BraMBLe: A Bayesian Multiple-Blob Tracker

M. Isard and J. MacCormick
Compaq Systems Research Center
Palo Alto, CA 94301, USA

Abstract

Blob trackers have become increasingly powerful in recent years largely due to the adoption of statistical appearance models which allow effective background subtraction and robust tracking of deforming foreground objects. It has been standard, however, to treat background and foreground modelling as separate processes — background subtraction is followed by blob detection and tracking — which prevents a principled computation of image likelihoods. This paper presents two theoretical advances which address this limitation and lead to a robust multiple-person tracking system suitable for single-camera real-time surveillance applications.

The first innovation is a multi-blob likelihood function *which assigns directly comparable likelihoods to hypotheses containing different numbers of objects. This likelihood function has a rigorous mathematical basis: it is adapted from the theory of Bayesian correlation, but uses the assumption of a static camera to create a more specific background model while retaining a unified approach to background and foreground modelling. Second we introduce a Bayesian filter for tracking multiple objects when the number of objects present is unknown and varies over time. We show how a particle filter can be used to perform joint inference on both the number of objects present and their configurations. Finally we demonstrate that our system runs comfortably in real time on a modest workstation when the number of blobs in the scene is small.*

1. Introduction

This paper brings together two rapidly developing areas of research in the field of visual tracking: blob tracking and particle filtering. Blob trackers have been very successful in following single objects through a scene [11, 19], particularly when stereo information is available [10]. Typically the foreground object is modelled as an ellipse in the image plane and the object is matched from frame to frame either by correlation [4] or using some statistical properties of the object; colour histograms [1] have been notably successful in this respect. To improve robustness when the camera is static, blob trackers typically make use of sophisticated background subtraction. Statistical background subtraction based on the output of filter banks has been found to be particularly effective [5]. However, the temporal filter used for tracking blobs is generally a simple one—usually a constant velocity predictor followed by matching to the closest foreground patch in the neighbourhood of the prediction. This can be very effective in simple scenes, especially when only one object is present. There has been some work applying blob tracking to multiple objects, particularly in the domain of person tracking [3, 5, 6]; however, the lack of a sophisticated filter typically means occlusion reasoning is rudimentary and blobs corresponding to separate people are merged when the people come close together in the image, splitting again when the people separate. This makes it difficult to provide unique track labels for different people which would be very useful in surveillance applications. The problem is reduced by using a camera mounted high enough that the blobs are viewed from near vertical [3] but this is often impossible for indoor surveillance.

Particle filtering [2, 7] has been successfully applied to tracking objects in clutter, but has typically been used with edge-based [8] or kinematic [9, 16] imaging models both of which rely on an accurate model of the object shape for robust tracking. Importance sampling [8] has been used to combine blob detectors with an edge-based particle filter but the blobs were used primarily for initialisation; the filter still relied on an accurate outline model of the object for edge-based tracking. Sullivan et al. [17] proposed a region-based imaging model, Bayesian correlation, which is suitable for particle filtering. By using information in the entire region enclosed by a contour, Bayesian correlation allows successful tracking despite an imprecise object model. Furthermore, the flexibility of particle filtering permits a principled multiple-object tracker in which the number of objects and their locations are simultaneously estimated from image data. Multiple-object particle filtering was used in [18] but the hierarchical method proposed there does not permit the Bayesian interpretation central to the approach of this paper. The Markov chain Monte Carlo jump-diffusion methods of [14] are also related to multiple-object particle filtering, but the approach developed here can be used in a sequential filter and so is more suitable for online surveillance applications.

The Bayesian Multiple-BLob tracker (BraMBLe) presented here is, as far as we are aware, the first rigorous implementation of a particle filter in which the number of objects being tracked may vary during tracking. The key development is the use of a fast and robust observation model which accurately reflects the likelihood of differing numbers of objects being present; once such a model is available, it turns out that a standard particle filter can be applied to yield a posterior distribution over the number and configurations of objects. This is explained in section 3. The observation model introduced here, termed the multi-blob likelihood function, is based on the theory of Bayesian correlation [17], extended to give much increased speed and robustness in the context of a static, calibrated camera. The new model uses individually learnt background patches, pooled foreground patches, and geometric reasoning from the camera calibration. Details are given in section 2.

The resulting system, BraMBLe, uses a single static camera to robustly track several people as they enter, exit and move about in a scene. When one or two people are present, the system runs in real time on a modest single-processor workstation, and with offline processing it tracks complex scenes involving several people.

2. Observation likelihood

This section describes the *multi-blob likelihood function* $p(Z|X)$, which expresses the likelihood that a hypothesised configuration X of objects gave rise to an observed image Z. The configuration space for the blob tracking task is the union of all possible configurations of $0, 1, 2, \ldots$ objects; a typical element is written

$$X = (m, x^1, \ldots, x^m)$$

where m is the number of objects, or blobs, present and x^i is a vector encoding the state of the ith object. It is often useful to consider the position and shape of an object separately, so we decompose the state further as $x^i = (\mathcal{X}^i, \mathcal{S}^i)^T$. In the particular application of person-tracking described below, $\mathcal{X}$ is a 2D floor position in world coordinates and $\mathcal{S}$ gives the shape of a vertically-aligned generalised cylinder — details are given in section 2.2.

The multi-blob likelihood function $p(Z|X)$ is computed using a variation on the Bayesian correlation scheme presented by Sullivan et al. [17]. As in [17], the image is overlaid with a fixed grid of G locations $\{(u_g, v_g) : g = 1, \ldots, G\}$. In the experiments described below a rectangular grid is used with locations spaced at 5-pixel intervals in the horizontal and vertical image plane directions. At each location g, a bank of filters is applied to the image patch centred on (u_g, v_g) yielding a response vector z_g. In our experiments six filters are used: three radially symmetric Gaussians, one each for the Y, Cr and Cb image channels, and three radially symmetric Mexican hat functions (second derivative of Gaussians), again one for each image channel. All filters have standard deviation of 1.5 pixels and are truncated at radius 5 pixels. The response values are assumed conditionally independent given X, so

$$p(Z|X) = \prod_{g=1}^{G} p(z_g|X).$$

In [17] it is argued that while the Mexican hat filters can be treated as independent at the grid resolution we use, non-zero mean filters such as Gaussians are in fact correlated at these spacings. However, these dependencies are substantially mitigated in our model which adopts a different learnt response likelihood function for every background patch. Future work will quantify the reduction in correlation from this effect.

To calculate the individual response likelihoods $p(z_g|X)$ we assign each response g a label $l_g \in \{0, 1, \ldots, m\}$ according to whether X hypothesises that image patch g is centred in the background ($l_g = 0$) or one of the m foreground objects ($l_g = i$ for object i) and define $p(z_g|X) = p(z_g|l_g)$. Note that this is a simplification of [17] in which some patches were assigned a mixture of background and foreground. The likelihood can now be rewritten as

$$p(Z|X) = \prod_{g=1}^{G} p(z_g|X) = \prod_{g=1}^{G} p(z_g|l_g). \qquad (1)$$

Using a learned model of the background and foreground (described below) the terms in the right hand product in (1) can be computed directly. In fact, by pre-calculating $y_g^l = \log(p(z_g|l))$ for $g = 1, \ldots, G$ and $l \in \{0, 1, \ldots, m\}$ we can efficiently compute the log likelihood of any hypothesis X given the set of labels l_g simply by performing table lookups and additions. This can be further optimised by noting that the particle filter described in section 3 requires the likelihood at each time step only up to a multiplicative constant. Hence we are free to define

$$y_g^l = \log(p(z_g|l)) - \log(p(z_g|0)) \qquad (2)$$

so that $y_g^0 = 0$ for all g and only foreground responses contribute to the log likelihood. Note that y_g^l in (2) specifies the log-likelihood ratio comparing the hypothesis that the response was generated by object l with the hypothesis that it was generated by background. With the y_g^l computed from (2), the algorithm for label assignment and computation of the log likelihood L of the hypothesis X is given in figure 1. Step 1 is a simple way of enforcing an exclusion principle [13, 15] by assigning zero likelihood to hypotheses in which distinct objects occupy the same physical space in the world ($\| \bullet \|$ denotes Euclidean distance in $\mathbb{R}^2$) but note that objects can still overlap or completely occlude

```
1. if $\|\mathcal{X}^i - \mathcal{X}^j\| < \delta_{\mathrm{e}}$ for any $i \neq j$, set $L := -\infty$ and halt; otherwise, set $L := 0$ and continue.
2. for $g = 1$ to $G$ set $l_g := 0$.
3. sort the objects in depth order where $\psi(i), i = 1, \ldots, m$ enumerates the objects, closest first.
4. for $i = 1$ to $m$
   (a) compute the set $G_{\mathrm{obj}}^{\psi(i)}$ of responses in the image region occupied by object $\psi(i)$. If $G_{\mathrm{obj}}^{\psi(i)} = \emptyset$ set $L := -\infty$ and halt.
   (b) for all $g \in G_{\mathrm{obj}}^{\psi(i)}$ such that $l_g = 0$:
       i. set $L := L + y_g^{\psi(i)}$
       ii. set $l_g := \psi(i)$
```

Figure 1: **The multi-blob likelihood algorithm**

each other in the image. Hypotheses in which an object lies entirely off the visible image are rejected in step 4a. Section 2.2 describes the key used for sorting in step 3 and an efficient algorithm for computing $G_{\mathrm{obj}}^{\psi(i)}$ in step 4a.

2.1. Statistical appearance models

In [17], the foreground and background models were both pooled from a broad training set, since the application was primarily to locate objects in novel scenes. We can take advantage of our static camera assumption to adopt a precisely tuned background model where each background response likelihood is learned independently, much like the background subtraction scheme in [5]. Note that assuming a static camera is not the same as assuming a static background: motion and other variation (moving trees, twinkling surfaces) in the background are incorporated into the response likelihoods automatically. We adopt a 4-component mixture of Gaussians model for each response:

$$p(z_g|l_g = 0) = \sum_{k=1}^{4} \frac{1}{4} \mathcal{N}(\mu_g^k, \Sigma_g^k) \qquad (3)$$

learnt by performing k-means clustering with $k = 4$ on training data and discarding any very small clusters. Then μ_g^k takes the mean of cluster k and Σ_g^k is set to the diagonal approximation to the covariance of the cluster. We add a small multiple of the identity matrix $\Delta_B = \delta_B I$ to each learned covariance which is sufficient to suppress false foreground responses to shadows. Finally, the mixture of Gaussians is itself mixed with a lightly-weighted uniform distribution to provide robustness against responses not observed in the foreground or the background while training; this is equivalent to adding a small constant τ_B to the likelihood (3):

$$p(z_g|l_g = 0) = \sum_{k=1}^{4} \frac{1}{4} \mathcal{N}(\mu_g^k, \Sigma_g^k + \Delta_B) + \tau_B.$$

We currently learn a separate static background model for each test sequence using a few hundred empty background images recorded at the same time as the test sequence; a background model which updates online would be necessary for extended sequences of many hours.

Our foreground model is pooled among all discs labelled as foreground. The framework allows separate models for distinct foreground objects but we have not yet exploited this opportunity. We use a 16-component mixture of Gaussians learned using k-means in the same way as the background model (but without the addition of a constant covariance), so

$$p(z_g|l_g \neq 0) = \sum_{k=1}^{16} \frac{1}{16} \mathcal{N}(\mu_{\mathrm{fore}}^k, \Sigma_{\mathrm{fore}}^k) + \tau_F. \qquad (4)$$

Training foreground responses are generated using a training image sequence which contains foreground objects moving against a static background. Each filter response which is sufficiently unlikely to have been generated by a learned model of the background is included as a training response for the foreground clustering algorithm. Figure 2 shows the log-likelihood ratios y_g^l for a sample image.

The foreground model is pooled in the spirit of a colour histogram model [1] but is less powerful since the z_g are assumed conditionally independent — we can only express the notion that each response from a given foreground object is drawn from the same histogram rather than being able to specify a joint histogram of the responses. Future work is necessary to determine a good way of incorporating dependencies between foreground responses within the Bayesian correlation framework.

2.2. A generalised-cylinder object model

In person-tracking experiments below we use a calibrated camera to project from world coordinates to camera coordinates and subsequently to the image plane. A person is modelled as a generalised cylinder whose axis is vertical in the world coordinate frame. The shape of the cylinder is specified by the radius and height of four horizontal discs $(r_i, y_i), i = 1, \ldots, 4$ (figure 3). The object configuration vector $(\mathcal{X}, \mathcal{S})$ specifies a floor position $\mathcal{X} = (x, z)^T$ and a shape

$$\mathcal{S} = (w_f, w_w, w_s, w_h, h, \theta, \alpha_w, \alpha_s) \qquad (5)$$

from which the disc parameters (r_i, y_i) for the feet, waist, shoulders and top of the head respectively are computed as

$$(w_f, 0), (w_w\theta, \alpha_w h), (w_s\theta, \alpha_s h), (w_h, h).$$

The parameter $0.5 \leq \theta \leq 1$ encodes how much the person has turned away from the camera, so when $\theta = 0.5$ the person is viewed side on. The waist and shoulder heights are modelled as a proportion (respectively α_w and α_s) of

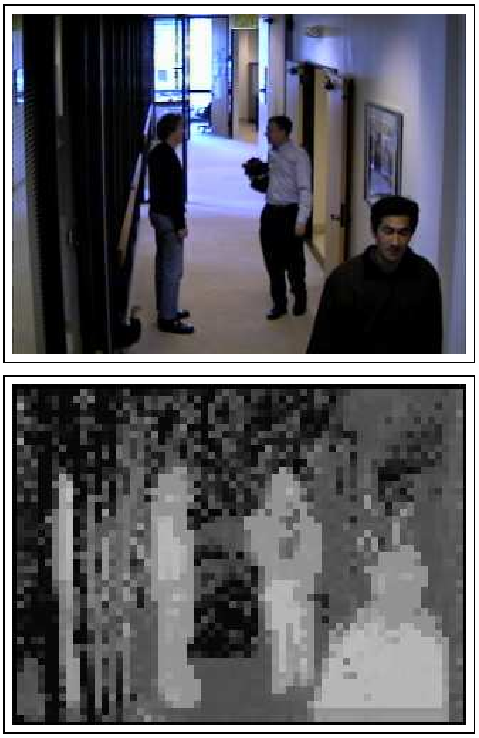

Figure 2: **Log-likelihood ratios y^l_g using learned foreground and background models.** *Whiter values are more likely to be drawn from the foreground model. Note the foreground responses from the reflection in the glass wall on the left — these cannot easily be suppressed, unlike shadows.*

the total height h. This avoids introducing implausibly large variances for the sizes of the head and torso.

The likelihood algorithm in figure 1 requires that objects be sorted in depth order; we project the world-coordinate centre $(x, 0, z)^T$ of each cylinder base to camera coordinates and use the resulting depth as the sort key. G_{obj} in figure 1 is found by rendering the object into the image plane at the resolution of the response grid (figure 4).

3. A Bayesian multiple-object filter

A key advantage of the multi-blob likelihood is that it can be used as the observation model for a particle filter. In particular, the multi-blob likelihood assigns directly comparable likelihoods to hypotheses containing different numbers of objects. Thus the standard machinery of particle filters can be used to produce a filter which tracks an unknown and varying number of objects.

The output of a Bayesian time-series filter is the posterior probability distribution $p(X_t|X_0, Z_{1:t})$, where X_t is the system state at time t, X_0 is a prior distribution and $Z_{1:t}$ is a sequence of observations from times 1 to t. A particle filter approximates this posterior using a weighted particle set $\{(X^n_t, \pi^n_t) : n = 1, \ldots, N\}$. We use here the simple particle filter referred to as CONDENSATION in [7], outlined in figure 5. The notation of section 2 must be augmented because we now want to track distinctly identified objects as they move over time: an object state becomes

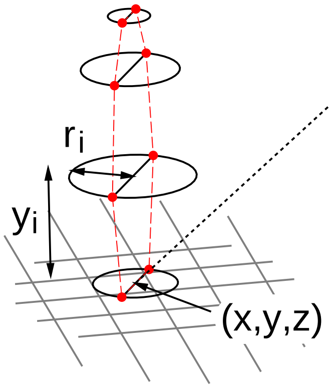

Figure 3: **Approximate projection of a generalised cylinder** *with base position (x, y, z) and k discs of radius and height $(r_1, y_1, \ldots, r_k, y_k)$ (in experiments $y = y_1 = 0$ and $k = 4$). A ray is projected from the camera's optical centre to (x, y, z) and the horizontal line perpendicular to that ray defines a vertical plane facing the camera. The approximate projection is given by the polygon (red dotted line) found by intersecting that plane with the cylinder.*

$$\tilde{x}^{n,i}_t = (\phi^{n,i}_t, x^{n,i}_t)$$

where $\phi^{n,i}_t$ is a unique identifier labelling which object is being referred to and $x^{n,i}_t = (\mathcal{X}^{n,i}_t, \mathcal{V}^{n,i}_t, \mathcal{S}^{n,i}_t)^T$ is the position and shape of the object as in section 2 now augmented with a velocity $\mathcal{V}^{n,i}_t = (v_x, v_z)^T$. The state of the system at time t is $X_t = (m_t, \tilde{x}^1_t, \ldots, \tilde{x}^{m_t}_t)$ and a particle is denoted

$$X^n_t = (m^n_t, \tilde{x}^{n,1}_t, \ldots, \tilde{x}^{n,m_t}_t). \qquad (6)$$

The prediction model $p(X_t|X_{t-1})$ states that each object will remain in the scene with probability λ_r at each time step, and additionally that there is probability λ_i that a new object will enter the scene at each time step (this is consistent with a Poisson distribution on object arrivals and an exponential distribution on their survival times). The algorithm for generating a new particle hypothesis $X^n_t = (m^n_t, \tilde{x}^{n,1}_t, \ldots)$ from a previous particle $X^{n'}_{t-1} = (m^{n'}_{t-1}, \tilde{x}^{n',1}_{t-1}, \ldots)$ is given in figure 6.

In our experiments the translational dynamics of $\mathcal{X}$ are modelled as damped constant velocity plus Gaussian noise and each shape parameter $s_i \in \mathcal{S}, i = 1, \ldots, 8$ obeys an independent 1st order auto-regressive process model with

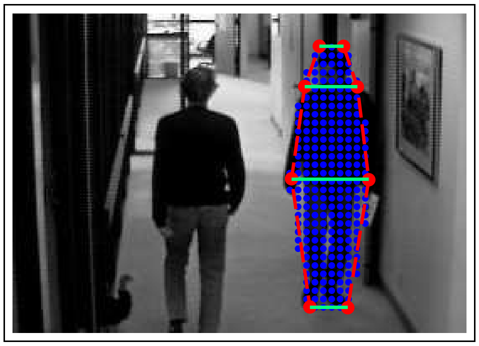

Figure 4: **Approximate rendering of a generalised cylinder.** *The red dots show the vertices of the polygon computed in figure 3. Pairs of vertices projected from the same disc have the same world height. We assume the camera is mounted nearly horizontal and the lens is not too distorting so these horizontal world lines are well-approximated by horizontal image lines shown in green. The set G_{obj} of responses in the region enclosed by the object (blue circles) is found by interpolating between the horizontal lines at the resolution of the grid G.*

```
1. initialise $\{(X_0^n, \pi_0^n)\}_{n=1}^N$ from the prior distribution $X_0$.
2. for $t > 0$:
   (a) resample $\{(X_{t-1}^n, \pi_{t-1}^n)\}$ into $\{({X'}_{t-1}^n, 1/N)\}$.
   (b) predict, generating $X_t^n \sim p(X_t | X_{t-1} = {X'}_{t-1}^n)$ to give $\{(X_t^n, 1/N)\}$.
   (c) weight, setting $\pi_t^n \propto p(Z_t | X_t = X_t^n)$ to give $\{(X_t^n, \pi_t^n)\}$ normalised so $\sum_{n=1}^N \pi_t^n = 1$.
   (d) estimate $\hat{X}_t$ for display.
```

Figure 5: **The CONDENSATION algorithm**

```
1. set $m_t^n := 0$.
2. for $i = 1$ to $m_{t-1}^{n'}$:
   (a) generate $r$ distributed as $U[0,1)$.
   (b) if $r < \lambda_r$ set $m_t^n := m_t^n + 1$ and $\tilde{x}^{n,m_t^n} := f(\tilde{x}_{t-1}^{n',i})$
3. generate $r$ distributed as $U[0,1)$.
4. if $r < \lambda_i$ set $m_t^n := m_t^n + 1$ and set $\tilde{x}^{n,m_t^n} := g(t)$
```

Figure 6: **The multi-object prediction algorithm**

mean, steady-state standard deviation and process noise μ_i, σ_i and ρ_i respectively [12]. The turn parameter θ is additionally hard-limited to lie in the range $0.5 \leq \theta \leq 1$ (see section 2.2). The prediction function f preserves the unique identifier of an object and generates a new position and shape vector subject to the object dynamics described above:

$$f(\phi, (\mathcal{X}, \mathcal{V}, \mathcal{S})^T) = (\phi, (\mathcal{X}', \mathcal{V}', \mathcal{S}')^T)$$

$$\begin{aligned}
\mathcal{X}' &= \mathcal{X} + \lambda_v \mathcal{V} + b_X \omega_X \\
\mathcal{V}' &= \lambda_v \mathcal{V} + b_X \omega_X \\
\mathcal{S}' &= A_S(\mathcal{S} - \overline{\mathcal{S}}) + B_S \omega_S \\
\overline{\mathcal{S}} &= (\mu_1, \ldots, \mu_8) \\
B_S &= \mathrm{diag}(\rho_1, \ldots, \rho_8) \\
A_S &= \mathrm{diag}(a_1, \ldots, a_8)
\end{aligned} \tag{7}$$

where ω_X and ω_S are vectors of i.i.d. standard Gaussian-distributed random variables and $a_i = 1 - \sqrt{\rho_i^2/\sigma_i^2}$. Values for λ_v, b_X and the ARP parameters μ_i, σ_i and ρ_i used in experiments are given in section 4. The initialisation function g assigns the unique identifier t and generates a position, velocity and shape according to an object prior distribution:

$$g(t) = (t, ((x,z)^T, (0,0)^T, \mathcal{S})^T)$$

where (x,z) is a random position drawn uniformly from a rectangle corresponding to the visible floor area and $\mathcal{S}$ is drawn from the steady-state distribution of the shape ARP. For implementation reasons we limit the total number of distinct objects which can be present in a particle set to M_{max}, so we keep track of $\mathcal{M}_t = \{\Phi_1, \ldots, \Phi_{M_t}\}$, the set of all unique identifiers present in any particle in the distribution at time t. We only generate initialisation samples at time t if $M_{t-1} = |\mathcal{M}_{t-1}| < M_{\mathrm{max}}$. When no performance constraints are in effect M_{max} is set sufficiently large that initialisation samples are always generated. The initialisation at time $t = 0$ is particularly simple as each particle has equal weight and consists of the hypothesis that there are zero objects: $X_0^n = (0)$ in the notation of equation (6).

The algorithm to estimate $\hat{X}_t$ is shown in figure 7, where $\Pi_t^{\Phi_i}$ is the total probability that object Φ_i is present. This number is an asymptotically correct marginal probability, in the sense that it tends to the true marginal probability as the number of particles used tends to infinity. The parameter λ_d determines whether or not an object is displayed, and has no effect on the particle filter itself.

```
for $i = 1$ to $M_t$
   (a) compute $\mathcal{M}_t^{\Phi_i} = \{(n,j) : \phi_t^{n,j} = \Phi_i\}$.
   (b) compute $\Pi_t^{\Phi_i} = \sum_{(n,j) \in \mathcal{M}_t^{\Phi_i}} \pi_t^n$.
   (c) if $\Pi_t^{\Phi_i} > \lambda_d$ estimate $\hat{x}_t^{\Phi_i} = \sum_{(n,j) \in \mathcal{M}_t^{\Phi_i}} \pi_t^n s_t^{(n,j)} \,/\, \Pi_t^{\Phi_i}$.
```

Figure 7: **The estimation algorithm**

4 Results

Numerical values for the dynamical parameters used in all experiments are given in figure 9. We describe two scenarios: the first demonstrates the full power of the multiple-blob tracking but runs somewhat slower than real-time on a 667 MHz Alpha workstation. Secondly we demonstrate real-time settings for the filter which allow it to run comfortably in real time on a 447 MHz Pentium II workstation.

We recorded a 53-second sequence at CIF image resolution (320×240 pixels, $G = 63 \times 47 = 2961$) and 30 frames/second showing up to three people at a time performing complex interactions. We set $M_{\max} = 7$ so initialisation samples were always generated, and tracked the sequence using $N = 10000$ particles. Representative still frames are shown in figure 4. Tracking was successful throughout except when two people crossed in front of a third, at which point the labels assigned to the front two people were transposed, though at all times the filter reported the presence of three objects. Figure 8 shows a representation of the log-likelihood ratios y_g^l for an image during the cross-over. It is clear that from this information alone it is not possible to reliably determine the locations of the three people, and correct tracking with this appearance model could only be ensured by enforcing a very strict dynamical model which would reduce robustness in other cases. If a separate foreground model were learned for each object then it might be possible to disambiguate the three people and track through such sequences. An MPEG movie of the entire tracked sequence is included in the electronic paper submission. Table 1 shows the time taken by the Alpha workstation to compute the y_g^l for a single frame, and also to render a single object for likelihood evaluation. Together with the overhead from the rest of the algorithm, the sequence is tracked with a peak-load frame rate of 4 frames/second, though this increases to 40 frames/second when the scene is empty.

We have performed real-time experiments using a 447 MHz Pentium II workstation with QCIF image resolution (160×120 pixels, $G = 31 \times 23 = 713$) and 15 frames/second. Setting $M_{\max} = 1$ and $N = 500$ one person can be reliably tracked using 33% of the CPU. By increasing N to 1000 particles and setting $M_{\max} = 7$ at least two people can be successfully tracked using about 85% of the CPU, though severe occlusions often cause object labels to be incorrectly reassigned or objects to disappear and reinitialise with a new label. Table 1 shows the computation speed of the filter operations for comparison with the Alpha workstation.

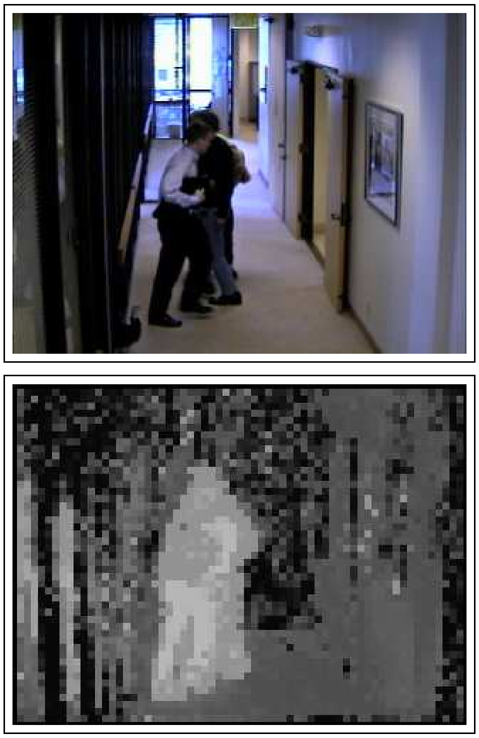

Figure 8: **The foreground model cannot distinguish between three people.** *Two people cross in front of a third who is almost entirely obscured. A single pooled model is used for all foreground objects, so it is impossible to distinguish between different people purely on the basis of the log-likelihood ratios y_g^l shown here. Separate object models learned online might help to disambiguate between people.*

CPU	Image	fixed (msec)	object (μsec)
667MHz Alpha	CIF	13	5
447MHz PII	CIF	42	27
447MHz PII	QCIF	10	15

Table 1: **Filter execution speed.** *Each time-step the algorithm takes a fixed time to compute all the y_g^l (shown in column 3) and a time to evaluate the object likelihoods which depends on the number of particles and the number of objects per particle. Column 4 shows τ_{obj}, the time to evaluate one object, so the total evaluation time for the particle set at time t is $\sum_n m_t^n \tau_{obj}$. Note that the smaller QCIF image requires approximately one quarter as many instructions to compute the y_g^l and that a smaller image also reduces the number of responses covered by an object and hence the time to compute an object likelihood.*

symbol	meaning	value
λ_r	object survival probability	0.99
λ_i	new object arrival probability	0.02
λ_d	object display threshold	0.8
δ_e	minimum physical separation between distinct objects (m)	0.5
δ_B	background likelihood additional covariance factor (grey-levels^2)	100
τ_B	background likelihood cutoff (grey-levels^{-6})	2.0×10^{-14}
τ_F	foreground likelihood cutoff (grey-levels^{-6})	3.0×10^{-13}
b_X	translation process noise (m)	0.11

	w_f	w_w	w_s	w_h	h	θ	α_w	α_s
mean μ_i	0.20m	0.22m	0.25m	0.08m	1.80m	0.75	0.60	0.83
steady-state standard deviation σ_i	0.03m	0.04m	0.04m	0.02m	0.05m	0.25	0.02	0.02
process noise ρ_i	0.003m	0.002m	0.002m	0.002m	0.003m	0.05	0.001	0.001

Figure 9: **Parameter values used for experiments.**

5 Conclusion

This paper presents two innovations. The first is an observation likelihood for an entire image generated by a known background occluded by an arbitrary number of foreground objects (which may in turn occlude each other). This likelihood is based on two sets of learned statistics: 1) for every location on a fixed grid in the image an independent model of the background near that location, and 2) for each foreground object, an appearance model pooled over the entire object. The second innovation is a particle filtering implementation of a Bayesian multiple-object tracker for which the number of objects is unknown and time varying.

We show that the filter implementation can be made efficient enough to allow robust tracking of a small number of objects in real time using a fraction of the processing power of a modern workstation. The Bayesian blob-tracker can therefore be considered a practical alternative to traditional blob trackers. Although only the single camera approach was demonstrated in this paper, it would be particularly appealing in the case of stereo tracking since it is straightforward to adapt a particle filter to make joint inferences from the inputs of multiple cameras [9].

The multiple-object filter is of general utility and could be applied in domains other than that of visual tracking. We have presented a simple particle filter implementation, but many variants can be constructed using the same basic model. There is a growing literature on particle filtering methods (see [2] for a survey), and in particular it may be possible to use importance sampling [8] to aid initialisation and techniques such as layered and partitioned sampling [13] to improve efficiency.

One common failure mode of the filter is that it becomes confused when one object passes in front of another, and switches the labels assigning identities to the objects. It should be possible to prevent these labelling failures by using separate foreground models for each object rather than the single default model we adopted in section 4. Our early attempts to use adaptive foreground models, however, have suffered from the traditional problems associated with adaptive templates, especially a tendency to grow to include patches of background. Further research is required to design distinct foreground models which do not suffer from this drawback. For really challenging image sequences it may be necessary to design more sophisticated foreground appearance models which track the shape of the objects more accurately, and perhaps use explicit edge information.

In summary, this paper advances the field of robust tracking of simple objects in two significant ways. First it describes a principled statistical treatment of the blob tracking problem which can be implemented in a reasonable computational budget. Second it introduces a Bayesian filter for an unknown, time-varying number of objects which is applicable to a wide variety of applications, including the most common types of surveillance tasks.

References

[1] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. In *Proc. Conf. Computer Vision and Pattern Recognition*, volume 2, pages 142–149, 2000.

[2] A. Doucet, N. de Freitas, and N. Gordon, editors. *Sequential Monte Carlo Methods in Practice*. Springer-Verlag, 2001.

[3] W.E.L. Grimson, C. Stauffer, R. Romano, and L. Lee. Using adaptive tracking to classify and monitor activities in a site. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 22–29, 1998.

[4] G. Hager and K. Toyama. The XVision system: A general-purpose substrate for portable real-time vision applications. *Computer Vision and Image Understanding*, 69(1):23–37, 1998.

[5] I. Haritaoglu, D. Harwood, and L. Davis. W^4S: A real-time system for detecting and tracking people in 2.5D. In *Proc. 5th European*

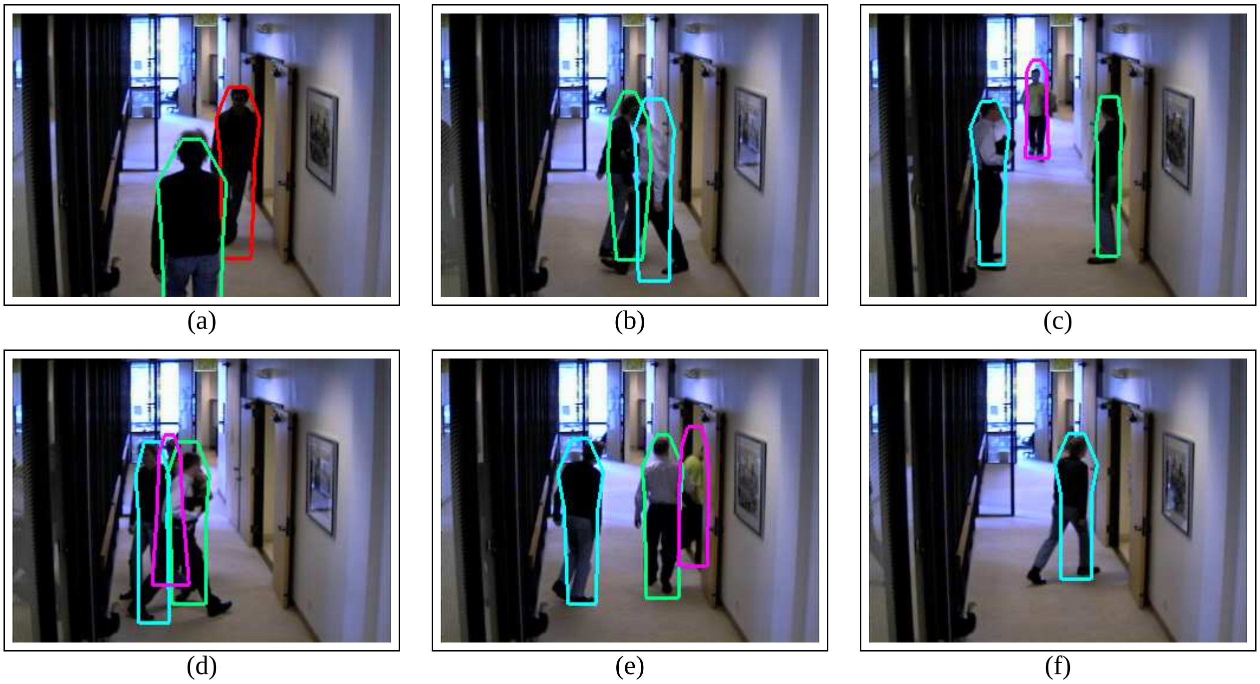

Figure 10: **Tracking results.** *(a) Person A and person B have entered the scene. (b) BraMBLe correctly infers that person B has exited towards the camera while C has entered from the side door and is about to walk in front of A. (c) BraMBLe tracks A and C as they swap positions, while D approaches from end of corridor. (d) A and C swap positions while conversing with D; note that BraMBLe incorrectly swaps the identity of A and C when all three people occupy the same image region. (e) BraMBLe correctly infers that D has moved behind C. (f) C and D have exited.*

Conf. Computer Vision, volume 1, pages 877–892, Freiburg, Germany, June 1998. Springer Verlag.

[6] S. Intille, J. Davis, and A. Bobick. Real-time closed-world tracking. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 697–703, 1997.

[7] M. Isard and A. Blake. Condensation —conditional density propagation for visual tracking. *Int. J. Computer Vision*, 28(1):5–28, 1998.

[8] M.A. Isard and A. Blake. ICondensation: Unifying low-level and high-level tracking in a stochastic framework. In *Proc. 5th European Conf. Computer Vision*, pages 893–908, 1998.

[9] Deutscher J., Blake A., and Reid I. Articulated body motion capture by annealed particle filtering. In *Proc. Conf. Computer Vision and Pattern Recognition*, volume 2, pages 126–133, 2000.

[10] N. Jojic, M. Turk, and T. Huang. Tracking self-occluding articulated objects in dense disparity maps. In *Proc. 7th Int. Conf. on Computer Vision*, volume 1, pages 123–130, 1996.

[11] 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*, pages 151–156, 1996.

[12] H. Lutkepohl. *Introduction to Multiple Time Series Analysis*. Springer-Verlag, 2nd edition, 1993.

[13] J. MacCormick and A. Blake. A probabilistic exclusion principle for tracking multiple objects. In *Proc. 7th Int. Conf. on Computer Vision*, pages 572–578, 1999.

[14] M.I. Miller, A. Srivastava, and U. Grenander. Conditional-mean estimation via jump-diffusion processes in multiple target tracking/recognition. *IEEE Transactions on Signal Processing*, 43(11):2678–2690, 1995.

[15] C. Rasmussen and G. Hager. Joint probabilistic techniques for tracking multi-part objects. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 16–21, 1998.

[16] M.J. Sidenbladh H., Black and D.J. Fleet. Stochastic tracking of 3D human figures using 2D image motion. In *Proc. 6th European Conf. Computer Vision*, volume 2, pages 702–718, Dublin, Ireland, June/July 2000. Springer Verlag.

[17] J. Sullivan, A. Blake, M. Isard, and J. MacCormick. Object localization by bayesian correlation. In *Proc. 7th Int. Conf. on Computer Vision*, volume 2, pages 1068–1075, 1999.

[18] H. Tao, H. Sawhney, and R. Kumar. A sampling algorithm for tracking multiple objects. In *Proc. ICCV Workshop on Vision Algorithms*, 1999.

[19] 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*, pages 51–56, 1996.