Sequential Monte Carlo Tracking by Fusing Multiple Cues in Video Sequences

Paul Brasnett [a], Lyudmila Mihaylova [*,b], David Bull [a], Nishan Canagarajah [a]

[a] *Department of Electrical and Electronic Engineering, University of Bristol, Bristol BS8 1UB, UK*

[b] *Department of Communication Systems, Lancaster University, Lancaster LA1 4WA, UK*

Abstract

This paper presents visual cues for object tracking in video sequences using particle filtering. A consistent histogram-based framework is developed for the analysis of colour, edge and texture cues. The visual models for the cues are learnt from the first frame and the tracking can be carried out using one or more of the cues. A method for online estimation of the noise parameters of the visual models is presented along with a method for adaptively weighting the cues when multiple models are used. A particle filter (PF) is designed for object tracking based on multiple cues with adaptive parameters. Its performance is investigated and evaluated with synthetic and natural sequences and compared with the mean-shift tracker. We show that tracking with multiple weighted cues provides more reliable performance than single cue tracking.

Keywords – particle filtering, tracking in video sequences, colour, texture, edges, multiple cues, Bhattacharyya distance

1 Introduction

Object tracking is required in many vision applications such as human-computer interfaces, video communication/compression, road traffic control, security and surveillance systems. Often the goal is to obtain a record of the trajectory of one or more targets over time and space. Object tracking in video sequences is a challenging task because of the large amount of data used and the common requirement for real-time computation. Moreover, most of the models encountered in visual tracking are *nonlinear*, *non-Gaussian*, *multi-modal* or any combination of these.

In this paper we focus on Monte Carlo methods (particle filtering) for tracking in video sequences. Particle filtering, also known as the Condensation algorithm [1] and bootstrap filter [2], has recently been proven to be a powerful and reliable tool for nonlinear systems [2–4]. Particle filtering is a promising technique because of its inherent property to allow fusion of different sensor data, to account for different uncertainties, to cope with data association problems when multiple targets are tracked with multiple sensors and to incorporate constraints. They keep track of the state through sample-based representation of probability density functions. Here we develop a particle filtering technique for object tracking in video sequences by visual cues. Further, methods are presented for combining the cues if they are assumed to be independent. By comparing results from single-cue tracking with multiple cues we show that multiple complementary cues can improve the accuracy of tracking. The features and their parameters are adaptively chosen based on appropriately defined distance function. The developed particle filter together with a mixed dynamic model enables recovery after a partial or full loss.

Different algorithms have been proposed for visual tracking and their particularities are mainly application dependent. Many of them rely on a single cue, which can be chosen according to the application context, e.g. in [5] a colour-based particle filter is developed. The colour-based particle filter has been shown [5] to outperform the mean-shift tracker proposed in [6, 7] in terms of reliability, at the price of increased computational time. However, both the particle filtering and mean shift tracking methods have real-time capabilities. Colour cues form a significant part of many tracking algorithms [5, 8–12], the advantage of colour is that it is a weak model and is therefore unrestrictive about the type of objects being tracked. The main problem for tracking with colour alone occurs when the region around the target object

[*] Corresponding author
Email addresses: paul.brasnett@bristol.ac.uk (Paul Brasnett), mila.mihaylova@lancaster.ac.uk (Lyudmila Mihaylova).

contains objects with similar colour. When the region is cluttered in this way a single cue does not provide reliable performance because it fails to fully model the target. Stronger models have been used but they rely on off-line learning and modelling of foreground and background models [1,13]. Multiple-cue tracking provides more information about the object and hence there is less opportunity for clutter to influence the result. In [9] colour cues are combined with motion and sound cues to provide better results. Motion and sound are both intermittent cues and therefore cannot always be relied upon. Colour and shape cues are used in [8], where shape is described using a parameterised rectangle or ellipse. The cues are combined by weighting each one based upon the performance in previous frames. A cue-selection approach to optimise the use of the cues is proposed in [14] which is embedded in a hierarchical vision-based tracking algorithm. When the target is lost, layers cooperate to perform a rapid search for the target and continue tracking. Another approach, called democratic integration [15] implements cues concurrently where all vision cues are complementary and contribute simultaneously to the overall result. The integration is performed through saliency maps and the result is a weighted average of saliency maps. Robustness and generality are major features of this approach resulting from the combination of the cues. Adaption schemes are sometimes used to handle changes to the appearance of the object being tracked [16]. Careful design has to make a trade off between adaption to rapidly changing appearance with adapting too quickly to incorrect regions. The work here does not involve an adaption scheme but an existing scheme (e.g. [16]) could be adopted within the framework. In this paper we also show that tracking with multiple weighted cues provides more reliable and accurate results. A framework is suggested for combining colour, texture and edge cues to provide robust and accurate tracking without the need for extensive off-line modelling. It is an extension and generalisation of the results reported in [17], with colour and texture only. A comparison is presented in [17] of a particle filter with a Gaussian sum particle filter, working separately with colour, with texture, and both with colour and texture features. Adaptive colour and texture segmentation for tracking moving objects is proposed in [11]. Texture is modelled by an autobinomial Gibbs Markov random field, whilst colour is modelled by a two-dimensional Gaussian distribution. In our paper texture is represented using a steerable pyramid decomposition which is a different approach to [11] but is related to the work in [16]. Additionally, in [11] segmentation-based tracking with a Kalman filter is considered instead of feature-based tracking proposed here.

The paper is organised as follows. Section 2 states the problem of visual tracking within a sequential Monte Carlo framework and presents the particle filter (PF) based on single or multiple information cues. Section 3 introduces the model of the region of interest. Section 4 describes the cues and their likelihoods used for the tracking process. Methods for adaptively changing the cues' noise parameters and adaptively weighting the cues are presented. The overall likelihood function of the particle filter represents a product of the separate cues. Section 5 investigates the particle filter performance and validates it over different scenarios. We show the advantages of fusing multiple cues compared to single-cue tracking using synthetic and natural video sequences. A comparison with the mean-shift algorithm is performed over natural video sequences and their computational time is characterised. Results are also presented for partial and full occlusions. Finally, Section 6 discusses the results and open issues for future research.

2 Sequential Monte Carlo Framework

The aim of sequential Monte Carlo estimation is to evaluate the *posterior* probability density function (pdf) $p(\boldsymbol{x}_k|\boldsymbol{Z}^k)$ of the state vector $\boldsymbol{x}_k \in \mathbb{R}^{n_x}$, with dimension n_x, given a set $\boldsymbol{Z}^k = \{\boldsymbol{z}^1, \ldots, \boldsymbol{z}^k\}$ of sensor measurements up to time k. The Monte Carlo approach relies on a sample-based construction to represent the state pdf. Multiple particles (samples) of the state are generated, each one associated with a weight $W_k^{(\ell)}$ which characterises the quality of a specific particle ℓ, $\ell = 1, 2, \ldots, N$.

An estimate of the variable of interest is obtained by the weighted sum of particles. Two major stages can be distinguished : *prediction* and *update*. During prediction, each particle is modified according to the state model of the region of interest in the video frame, including the addition of random noise in order to simulate the effect of the noise on the state. In the update stage, each particle's weight is re-evaluated based on the new data.

An inherent problem with particle filters is degeneracy, the case when only one particle has a significant weight. An estimate of the measure of degeneracy [18] at time k is given as

$$N_{eff} = \frac{1}{\sum_{\ell=1}^{N}(W_k^{(\ell)})}. \tag{1}$$

If the value of N_{eff} is below a user defined threshold N_{thres} a *resampling* procedure can help to avoid degeneracy by eliminating particles with small weights and replicating particles with larger weights.

2.1 A Particle Filter for Object Tracking Using Multiple Cues

Within the Bayesian framework, the conditional pdf $p(\boldsymbol{x}_{k+1}|\boldsymbol{Z}^k)$ is recursively updated according to the *prediction* step

$$p(\boldsymbol{x}_{k+1}|\boldsymbol{Z}^k) = \int_{\mathbb{R}^{n_x}} p(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k)p(\boldsymbol{x}_k|\boldsymbol{Z}^k)d\boldsymbol{x}_k \tag{2}$$

and the *update* step

$$p(\boldsymbol{x}_{k+1}|\boldsymbol{Z}^{k+1}) = \frac{p(\boldsymbol{z}_{k+1}|\boldsymbol{x}_{k+1})p(\boldsymbol{x}_{k+1}|\boldsymbol{Z}^{k})}{p(\boldsymbol{z}_{k+1}|\boldsymbol{Z}^{k})} \tag{3}$$

where $p(\boldsymbol{z}_{k+1}|\boldsymbol{Z}^{k})$ is a normalising constant. The recursive update of $p(\boldsymbol{x}_{k+1}|\boldsymbol{Z}^{k+1})$ is proportional to

$$p(\boldsymbol{x}_{k+1}|\boldsymbol{Z}^{k+1}) \propto p(\boldsymbol{z}_{k+1}|\boldsymbol{x}_{k+1})p(\boldsymbol{x}_{k+1}|\boldsymbol{Z}^{k}). \tag{4}$$

Table 1: The Particle Filter with Multiple Cues

Initialisation

(1) For $\ell = 1, \ldots, N$, generate samples $\{\boldsymbol{x}_0^{(\ell)}\}$ from the prior distribution $p(\boldsymbol{x}_0)$. Set initial weights $W_0^{(\ell)} = 1/N$.

For $k = 0, 1, 2, \ldots,$

Prediction Step

(2) For $\ell = 1, \ldots, N$, sample $\boldsymbol{x}_{k+1}^{(\ell)} \sim p(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k^{(\ell)})$ from the dynamic model presented in Section 3.

Measurement Update: evaluate the importance weights

(3) Compute the weights

$$W_{k+1}^{(\ell)} \propto W_k^{(\ell)} \mathcal{L}(\boldsymbol{z}_{k+1}|\boldsymbol{x}_{k+1}^{(\ell)}).$$

based on the likelihood $\mathcal{L}(\boldsymbol{z}_{k+1}|\boldsymbol{x}_{k+1}^{(\ell)})$ given in Section 4.

(4) Normalise the weights, $\widehat{W}_{k+1}^{(\ell)} = \frac{W_{k+1}^{(\ell)}}{\sum_{\ell=1}^{N} W_{k+1}^{(\ell)}}$.

Output

(5) The state estimate $\hat{\boldsymbol{x}}_{k+1}$ is the probabilistically averaged sum

$$\hat{\boldsymbol{x}}_{k+1} = \sum_{\ell=1}^{N} \widehat{W}_{k+1}^{(\ell)} \boldsymbol{x}_{k+1}^{(\ell)}.$$

(6) Estimate the effective number of particles N_{eff}

$$N_{eff} = \frac{1}{\sum_{\ell=1}^{N} (\widehat{W}_{k+1}^{(\ell)})^2}$$

If $N_{eff} \leq N_{thres}$ then perform resampling

Resampling Step

(7) Multiply/suppress samples $\boldsymbol{x}_{k+1}^{(\ell)}$ with high/ low importance weights $\widehat{W}_{k+1}^{(\ell)}$.

(8) For $\ell = 1, \ldots, N$, set $W_{k+1}^{(\ell)} = \widehat{W}_{k+1}^{(\ell)} = 1/N$.

Usually, there is no simple analytical expression for propagating $p(\boldsymbol{x}_{k+1}|\boldsymbol{Z}^{k+1})$ through (4) so numerical methods are used.

In the particle filter approach, a set of N weighted particles, drawn from the posterior conditional pdf, is used to map integrals to discrete sums. The posterior $p(\boldsymbol{x}_{k+1}|\boldsymbol{Z}^{k+1})$ is approximated by

$$\hat{p}(\boldsymbol{x}_{k+1}|\boldsymbol{Z}^{k+1}) \approx \sum_{\ell=1}^{N} \widehat{W}_{k+1}^{(\ell)} \delta(\boldsymbol{x}_{k+1} - \boldsymbol{x}_{k+1}^{(\ell)}) \tag{5}$$

where $\widehat{W}_k^{(\ell)}$ are the normalised importance weights. New weights are calculated, putting more weight on particles that are important according to the *posterior* pdf (5).

It is often impossible to sample directly from the posterior density function $p(\boldsymbol{x}_{k+1}|\boldsymbol{Z}^{k+1})$. This difficulty is circumvented by making use of the importance sampling from a known *proposal distribution* $p(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k)$. The particle filter is given in Table 1. The residual resampling algorithm described in [19, 20] is applied at step (7). This is a two step process making use of the sampling-importance-resampling scheme.

3 Dynamic Models

The considered model for the moving object provides invariance to different motions, such as translations, rotations, and to changes in the object size. This allows to cover the different types of motion of the object, also the case when the object size varies considerably (the object get closer to the camera or moves far away from it) and hence ensures reliable performance of the PF. In our particular implementation two generic models are used. We adopted a constant velocity model for the translational motion and the random walk model for the rotation and scaling. A mixed dynamic motion model is also presented which allows more than one model to be used for dealing with occlusions.

For the purpose of tracking an object in video sequences we initially choose a region which defines the object. The shape of this region is fixed *a priori* and here is a rectangular box.

Denote by (x, y) the coordinates of the centre of the rectangular region, by θ the angle through which the region is rotated, and by s the scale, $(\dot{x}, \dot{y})$ are the respective velocity components.

3.1 Constant Velocity Model for Translational Motion

The translational motion of the region of interest in x direction can be described by a constant velocity model [21]

$$\boldsymbol{x}_{k+1} = \boldsymbol{F}\boldsymbol{x}_k + \boldsymbol{w}_k, \quad \boldsymbol{w}_k \backsim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}), \tag{6}$$

where the state vector is $\boldsymbol{x} = (x, \dot{x})^T$. The matrix $\boldsymbol{F}$

$$\boldsymbol{F} = \begin{pmatrix} 1 & T \\ 0 & 1 \end{pmatrix},$$

describes the dynamics of the state over time and T is the sampling interval. The system noise $\boldsymbol{w}_k$ is assumed to be a zero-mean white Gaussian sequence, $\boldsymbol{w}_k \backsim \mathcal{N}(0, \boldsymbol{Q})$, with the covariance matrix

$$\boldsymbol{Q} = \begin{pmatrix} \frac{1}{4}T^4 & \frac{1}{2}T^3 \\ \frac{1}{2}T^3 & T^2 \end{pmatrix} \sigma^2 \tag{7}$$

and σ is the noise standard deviation.

3.2 *Random Walk Model for Rotational Motion and for the Scale*

A random walk model propagates the state $\boldsymbol{x} = (\theta, s)^T$ by

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{w}_k, \tag{8}$$

where $\boldsymbol{w}_k \backsim \mathcal{N}(0, \boldsymbol{Q})$ is a zero-mean Gaussian noise, with covariance matrix $\boldsymbol{Q} = \mathrm{diag}\{\sigma_\theta^2, \sigma_s^2\}$, describing the uncertainty in the state vector.

3.3 *Multi-Component State*

The motion of the object being tracked is described using the translation (x, y), rotation (θ) and scaling (s) components. The translation components are modelled using the constant velocity model (6) and the rotation and scaling components are modelled using the random walk model (8). The full augmented state of the region is then given as

$$\boldsymbol{x} = (x, \dot{x}, y, \dot{y}, \theta, s)^T. \tag{9}$$

The dynamics of the full state can then be modelled as

$$\boldsymbol{x}_{k+1} = \boldsymbol{G}\boldsymbol{x}_k + \boldsymbol{w}_k, \quad \boldsymbol{w}_k \backsim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}), \tag{10}$$

where the matrix $\boldsymbol{G}$ has the form

$$\boldsymbol{G} = \begin{pmatrix} \boldsymbol{F} & \boldsymbol{0}_{2\times2} & \boldsymbol{0}_{2\times1} & \boldsymbol{0}_{2\times1} \\ \boldsymbol{0}_{2\times2} & \boldsymbol{F} & \boldsymbol{0}_{2\times1} & \boldsymbol{0}_{2\times1} \\ \boldsymbol{0}_{1\times2} & \boldsymbol{0}_{1\times2} & 1 & 0 \\ \boldsymbol{0}_{1\times2} & \boldsymbol{0}_{1\times2} & 0 & 1 \end{pmatrix}. \tag{11}$$

The covariance matrix of the zero-mean Gaussian is

$$\boldsymbol{Q} = \begin{pmatrix} \boldsymbol{Q}_x & \boldsymbol{0}_{2\times2} & \boldsymbol{0}_{2\times1} & \boldsymbol{0}_{2\times1} \\ \boldsymbol{0}_{2\times2} & \boldsymbol{Q}_y & \boldsymbol{0}_{2\times1} & \boldsymbol{0}_{2\times1} \\ \boldsymbol{0}_{1\times2} & \boldsymbol{0}_{1\times2} & \sigma_\theta^2 & 0 \\ \boldsymbol{0}_{1\times2} & \boldsymbol{0}_{1\times2} & 0 & \sigma_s^2 \end{pmatrix}, \tag{12}$$

where $\boldsymbol{Q}_x$ is the covariance matrix of the constant-velocity model for the x component (7), $\boldsymbol{Q}_y$ is the covariance matrix of the y component and σ_θ^2 and σ_s^2 are the covariances for the rotation (θ) and scaling (s).

3.4 *Mixed Dynamic Model*

A mixed-dynamic model allows the system to be described through more than one dynamic model [16,22,23] and provides abilities to the tracking algorithm to cope with occlusions. Here, we make use of two models: the constant velocity model as given in Section 3.1 and the other is the reinitialisation model described below. When the object is occluded the tracker might lose it temporarily. In this case the reininialisation model that ensures uniform spread of particles along the image guarantees robustness. The new location of the object is recovered after processing the information from separate cues in the measurement update step.

Samples are generated from the constant velocity model with probability j, set *a priori*, and from the reinitialisation model with probability $1-j$. Hence, from the mixed model samples are generated by the following steps:

(1) Generate a number $\gamma \in [0, 1)$ from a uniform distribution $\mathcal{U}$
(2) If $\gamma > j$, then sample from the constant velocity model (6)
(3) else use the reinitialisation model

$$p(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k) \sim \mathcal{U}(\boldsymbol{0}, \boldsymbol{x}_{\max}). \tag{13}$$

where $\boldsymbol{x}_{\max}$ is a vector with the maximum allowed values for the state vector components. This is repeated until the required number of samples are obtained.

4 Likelihood Models

This section describes how we model the separate cues of the rectangular region $\mathcal{S}_{\boldsymbol{x}}$, surrounding the moving object and the likelihood models of the cues. One of the particularities of tracking moving objects in video sequences compared to other tracking problems, such as tracking of airborne targets with radar data, is that there are no measurement models in explicit form. The estimated state variables of the object are connected to features of the video sequences. Practically, the likelihood models of the features provide information about the changes in the motion of the object.

All of the models are based on histograms. Histograms have the useful property that they allow some change in the object appearance without changing the histogram.

4.1 Colour Cue

Colour cues are flexible in the type of object that they can be used to track. However, the main drawbacks of colour cues are:

- the effect of other similar coloured regions and
- the lack of discrimination with respect to rotation (obvious on Fig. 1).

A histogram, $\boldsymbol{h}_{\boldsymbol{x}} = (h_{1,\boldsymbol{x}}, \ldots, h_{B_C,\boldsymbol{x}})$, for a region $\mathcal{S}_{\boldsymbol{x}}$ corresponding to a state $\boldsymbol{x}$ is given by

$$h_{i,\boldsymbol{x}} = \sum_{\boldsymbol{u} \in \mathcal{S}_{\boldsymbol{x}}} \delta_i(b_{\boldsymbol{u}}), \quad i = 1 \ldots B_C \qquad (14)$$

where δ_i is the Kronecker-delta function at the bin index i, $b_{\boldsymbol{u}} \in \{1, \ldots, B_C\}$ is the histogram bin index associated with the intensity at pixel location $\boldsymbol{u} = (x, y)$ and B_C is the number of bins in each colour channel. The histogram is normalised such that $\sum_{i=1}^{B_C} h_{i,\boldsymbol{x}} = 1$.

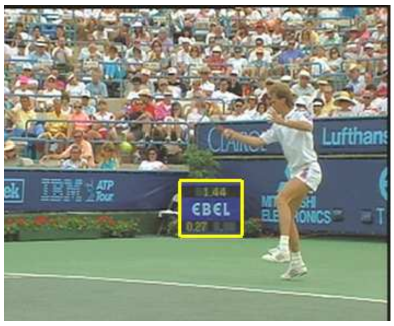
A histogram is constructed for each channel in the colour space. For example, we use 8x8x8 bin histograms in the three channels of red, green, blue (RGB) colour space [9], other colour spaces could be used to improve robustness to illumination or appearance changes.

4.2 Texture Cue

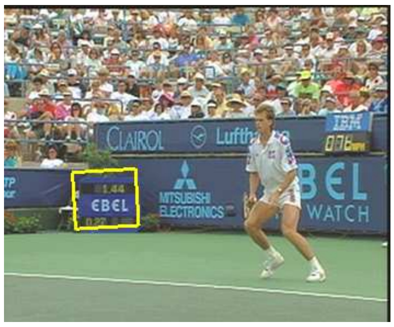
Although there is no unique definition of texture, it is generally agreed that texture describes the spatial arrangements of pixel levels in an image, which may be stochastic or periodic, or both [24]. Texture can be qualitatively characterised such as *fine*, *coarse*, *grained* and *smooth*. When a texture is viewed from a distance it may appear to be *fine*, however, when viewed from close up it may appear to be *coarse*.

The texture description used for this work is based on steerable pyramid decomposition [25]. The first derivative filter as developed in [26] is steered to 4 orientations at two scales (subsampled by a factor of two). A histogram is then constructed for each of the 8 bandpass filter outputs

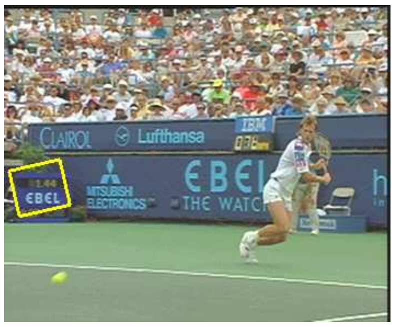
$$t_{i,\boldsymbol{x}} = \sum_{\boldsymbol{u} \in \mathcal{S}_{\boldsymbol{x}}} \delta_i(t_u), \quad i = 1, \ldots, B_T \qquad (15)$$



(a) Frame 1



(b) Frame 40



(c) Frame 70

Fig. 1. Colour cues provide a flexible model for tracking but lack discrimination with respect to rotation. The time board is tracked with colour cues, it can be seen that at frame 40 (b) the region has rotated, this has got worse by frame 70 (c).

where $t_u \in \{1, \ldots, B_T\}$ is the histogram bin index associated with the steerable filter output $\hat{\theta}$ at pixel location $\boldsymbol{u}$, with B_E number of bins. The histogram is normalised such that $\sum_{i=1}^{B_E} e_{i,\boldsymbol{x}} = 1$.

4.3 Edge Cue

Edge cues are useful for modelling the structure of the object to be tracked. The edges are described using a histogram based on the estimated edge direction. Given an image region $\mathcal{S}_{\boldsymbol{x}}$ the intensity of the pixels in that region are $\boldsymbol{I}(\mathcal{S}_{\boldsymbol{x}})$. The edge images are constructed by estimating the gradients $\frac{\partial \boldsymbol{I}}{\partial x}$ and $\frac{\partial \boldsymbol{I}}{\partial y}$ in the x and y directions respectively by Prewitt operators. The edge strength m

and direction θ are then approximated as

$$m(\boldsymbol{u}) = \sqrt{\frac{\partial \boldsymbol{I}}{\partial x} + \frac{\partial \boldsymbol{I}}{\partial y}}, \;\; \theta(\boldsymbol{u}) = tan^{-1}\left(\frac{\partial \boldsymbol{I}}{\partial y} \Big/ \frac{\partial \boldsymbol{I}}{\partial x}\right). \quad (16)$$

The edge direction is filtered to include only edges with magnitude above a predefined threshold

$$\hat{\theta}(\boldsymbol{u}) = \begin{cases} \theta(\boldsymbol{u}), & m(\boldsymbol{u}) > \text{threshold} \\ 0, & \text{otherwise.} \end{cases} \quad (17)$$

A histogram $e_{i,\boldsymbol{x}}$ of the edge directions $\hat{\theta}$ is then constructed

$$e_{i,\boldsymbol{x}} = \sum_{\boldsymbol{u} \in \mathcal{S}_{\boldsymbol{x}}} \delta_i(b_u), \quad i = 1, \ldots, B_E \quad (18)$$

where $b_u \in \{1, \ldots, B_E\}$ is the histogram bin index associated with the thresholded edge gradient $\hat{\theta}$ at pixel location $\boldsymbol{u}$, with B_E number of bins. The histogram is normalised such that $\sum_{i=1}^{B_E} e_{i,\boldsymbol{x}} = 1$.

4.4 Weighted Histograms

The above histograms discard all information about the spatial arrangement of the features in the image. An alternative approach that incorporates the pixel distribution can help to give better performance [5]. More specifically we can give greater weighting to pixels in the center of the image region. This weighting can be done through the use of a convex and monotonically decreasing kernel, for example the Epanechnikov kernel [5] or the elliptical Gaussian function

$$K(\boldsymbol{u}) = \frac{1}{2\pi\rho_x\rho_y} \exp\left(-\frac{(x-\hat{x})^2}{2\rho_x^2} + \frac{(y-\hat{y})^2}{2\rho_y^2}\right) \quad (19)$$

where the values ρ_x^2 and ρ_y^2 control the spatial significance of the weighting function in the x and y directions and the centre pixel in the target region is at $(\hat{x}, \hat{y})$. This kernel can be used to weight the pixel when extracting the histogram

$$h_{i,\boldsymbol{x}} = \sum_{\boldsymbol{u} \in \mathcal{S}_{\boldsymbol{x}}} K(\boldsymbol{u})\, \delta_i(b_{\boldsymbol{u}}), \quad i = 1 \ldots B_C \quad (20)$$

The histogram is normalised so that $\sum_{i=1}^{B_C} h_{i,\boldsymbol{x}} = 1$.

4.5 Distance Measure

The Bhattacharyya measure [27] has been used previously for colour cues [5, 7] because it has the important property that $\rho(p,p) = 1$. In the case here the distributions for each cue are represented by the respective histograms [28]

$$\rho(\boldsymbol{h}_{\text{ref}}, \boldsymbol{h}_{\text{tar}}) = \sum_{i=1}^{B} \sqrt{h_{\text{ref},i} h_{\text{tar},i}}, \quad (21)$$

where two normalised histograms $\boldsymbol{h}_{\text{tar}}$ and $\boldsymbol{h}_{\text{ref}}$ describe the cues for a *target region* defined in the current frame and a *reference region* in the first frame respectively. The measure of the similarity between these two distributions is then given by the Bhattacharyya distance

$$d(\boldsymbol{h}_{\text{ref}}, \boldsymbol{h}_{\text{tar}}) = \sqrt{1 - \rho(\boldsymbol{h}_{\text{ref}}, \boldsymbol{h}_{\text{tar}})}. \quad (22)$$

The larger the measure $\rho(\boldsymbol{h}_{\text{ref}}, \boldsymbol{h}_{\text{tar}})$ is, the more similar the distributions are. Conversely, for the distance d, the smaller the value the more similar the distributions (histograms) are. For two identical normalised histograms we obtain $d = 0$ ($\rho = 1$) indicating a perfect match.

Based on (22) a distance D^2 for colour can be defined that takes into account all of the colour channels

$$D^2(\boldsymbol{h}_{\text{ref}}, \boldsymbol{h}_{\text{tar}}) = \frac{1}{3} \sum_{c \in \{R,G,B\}} d^2(\boldsymbol{h}_{\text{ref}}^c, \boldsymbol{h}_{\text{tar}}^c) \quad (23)$$

the distance D^2 for the edges is equal to d^2 since there is only one component. The distance D^2 for texture is

$$D^2(\boldsymbol{h}_{\text{ref}}, \boldsymbol{h}_{\text{tar}}) = \frac{1}{8} \sum_{\omega \in \{1,\ldots,8\}} d^2(\boldsymbol{h}_{\text{ref}}^\omega, \boldsymbol{h}_{\text{tar}}^\omega) \quad (24)$$

where ω is the channel in the steerable-pyramid decomposition.

The *likelihood function* for the cues can be defined by [9]

$$\mathcal{L}(\boldsymbol{z}|\boldsymbol{x}) \propto \exp\left(-\frac{D^2(\boldsymbol{h}_{\text{ref}}, \boldsymbol{h}_{\boldsymbol{x}})}{2\sigma^2}\right) \quad (25)$$

where the standard deviation σ specifies the Gaussian noise in the measurements. Note that small Bhattacharyya distances correspond to large weights in the particle filter. The choice of an appropriate value for σ is usually left as a design parameter, a method for setting and adapting the value is proposed in Section 4.6.

4.6 Dynamic Parameter Setting

Figure 2 shows the likelihood surface for a single cue, applied to one frame, with different values of σ. It can be seen that as σ is varied the likelihood surface changes significantly. The likelihood becomes more discriminating as the value of σ is decreased. However, if σ is too small and there has been some change in the appearance of the object, due to noise, then the likelihood function may have all values near to or equal to zero. The value of the noise parameter σ has a major influence on the properties of the likelihood (25). Typically, the choice of this value is left as a design parameter to be determined usually by experimentation. For a well constrained problem, such as face tracking, analysis can be performed off-line to determine an appropriate value. However, if the algorithm is to be used to track *a priori* unknown objects, it may not be possible to determine one value for all objects. To overcome the problems of choosing an appropriate value for σ, an adaptive scheme is presented here which aims to maximise the information available in the likelihood, using the Bhattacharyya distance d. We define the minimum squared distance $D^2_{l,\min}$ as the minimum distance D^2 of the set of distance calculated for all particles with a particular cue l ($l = 1, 2, \dots, L$) where L is the number of cues. Rearranging the likelihood yields

$$log(\mathcal{L}) = -\frac{D^2}{2\sigma^2} \tag{26}$$

from which we can get

$$\sigma = \frac{\sqrt{2}}{2}\sqrt{-\frac{D^2_{l,\min}}{log(\mathcal{L})}}. \tag{27}$$

For example if the choice is made to maximise the information by setting σ to give a maximum likelihood $log(\mathcal{L}) = -1$ ($\mathcal{L} \approx 0.36$) then $\sigma = (\sqrt{2D^2_{l,\min}})/2$.

4.7 Multiple Cues

The relationship between different cues has been treated differently by different authors. For example, [29] makes the assumption that colour and texture are not independent. However, other works [11, 30] assume that colour and texture cues are independent. For the purposes of image classification the independence assumption between colour and texture is applied for feature fusion in a Bayesian framework [31].There is generally agreement that in practice colour and texture and colour and edges do combine well together.

We assume that the considered cue combinations, colour and texture and colour and edges are independent. With this assumption the overall likelihood function of the particle filter represents a product of the likelihoods of the separate cues

$$\mathcal{L}^{fused}(\boldsymbol{z}_k|\boldsymbol{x}_k) = \prod_{l=1}^{L} \mathcal{L}_l(\boldsymbol{z}_{l,k}|\boldsymbol{x}_k)^{\epsilon_l} \tag{28}$$

The cues are adaptively weighted by weighting coefficients ϵ_l, $\boldsymbol{z}_k$ denotes the measurement vector, composed of the measurement vectors $\boldsymbol{z}_{l,k}$ from the l^{th} cue for $l = 1, \dots, L$.

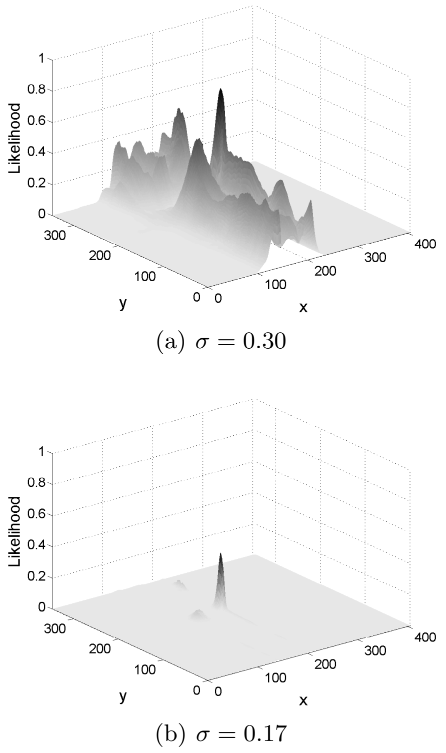


Fig. 2. The effect of the σ value on the cues. The results shown are for the colour cue, however, similar results apply for texture and edges. It can be seen that as σ decreases there is more discrimination in the likelihood. As σ becomes smaller the likelihood tends to zero. A method for dynamic setting of σ is presented in Section 4.6

4.8 Adaptively Weighted Cues

A method is presented here which takes account of the Bhattacharyya distance (22) to give some significance to the likelihood obtained for each cue based on the current frame. This is different to previous works which use the performance of the cues over the previous frames [9], not taking into account information from the latest measurements. This allows an estimate to be made for ϵ_l in (28). Using the smallest value of the distance measure $D^2_{l,\min}$ for each cue the weight for each cue l is determined by

$$\hat{\epsilon}_l = \frac{1}{D^2_{l,\min}}, \qquad l = 1, \dots, L. \tag{29}$$

The weights are then normalised such that $\sum_{l=1}^{L} \epsilon_l = 1$

$$\epsilon_l = \frac{\hat{\epsilon}_l}{\sum_{l=1}^{L} \hat{\epsilon}_l}, \qquad l = 1, \dots, L. \tag{30}$$

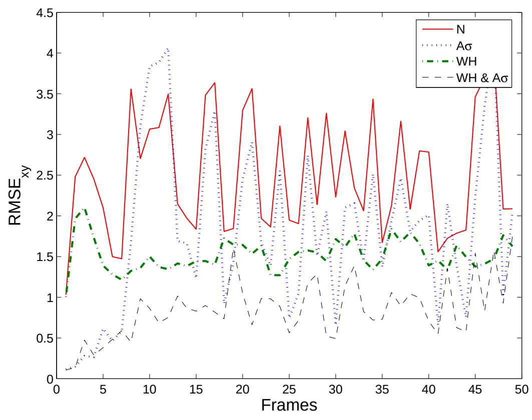


Fig. 3. Results from: 1) nonadaptive cues (N), 2) adaptive cues with automatic setting of σ (Aσ), 3) Gaussian weighting kernel for the histogram (WH), 4) both WH & Aσ.

5 Experimental Results

This section evaluates the performance of : *i*) the colour, texture and edge cues *ii*) combined cues and *iii*) the mixed-state model in sequences with occlusion.

The combined root mean squared error (RMSE) [32]

$$RMSE_{xy} = \sqrt{\frac{1}{R}\sum_{i=0}^{R}(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2} \qquad (31)$$

of the pixel coordinates (x_i, y_i) to their estimates $(\hat{x}_i, \hat{y}_i)$ is the measure used to evaluate the performance of the developed technique in each frame $i = 1, \ldots, N_f$ over $R = 100$ independent Monte Carlo realisations.

5.1 *Dynamic σ and Weighted Histograms*

Two techniques were given in this paper to improve the performance of the cues: *i*) automatic setting of the noise parameters σ for the likelihood (Section 4.6) and *ii*) weighting of the pixels in the histogram extraction process (Sections 4.1, 4.2 and 4.3). The effect of these techniques can be seen in Fig. 3, where the $RMSE_{xy}$ is shown for 100 realisations each with $N = 500$ particles, $N_{thresh} = N/2$ for tracking in a synthetic sequence using colour cues. Four different implementations are compared: 1) nonadaptive cues (N), 2) adaptive cues with automatic setting of σ (Aσ), 3) Gaussian weighting kernel for the histogram (WH), 4) and both WH & Aσ. The automatic setting of σ and the use of a Gaussian weighting kernel for the histogram both provide an improvement. The smallest error is seen when the Gaussian weighting kernel for the histogram is combined with automatic setting of σ (WH & Aσ).

5.1.1 *Single Cues*

Three very different tracking scenarios are used to highlight some of the strengths and weaknesses of the individual cues. All of the results are obtained using 500 particles. The first sequence is a wildlife problem which involves tracking a penguin [33] moving across a snowy background, Fig. 4. As is common in wildlife scenarios the colour of the object to be tracked is similar to the background. The particle filter with colour cues (Fig. 4 (a)-(c)) is distracted by similar coloured background regions. Both the particle filter with edge cues (Fig. 4 (d)-(f)) and with texture cues (Fig. 4 (g)-(i)) perform much better and track the penguin successfully.

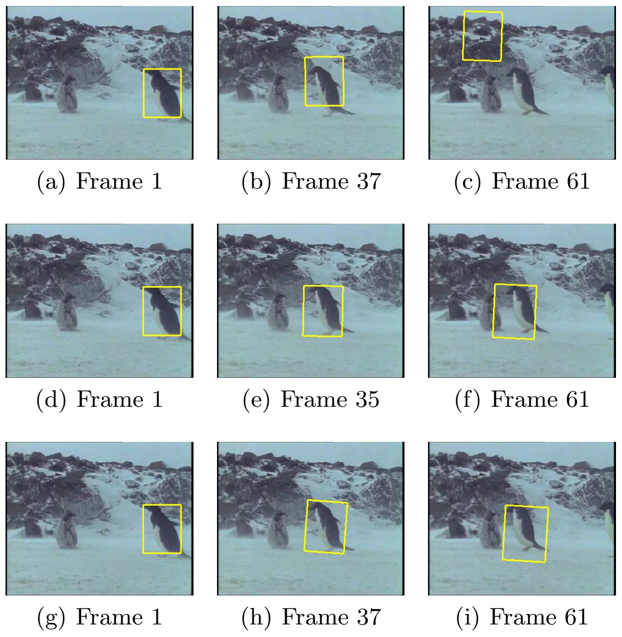
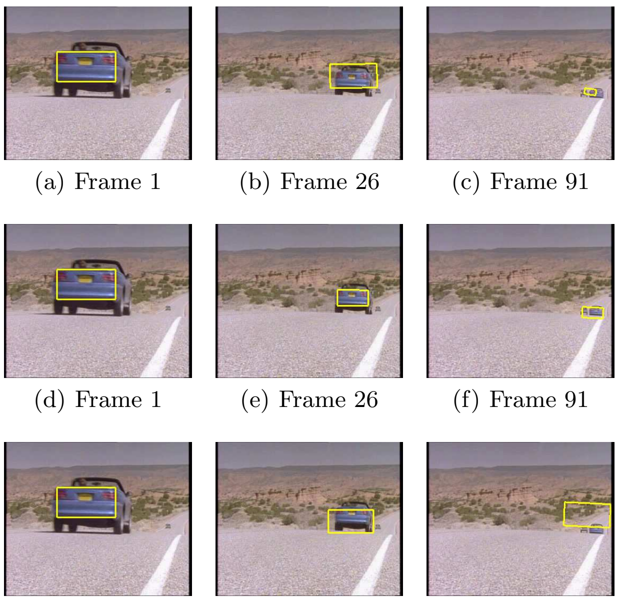


Fig. 4. Tracking a penguin against a snowy background. The colour cues (a)-(c) are distracted by the snowy background. Using either the edge (d)-(f) or the texture cues (g)-(i) provides an improvement.

The second sequence is a car tracking [33] problem with the car undergoing a significant and rapid change in scale. It can be seen that despite the change in scale both the particle filters with colour (Fig. 5 (a)-(c)) and edge cues (Fig. 5 (d)-(f)) are able to keep track of the full state of the car. However, the texture cues (Fig. 5 (g)-(i)) fail because of the change in appearance and distractions in the background.

The final example from single cues is an example of tracking a logo, in a longer sequence, that is undergoing translation, rotation and some small amount of scale change. All three cues are able to keep track of the location of the sequence but the particle filter with colour cues (Fig. 6 (a)-(c)) does not provide accurate state information for the rotation of the object. The particle fil-


(a) Frame 1 (b) Frame 26 (c) Frame 91

(d) Frame 1 (e) Frame 26 (f) Frame 91

(g) Frame 1 (h) Frame 26 (i) Frame 91

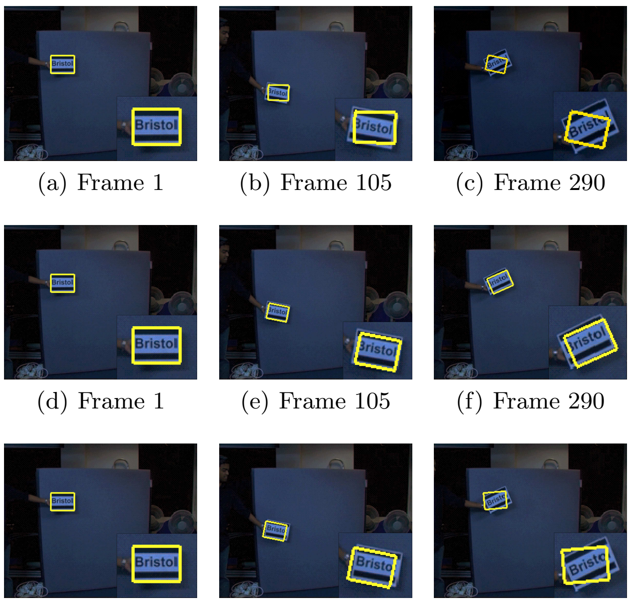
Fig. 5. Tracking a car, as it moves away it undergoes a significant change in scale. The colour (a)-(c) and edge (d)-(f) cues both cope with the scale change. After the original object has undergone some change the texture cues get distracted by the background.

ter with texture cues (Fig. 6 (d)-(f)) provides better information about the rotation of the object and the particle filter with edge cues (Fig. 6 (g)-(i)) provides the most accurate result of the three.

5.1.2 Comparison with Mean-Shift Tracker

An alternative tracking technique that has received a considerable amount of research interest recently is the mean-shift tracker [6,34]. A comparison between the visual particle filter presented here and the mean-shift tracker is particularly relevant because they are both based on histogram analysis. The mean-shift tracker is a mode-finding technique that locates the local minimum of the posterior density function. Based on the mean-shift vector, received as an estimation of the gradient of the Bhattacharyya function, the new object state estimate is calculated. The mean-shift algorithm was implemented with Epanechnikov kernel [6].

A comparison between the results of the particle filter and the mean-shift tracker can be seen in Figure 7, both algorithms use colour cues. The mean-shift tracker is unable to track successfully as the hand is moved in front of the face. The particle filter is slightly distracted by the face, however, it is successfully tracks the hand due to the fact that it can maintain a multi-modal distribution for a number of frames whereas the mean-shift tracker is not able to. This illustrates the superiority of the particle filter with respect to the mean-shift tracker in the presence of ambiguous situations.

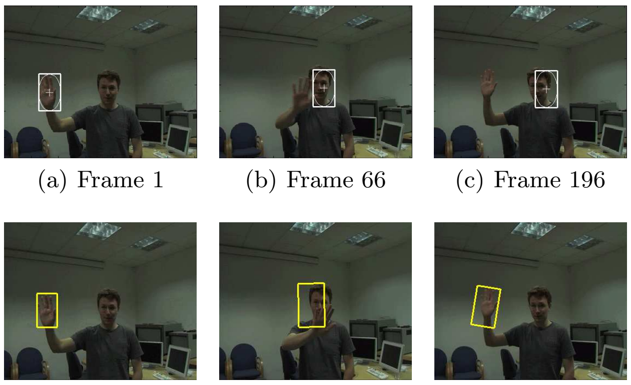


(a) Frame 1 (b) Frame 105 (c) Frame 290

(d) Frame 1 (e) Frame 105 (f) Frame 290

(g) Frame 1 (h) Frame 105 (i) Frame 290

Fig. 6. Logo tracking as it undergoes translation, rotation and mild scale change. The colour (a)-(c) cues are able to locate the object but it does not successfully capture the object rotation. The texture cues (g)-(i) provide more accurate information about the rotation and the edge cues (d)-(f) provide the most accurate information about the rotation.


(a) Frame 1 (b) Frame 66 (c) Frame 196

(d) Frame 1 (e) Frame 66 (f) Frame 196

Fig. 7. Tracking a hand using colour cues to compare the performance of the mean-shift tracker with the particle filter. The mean shift(a)-(c) tracker gets distracted by the face and does not recover. The particle filter (d)-(f) is distracted by the face but because it is able to maintain a multi-modal distribution it is able to recover.

The results presented here were obtained from Matlab implementation, where the particle filter takes in the order of 10 times longer than the mean shift tracker. The PF with colour cue was also implemented in C++ software on a standard PC computer (with Pentium CPU and 2.66 GHz) and has shown abilities to process 25-30 frames per second with particles in the range from 500 to 100. This result shows the applicability of the PF to real-time problems. The same algorithm, ran with the Matlab code needs 8 times more computational time than its C++ version.

5.1.3 Multiple Cues

From the results presented in the previous section it can be seen that no single cue can provide accurate results under all conditions. In this section we look at the performance change when combining the cues.

Firstly, the behaviour of the cue weighting scheme introduced in Section 4.7 is explored with an example as shown in Fig. 8. In (d) the change in weights from edge to colour can be seen. At the start of the sequence the edges provide more accurate results and is therefore given a higher weighting. As the players turn around the edges in the scene change and therefore the model learnt from the first frame become less reliable. In contrast the colour of the region does not change significantly and so becomes relatively more reliable and is given a higher weighting.

In the previous section the PF with the colour cue failed to track the penguin successfully (Fig. 4), we now look at how the performance of the PF is effected if the colour cues are combined with the edge and texture information. It can be seen in Fig. 9 that the previous performance of the edge and texture cues is maintained even when the colour cues are combined with them. In a hand tracking scenario the particle filter with edge cues (Fig. 10 (a)-(c)) fails but the particle filter with colour cues (Fig. 10 (d)-(f))is successful. This is due to the fact that the particle filter can maintain multi-modal posterior distributions. The particle filter with combined colour and edge cues (Fig. 10 (g)-(i)) successfully tracks the hand through the entire sequence.

5.2 Occlusion Handling and Handling the Changeable Window Size

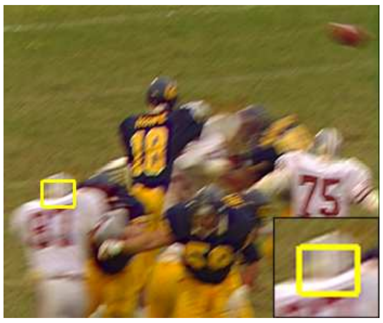
It is important that the tracking process is robust to both partial occlusions and is able to recover after a full occlusion. The sequence shown in Fig. 11 contains full occlusions from which the tracker successfully recovers. This is due to the mixed-state model described in Section 3.4 that provides the ability to recover when the object undergoes full occlusion or re-enters the frame after leaving. Additionally, the use of the Gaussian kernel enhances the accuracy when features are extracted from the frame.

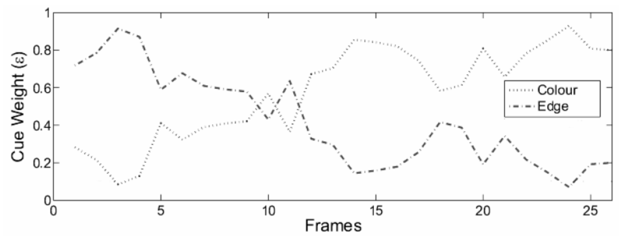
6 Conclusions and Future Work

This paper has presented a sequential Monte Carlo technique for object tracking in a broad range of video sequences with visual cues. The visual cues, colour, edge, and texture, form the likelihood of the developed particle filter. A method for automatic dynamic setting of the noise parameters of the cues is proposed to allow more flexibility in the object tracking. Multiple cues are combined for tracking, which has been shown to make the particle filter more able to accurately and robustly track a range of objects. These techniques can be further extended with other visual cues such as motion and

(a) Frame 1

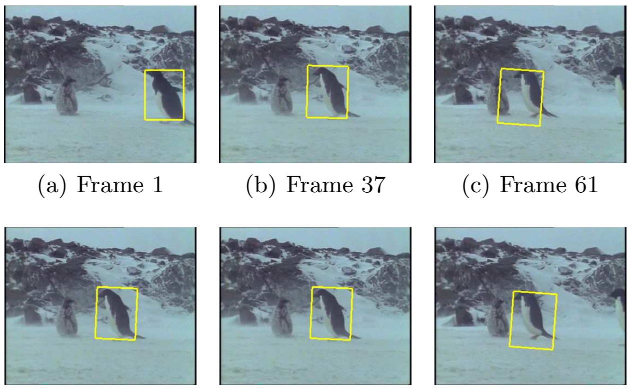
(b) Frame 16


(c) Frame 26



(d) Weights of each cue through the sequence

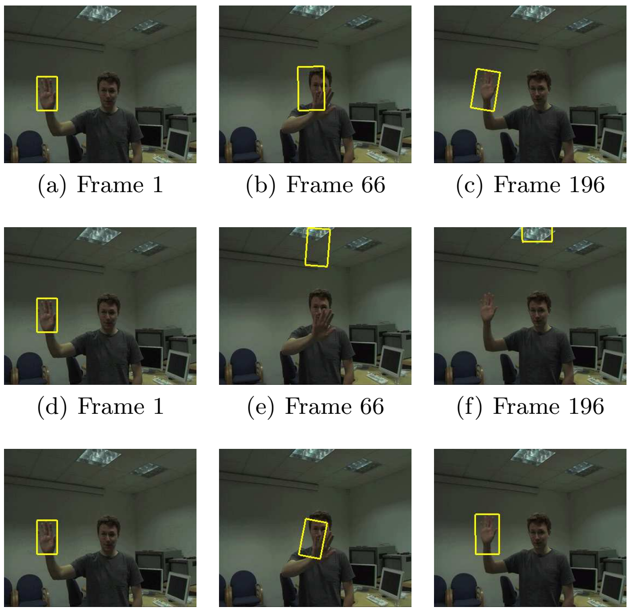
Fig. 8. The particle filter is run with adaptively weighted colour and edge cues to track the football player's helmet. (a)-(c) show that the helmet is successfully tracked. The weighting assigned to the cues is shown in (d).



(a) Frame 1 (b) Frame 37 (c) Frame 61

(d) Frame 1 (e) Frame 35 (f) Frame 61

Fig. 9. The same sequence as in Fig 4 for which colour tracking failed. (a)-(c) show the results from tracking with combined colour and edge cues, (d)-(f) from colour and texture cues. It can be seen that combining the colour cues with either edge and texture cues provides accurate tracking.



(a) Frame 1 (b) Frame 66 (c) Frame 196

(d) Frame 1 (e) Frame 66 (f) Frame 196

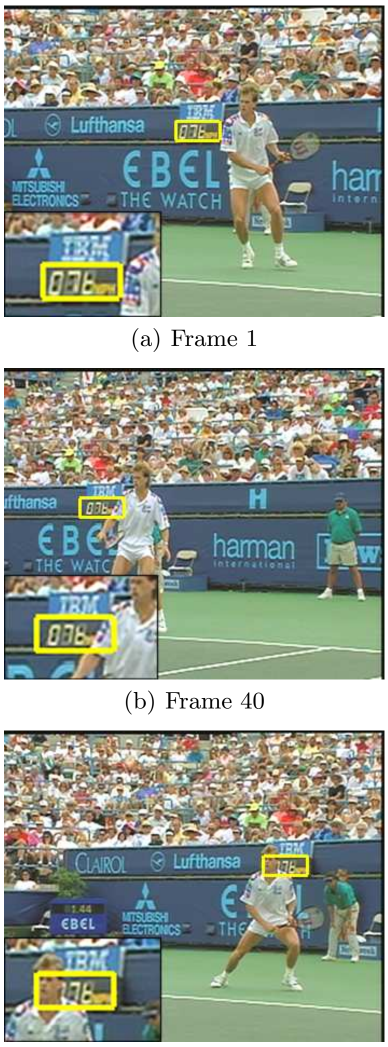
(g) Frame 1 (h) Frame 66 (i) Frame 196

Fig. 10. Hand tracking as it undergoes translation, rotation and mild scale change. The colour (a)-(c) cues track the object, although some distraction is caused by the face. The edge cues (d)-(f) get distracted by the edge information in the light. Combining colour and edges (g)-(i) provides more accurate tracking of the hand.

non-visual cues.

The developed particle filter is compared with the mean-shift algorithm and its reliability is shown also in the presence of ambiguous situations. Nevertheless that the particle filter is more time consuming than the mean-shift algorithm, it runs comfortably in real time.

Current and future areas for research include the investigation of alternative data fusion schemes, improved proposal distributions, tracking multiple objects and online adaption of the target model.



(a) Frame 1

(b) Frame 40

(c) Frame 168

Fig. 11. In this sequence the *serve speed* board is being tracked, it undergoes both partial and full occlusion. The results show that the cues are resilient to partial occlusion, see (b) and (c). Using the mixed-state model the tracker is able to recover following a full occlusion.

Acknowledgements

The authors are grateful to the financial support by the UK MOD Data and Information Fusion Defence Technology Centre for this work.

References

[1] M. Isard and A. Blake, “Condensation – conditional density propagation for visual tracking,” *International Journal of Computer Vision*, vol. 28, no. 1, pp. 5–28, 1998.

[2] N. Gordon, D. Salmond, and A. Smith, "A novel approach to nonlinear / non-Gaussian Bayesian state estimation," *IEE Proceedings-F*, vol. 140, pp. 107–113, April 1993.

[3] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statistics and Computing*, vol. 10, no. 3, pp. 197–208, 2000.

[4] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–188, 2002.

[5] K. Nummiaro, E. Koller-Meier, and L. V. Gool, "An adaptive color-based particle filter," *Image and Vision Computing*, vol. 21, no. 1, pp. 99–110, 2003.

[6] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," in *Proceedings of the 1st Conference on Computer Vision and Pattern Recognition.* Hilton Head, SC, 2000, pp. 142–149.

[7] ——, "Kernel-based object tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, pp. 564–577, 2003.

[8] C. Shen, A. van den Hengel, and A. Dick, "Probabilistic multiple cue integration for particle filter based tracking," in *Proc. of the VIIth Digital Image Computing: Techniques and Applications.* C. Sun, H. Talbot, S. Ourselin, T. Adriansen, Eds., 10-12 Dec. 2003.

[9] P. Pérez, J. Vermaak, and A. Blake, "Data fusion for tracking with particles," *Proceedings of the IEEE*, vol. 92, no. 3, pp. 495–513, March 2004.

[10] M. Spengler and B. Schiele, "Towards robust multi-cue integration for visual tracking," *Machine Vision and Applications*, vol. 14, no. 1, pp. 50–58, 2003.

[11] E. Ozyildiz, N. Krahnstöver, and R. Sharma, "Adaptive texture and color segmentation for tracking moving objects," *Pattern Recognition*, vol. 35, pp. 2013–2029, October 2002.

[12] P. Pérez, C. Hue, J. Vermaak, and M. Ganget, "Color-based probabilistic tracking," in *Proceedings of the 7th European Conference on Computer Vision. Vol. 2350, LNCS*, Copenhagen, Denmark, May 2002, pp. 661–675.

[13] E. Poon and D. Fleet, "Hybrid Monte Carlo filtering: Edge-based tracking people," in *Proceedings of the IEEE Workshop on Motion and Video Computing*, Orlando, Florida, Dec. 2002, pp. 151–158.

[14] K. Toyama and G. Hager, "Incremental focus of attention for robust vision-based tracking," *International Journal of Computer Vision*, vol. 35, no. 1, pp. 45–63, 1999.

[15] J. Triesch and C. von der Malsburg, "Democratic integration: Self-organized integration of adaptive cues," *Neural Computation*, vol. 13, no. 9, pp. 2049–2074, 2001.

[16] A. Jepson, D. Fleet, and T. F. El-Maraghi, "Robust online appearance models for visial tracking," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 25, no. 10, pp. 1296–1311, 2003.

[17] P. Brasnett, L. Mihaylova, N. Canagarajah, and D. Bull, "Particle filtering with multiple cues for object tracking in video sequences," in *Proc. of SPIE's 17th Annual Symposium on Electronic Imaging, Science and Technology, V. 5685*, 2005, pp. 430–441.

[18] N. Bergman, "Recursive Bayesian estimation: Navigation and tracking applications," Ph.D. dissertation, Linköping University, Linköping, Sweden, 1999.

[19] J. Liu and R. Chen, "Sequential Monte Carlo methods for dynamic systems," *Journal of the American Statistical Association*, vol. 93, no. 443, pp. 1032–1044, 1998. [Online]. Available: citeseer.nj.nec.com/article/liu98sequential.html

[20] E. Wan and R. van der Merwe, *Kalman Filtering and Neural Networks.* Wiley Publishing, September 2001, chapter 7. the Unscented Kalman filter 7, pp. 221–280.

[21] Y. Bar-Shalom and X. Li, *Estimation and Tracking: Principles, Techniques and Software.* Artech House, 1993.

[22] M. Isard and A. Blake, "Condensation - conditional density propogation for visual tracking," *International Journal of Computer Vision*, vol. 29, no. 1, pp. 5–28, 1998.

[23] ——, "Icondensation: Unifying low-level and high-level tracking in a stochastic framework," in *Proceedings of the 5th European Conference on Computer Vision*, vol. 1, 1998, pp. 893–908.

[24] R. Porter, *Texture Classification and Segmentation, PhD Thesis*, University of Bristol, Center for Communications Research, 1997.

[25] W. Freeman and E. Adelson, "The design and use of steerable filters," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, 1991.

[26] A. Karasaridis and E. P. Simoncelli, "A filter design technique for steerable pyramid image transforms," in *Proceedings of the ICASSP*, Atlanta, GA, May 1996.

[27] A. Bhattacharayya, "On a measure of divergence between two statistical populations defined by their probability distributions," *Bulletin of the Calcutta Mathematical Society*, vol. 35, pp. 99–110, 1943.

[28] D. Scott, *Multivariate Density Estimation: Theory, Practice and Visualization*, ser. Probability and Mathematical Statistics. John Wiley and Sons, 1992.

[29] R. Manduchi, "Bayesian fusion of colour and texture segmentations," in *Proc. of the IEEE International Conference on Computer Vision*, September 1999.

[30] E. Saber and A. M. Tekalp, "Integration of color, edge, shape and texture features for automatic region-based image annotation and retrieval," *Journal of Electronic Imaging (Special Issue)*, vol. 7, no. 3, pp. 684–700, 1998.

[31] X. Shi and R. Manduchi, "A study on Bayes feature fusion for image classification," in *Proceedings of the IEEE Workshop on Statistical Analysis in Computer Vision*, vol. 8, June 2003.

[32] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation.* John Wiley and Sons, 2001.

[33] "BBC Radio 1, `http://www.bbc.co.uk/calc/radio1/ index.shtml`," 2005, Creative Archive Licence.

[34] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 5, pp. 603–619, 2002.