increases p_m when the area inside the curve scores highly according to the skin-colour discriminant, and decreases it when it scores poorly. Independence of the measurement points is assumed, so the density is given by

$$\log p(\mathbf{Z}|\mathbf{x}) = \text{const} + \sum_m p_m.$$

Constants are set manually, and the density is somewhat ad hoc; determining a more rigorous measurement density, possibly learned from training images, is deferred to future research.

4.4 Speed enhancements

Importance sampling has been presented as a mechanism to use complementary sources of visual information to choose an effective set of positions in state-space for a finite set of N samples. Given the constraint of real-time operation, a certain amount of care in the detailed implementation is necessary in order to maximise N. Much of this consists simply of standard code optimisation, but some parts of the algorithm can be redesigned for greater efficiency.

Base samples $\mathbf{s}_{t-1}^{(i)}$ are used both in standard CONDENSATION and for the hybrid importance sampling. These are chosen according to the $\pi_{t-1}^{(i)}$ and this selection can be done efficiently using cumulative probabilities $c_{t-1}^{(i)} = \sum_{j=1}^{i} \pi_{t-1}^{(i)}$. Quantising the cumulative probabilities

$$\tilde{c}_{t-1}^{(i)} = \lfloor c_{t-1}^{(i)} N \rfloor$$

suggests the following algorithm for generating samples which eliminates the $O(\log N)$ binary search stage in [14].

initialise: $i = 1$. **for** $n = 1 \ldots N$
  while $(\tilde{c}_{t-1}^{(i)} < n)$ $i{+}{+}$
  generate $\mathbf{s}_t^{(n)}$ by sampling from $p(\mathbf{X}_t|\mathbf{X}_{t-1} = \mathbf{s}_{t-1}^{(i)})$.

Since predictions from a given base sample are made consecutively this also offers a saving in calculating the deterministic portion of the prediction, which need only be done once per base sample. Practical experience shows that this

can lead to significant economy, since typically only 10% of samples may have high enough weight to be used as a base.

Software profiling shows that most of the tracker's computation is expended, as might be expected, on calculating the measurement density. It has been found that a significant speed improvement can be gained by presorting the measurement points in raster order before performing the convolution and colour-segmentation calculations of section 4.3. This has the advantage, as for the base samples, that identical measurement points are processed consecutively, which cuts down on the number of convolutions (typically the number of distinct measurement points is just over half of the total number of points). Clearly, performing the sort generates a large overhead, and in fact profiling reveals that typically 30% of time is spent sorting compared with about 50% making the measurements. This might seem to almost cancel the performance improvement from sorting, however the speedup on an SGI O2 is significantly greater than implied by these numbers, since the cache behaviour of evaluating points in raster order is much more benign than if the points are presented unsorted.

5 The tracker in operation

The hand-tracker has been implemented on an SGI O2 R5000 SC180 entry-level desktop workstation. The results shown were produced using $N = 400$ samples, which allows the tracker to run comfortably in real-time (30 Hz, using every other NTSC field). The number of samples can be increased to approximately $N = 575$ before any frames are dropped. Acceptable performance is obtained with $N = 150$ samples, and this runs comfortably at the field-rate of 60 Hz, although there is no noticeable benefit from using the additional fields. The main observed differences when using a smaller number of samples are a slight jitter when the hand is stationary and a longer time taken before the system reinitialises. As the number of samples is reduced below $N = 150$ the tracker begins to be distracted by clutter, although reinitialisation still functions to recover from these distractions.

The initialisation behaviour of the tracker is shown in figure 4. Initially the hand on the left is being tracked, and the spatial coherence inherent in the motion model means that the other blobs in the scene do not distract the tracker. When the thumb and forefinger are retracted, the shape no longer fits the template as accurately, and tracking reinitialises on the other hand within half a second. This behaviour corresponds to state transitions $2 \rightarrow 3 \rightarrow 1$ in figure 3. Figure 5 (a) demonstrates successful tracking at high speed, with interlace shown to indicate image velocities. Even if the hand makes a sudden movement which is not predicted by the motion model, the blob tracker will detect the new position, and importance samples will be generated in the vicinity of the hand allowing the motion to be tracked. Figure 5 (b) shows the advantage of using outline information as well as colour blob segmentation. The hands are adjacent, so their blobs merge, yet tracking continues to distinguish between the hands.

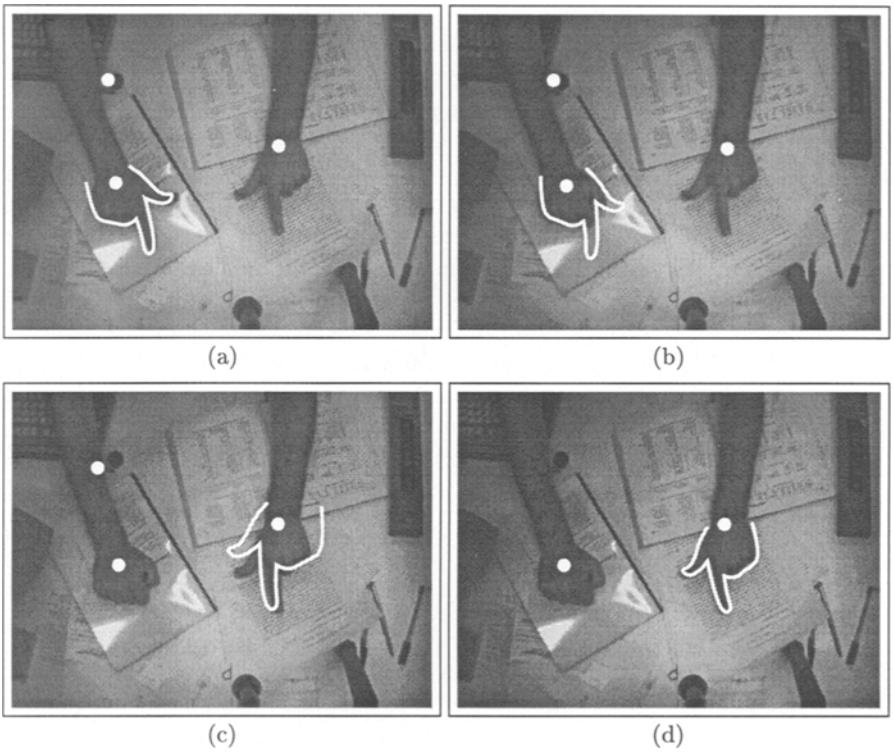

Fig. 4. The tracker reinitialises when one hypothesis takes precedence over another. *While the user's right hand fits the template well (a) the motion coherence in the dynamical model ensures that the tracker remains locked on (discrete state 2 in figure 3). When the shape no longer fits the template (b) the probability of reinitialising to another hypothesis increases. After 5 frames (1/6 s) the tracker has switched to the neighbourhood of the left hand, selecting the left-handed template (discrete state 3 followed by 1), and another 5 frames later, a total of 1/3 s from the time that the right hand no longer fit the template, the tracker has locked on to the user's left hand (discrete state 1). Detected blobs are shown as circles, and N=400 samples are used.*

6 Extending the tracker for multiple users

The hand tracker described so far is specialised to a single hand shape, encoded in the template $\bar{\mathbf{x}}$. This requires the user's hand to be held fairly rigidly in the template pose, and necessarily means that some users' hands will fit the template better than others'. The main problem with an ill-fitting template is slow re-initialisation, but it also increases the chance of clutter distractions. It would be desirable, therefore, to allow some variation in $\bar{\mathbf{x}}$. A shape-space of hand-deformations was therefore established by using Principal Components Analysis (PCA) on sequences of images collected from several subjects [4]. Each subject placed his or her left hand in a reference position and orientation, with

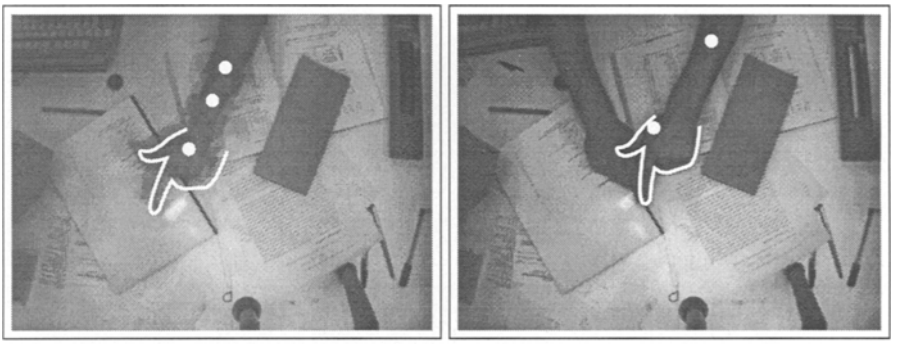

Fig. 5. Tracking is robust to high-speed motion and failures of colour segmentation. *Both fields of a frame are shown in (a) and interlacing artifacts demonstrate the rapid translation of the hand. When the hands move close to each other in (b), their colour-segmentation blobs merge, giving a mean value (circle) between the hands. The contour tracker continues to follow the left hand using motion coherence and edge information. The tracker is using N=400 samples.*

thumb and forefinger outstretched, and then made small movements to represent the variation of poses in which that user's hand will be presented to the system. These movements were recorded by a B-spline tracker, the spline positions from the separate sequences were concatenated, and PCA was used to find a six-dimensional space of deformations. In the interests of real-time tracking, it is not desirable to increase the dimension of $\mathbf{x}_t$ from 4 continuous Euclidean similarity parameters to 10 for rigid transformations plus deformation.

The solution adopted was to run two entirely separate trackers, one in the Euclidean similarity space as before, and one in a separate six-dimensional deformation space with N_D samples and parameter $\mathbf{y}_t$, so that $\bar{\mathbf{x}} = \bar{\mathbf{x}}_0 + W\mathbf{y}_t$ where $\bar{\mathbf{x}}_0$ and W were estimated by PCA. Since $\mathbf{y}_t$ is expected to vary slowly, only a small number of samples need be used in the deformation tracker, and good results are obtained using $N_D = 50$ samples. At each time-step, the trackers are run consecutively. First of all $\mathbf{y}_t$ is held fixed at $\mathbf{y}_t = \hat{\mathbf{y}}_{t-1}$ to establish $\bar{\mathbf{x}}_t = \bar{\mathbf{x}}_0 + W\hat{\mathbf{y}}_{t-1}$. The Euclidean transformation tracker is then run, exactly as before, and an estimated Euclidean vector $\hat{\mathbf{x}}_t$ is calculated from (1). Now the Euclidean transformation is held fixed at $\mathbf{x}_t = \hat{\mathbf{x}}_t$ and the deformation tracker is run to estimate the new sample-set distribution for $\mathbf{y}_t$ from which $\hat{\mathbf{y}}_t$ can be estimated, again using (1). This procedure is not entirely satisfactory, since it prevents a meaningful probabilistic interpretation of the state densities. It would be preferable to combine the estimation of deformation and rigid motion in a consistent Bayesian framework while keeping the economy of computation, and this is discussed briefly in section 7. Despite this caveat, results using the two-tracker system are promising. The deformation-space tracker was run as a standard CONDENSATION tracker using dynamics learned from the hand-shape

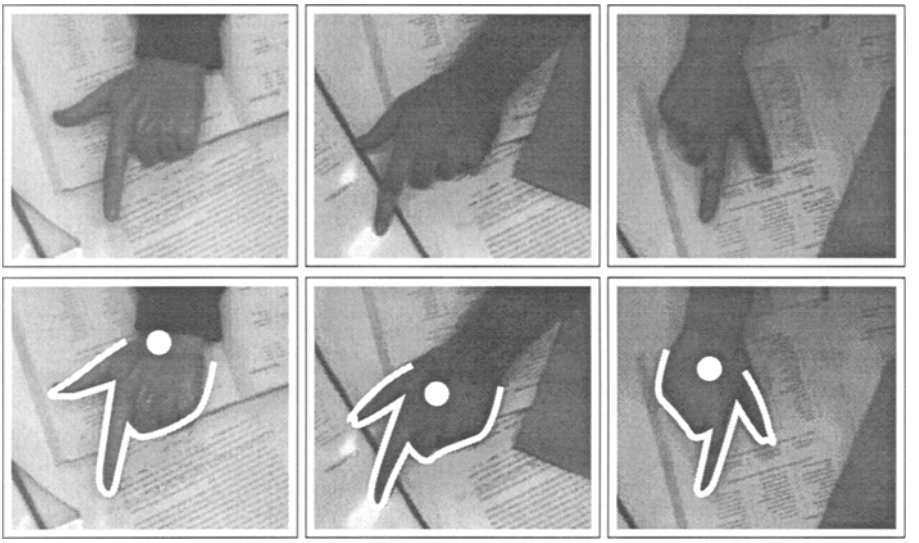

Fig. 6. The deformation tracker accommodates different hands. *Columns show different users. The lower images show tracked output approximately a second after the hand was introduced into the scene. The left-hand user was included in hand-shape training data, the other two were not. In this example $N=400$ for the Euclidean similarity tracker and $N_D=50$ for the deformation tracker.*

training sequences [4, 14]. The shape rapidly deforms to fit a hand as it enters the scene, and figure 6 shows that the tracker adapts to different hands as required.

7 Conclusions and future work

The hand-tracking system presented in this paper has proved to be extremely robust and agile over a wide range of desk clutter, for every test user who has so far tried it (about 10 people). Since it runs comfortably in real time, it offers the exciting possibility of being used as the user-interface back-end to drive some kind of interactive package. Perhaps as robust yet detailed trackers such as this are developed, the promise of the digital desk [23] will at last begin to be realised.

In the last few years real-time tracking has rapidly progressed, and expertise has been developed in exploiting a wide range of methodologies. ICONDENSATION offers a bridge between some competing approaches, and thus there exists considerable scope for building on the work presented here. Hand-tracking is a very specialised area since colour segmentation of skin is so accurate. It may be possible to build similar systems using other low-level approaches such as motion segmentation and image-differencing, opening the framework to other application domains. Within the hand-tracking system certain problems remain. Before

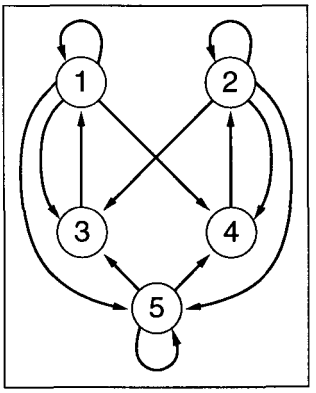

Fig. 7. A five-state model could encompass the possibility that no hand is present. *States 1 and 2 denote tracking the left and right hand, respectively, states 3 and 4 correspond to initialising a sample to start tracking with either the left or the right model, and state 5 is entered when no hand is present in the scene.*

the system can be used as an input device it will probably be necessary to devise a robust measure to determine whether or not the hand is in the scene, possibly using a contour discriminant [19]. The discrete model would then be expanded to include a fifth state (figure 7), denoting the event that no hand is present. A better method of accommodating deformations of the hand-template may also be possible. It is profligate to allow an entire 6-dimensional linear space to represent hand deformations, since most of the space consists of shapes which are very unlike a hand. It may be possible to collect a large number (perhaps 20–100) of template shapes, and include them as discrete states in the model, relying on each user's hand to be close to one of the example templates. Alternatively it may be possible to specify a non-linear parameterisation of the deformations which is efficient enough to allow the deformation space to be included directly with the Euclidean similarity parameters in the main tracker.

Finally it is worth noting that the combination of low-level and high-level approaches presented here may be applicable to problems which have been tackled using multi-resolution techniques. Importance sampling allows measurements to be combined even when no knowledge is available about their statistical dependence, and therefore may be applicable to other tasks where measurements are available at different granularities, but their dependence is poorly understood.

References

1. A. Baumberg and D. Hogg. Generating spatiotemporal models from examples. In *Proc. British Machine Vision Conf.*, volume 2, 413–422, 1995.
2. M.J. Black and A.D. Jepson. Eigentracking: robust matching and tracking of articulated objects using a view-based representation. In *Proc. 4th European Conf. Computer Vision*, 329–342, 1996.

3. A. Blake and M.A. Isard. *Active contours.* Springer, 1998.
4. A. Blake, M.A. Isard, and D. Reynard. Learning to track the visual motion of contours. *J. Artificial Intelligence*, 78:101–134, 1995.
5. T.F. Cootes, C.J. Taylor, A. Lanitis, D.H. Cooper, and J. Graham. Building and using flexible models incorporating grey-level information. In *Proc. 4th Int. Conf. on Computer Vision*, 242–246, 1993.
6. J. Costeira and T. Kanade. A multi-body factorization method for motion analysis. In *Proc. 5th Int. Conf. on Computer Vision*, 1071–1076, 1995.
7. J.D. Foley, A. van Dam, S.K. Feiner, and J.F. Hughes. *Computer Graphics: Principles and Practice*, chapter 13, 563–604. Addison-Wesley, 1990.
8. N. Gordon and D. Salmond. Bayesian state estimation for tracking and guidance using the bootstrap filter. *Journal of guidance, control and dynamics*, 18(6):1434–1443, November–December 1995.
9. 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.
10. H.P. Graf, E. Cosatto, D. Gibbon, M. Kocheisen, and E. Petajan. Multi-modal system for locating heads and faces. In *Proc. 2nd Int. Conf. on Automatic Face and Gesture Recognition*, 88–93, 1996.
11. U. Grenander, Y. Chow, and D.M. Keenan. *HANDS. A Pattern Theoretical Study of Biological Shapes.* Springer-Verlag. New York, 1991.
12. G. Hager and K. Toyama. X vision: Combining image warping and geometric constraints for fast visual tracking. In *Proc. 4th European Conf. Computer Vision*, volume 2, 507–517, 1996.
13. T. Heap and D. Hogg. Wormholes in shape space: Tracking through discontinuous changes in shape. In *Proc. 6th Int. Conf. on Computer Vision*, 1998.
14. 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.
15. 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.
16. 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.
17. R. Kjeldsen and J. Kender. Toward the use of gesture in traditional user interfaces. In *Proc. 2nd Int. Conf. on Automatic Face and Gesture Recognition*, 151–156, 1996.
18. D.G. Lowe. Robust model-based motion tracking through the integration of search and estimation. *Int. J. Computer Vision*, 8(2):113–122, 1992.
19. J. MacCormick and A. Blake. A probabilistic contour discriminant for object localisation. In *Proc. 6th Int. Conf. on Computer Vision*, 390–395, 1998.
20. B.D. Ripley. *Stochastic simulation.* New York: Wiley, 1987.
21. Torr P.H.S. An assessment of information criteria for motion model selection. In *Proc. Conf. Computer Vision and Pattern Recognition*, 1997.
22. Torr P.H.S. and Murray D.W. Stochastic motion clustering. In *Proc. 3rd European Conf. Computer Vision*, 328–337. Springer-Verlag, 1994.
23. P. Wellner. The Digital Desk calculator — tangible manipulation on a desk-top display. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, Hilton Head, November 1991.
24. C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: Real-time tracking of the human body. In *Proc. 2nd Int. Conf. on Automatic Face and Gesture Recognition*, 51–56, 1996.