Proc. VIIth Digital Image Computing: Techniques and Applications, Sun C., Talbot H., Ourselin S. and Adriaansen T. (Eds.), 10-12 Dec. 2003, Sydney

Probabilistic Multiple Cue Integration for Particle Filter Based Tracking

Chunhua Shen, Anton van den Hengel, and Anthony Dick

School of Computer Science, University of Adelaide, Adelaide, SA 5005, Australia
CRC for Sensor Signal & Information Processing, Mawson Lakes, SA 5095, Australia
{chhshen,anton,ard}@cs.adelaide.edu.au

Abstract. Robust visual tracking has become an important topic in the field of computer vision. The integration of cues such as color, edge strength and motion has proved to be a promising approach to robust visual tracking in situations where no single cue is suitable. In this paper, an algorithm is presented which integrates multiple cues in a probabilistic manner. Specifically the likelihood of each cue is calculated and weighted before Bayes' rule is applied to obtain the resultant posterior. This posterior is generally not well represented analytically, and is therefore represented as a set of weighted particles, which is updated at each frame by a particle filter. This paper demonstrates how the combination of multiple cue integration and particle filtering results in a robust tracking method. We also demonstrate how each cue's weight can be adapted on-line during the tracking procedure.

1 Introduction

In recent years there has been much interest in visual tracking. Its potential applications include intelligent robotics, video surveillance, image sequence analysis and human-computer interfaces [1]. However, developing a visual tracking algorithm that is robust to a wide variety of conditions is still an open problem.

Part of this problem is the choice of what to track. Color trackers, for example, are distracted from their target by other objects of similar color, while edge-based trackers can be misled by clutter in the background. The combination of multiple cues can achieve more reliable tracking than any one cue in isolation. Birchfield [2], for example, uses intensity gradient and a color histogram of the target for robust head tracking. This algorithm improves tracking robustness and accuracy by utilizing the property of cue complementarity. The primary limitation of such an integration strategy is that each cue has the same *fixed* level of confidence associated with it. This means that each cue is assumed to possess the same reliability in each frame of video.

Democratic integration, proposed by Triesch and von der Malsburg [3] fuses multiple cues in an adaptive manner so that the contribution of each cue depends on its estimated reliability in the current environment. Consequently such an algorithm is robust with respect to a dynamically changing background. This approach is based on voting techniques in which each cue generates an independent decision concerning the tracking before these decisions are integrated using a weighted sum. Recently Spengler and Schiele [4] have extended this method to multiple hypotheses tracking by integrating it with the CONDENSATION [5] algorithm. Their scheme, however, is nonadaptive. Due to the conflict arising from the single-hypothesis property of democratic integration and the multiple hypotheses generated by CONDENSATION, the saliency maps recovered from each independent cue are combined together with a fixed weighted sum. An apparent drawback of this method is that the methodology employed for fusing the different cues is *ad hoc*, depending upon manual tuning of the weights.

In the field of image segmentation, which shares many techniques with tracking, some methods based on Bayes' rule and the Expectation Maximization (EM) algorithm have been introduced to probabilistically combine cues based on their estimated likelihood (e.g. [6, 7]). For probabilistic tracking with multiple cues, most previous work focuses on incorporating reliability measures with dynamic Bayes networks (DBNs); usually Gaussian Mixture Models (GMMs) are used to represent the probability density distributions [8, 9]. The advantage of such an approach is that the model can be updated tractably with the EM algorithm. Toyama and Horvitz [10] also use a DBN modelling the probabilistic dependence of multiple cues to weight the cues based on their inferred accuracies. In [11], Wu *et al.* propose a novel learning method based on a factorial graphical model where different shape and color distributions are adapted on-line in a co-inference way.

The method presented here uses a feedback loop to perform the integration of cues in a manner similar to that used by democratic integration. Within this loop, discordant cues are suppressed quickly while cues that have proved to be reliable in the recent past are given more weight. In our experiments, two statistical features are used, namely, color distribution density and edge information.

Color information is an appealing feature for deformable object tracking due to its robustness to spatial rotation, non-rigidity and partial occlusion [12–14]. The mean shift tracker in [12] is essentially a nonparametric density gradient estimator in color histogram density space. Unlike this deterministic search, [13, 14] propose a probabilistic method using a color-based particle filter based on mean shift density estimation. The color cue used by our algorithm is based on this model.

Despite the robustness of the color cue, it does not contain any information about the spatial adjacency of pixels in the object. To complement the color cue, edge detection is adopted to describe shape information, which has proved to be an effective feature in visual contour tracking [1, 5]. In this paper, these two features are integrated to demonstrate the validity of the proposed algorithm. In most applications, environmental changes do not affect both these visual cues simultaneously. Thus dynamically adjusting their influence improves the system's performance. Details on implementing these cues are shown in Section 3.

Tracking a deformable object in a high dimensional state space requires tractable methods for inferring the posterior probability distributions. Particle filters, also known as bootstrap filters, sequential Monte Carlo methods or Condensation [5, 15, 16], together with their variants [17, 18], have been extensively studied and successfully applied to visual tracking. The tracking engine in our tracking framework is a modified particle filter.

The outline of the paper is as follows. In Section 2 we review visual tracking as probability propagation in more detail. Section 3 describes specific cues used in our work. The probabilistic integration scheme of multiple cues is presented in Section 4. We also present a method to incorporate reliability measurement and discuss how to continuously update each cue's weight in our probabilistic cue integration in Section 4. Section 5 presents a method for target model updating. Experiments on real video sequences are demonstrated in Section 6. Finally some discussion is reported and the concluding remarks are drawn in Section 7.

2 Particle Filtering for Visual Tracking

In this section, we briefly present the probability propagation model for tracking, and its dynamical model component. The observation model, which concerns multiple cue integration, will be demonstrated in the following section.

2.1 Particle Filtering

With the first-order Markovian assumption $p(\mathbf{x}_t|\mathbf{x}_{1:t-1}) = p(\mathbf{x}_t|\mathbf{x}_{t-1})$, the problem of tracking can be formulated in the Bayesian filtering framework [15] as

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) \propto \mathcal{L}(\mathbf{z}_t|\mathbf{x}_t) \int p(\mathbf{x}_t|\mathbf{x}_{t-1}) p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})\, d\mathbf{x}_{t-1} \tag{1}$$

where $\mathbf{x}_t$ and $\mathbf{z}_t$ are the state and observation vectors at time t, and $\mathbf{x}_{1:t-1}$ and $\mathbf{z}_{1:t-1}$ their histories respectively. With Eq. (1) we can calculate the posterior recursively with a particle filer, given the dynamical model $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ and observation likelihood model $\mathcal{L}(\mathbf{z}_t|\mathbf{x}_t)$. Unlike linear prediction algorithms such as the Kalman filter, no assumptions of Gaussian transition and observation noise are made for the particle filter.

The key feature of the particle filter is that the posterior is approximately represented by a set of particles, each particle including a state vector $\mathbf{x}$ and an associated weight ω: $\{(\mathbf{x}_t^{(i)}, \omega_t^{(i)}), i = 1, ..., N\}$. The weights are normalized such that $\sum_i \omega_t^{(i)} = 1$. Suppose we could sample the particles from an auxiliary density, i.e. $\mathbf{x}_t^{(i)} \sim f(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)}, \mathbf{z}_{1:t})$. Then the new particle weights are set to

$$\omega_t^{(i)} \propto \frac{\mathcal{L}(\mathbf{z}_t|\mathbf{x}_t^{(i)}) p(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(i)})}{f(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(i)}, \mathbf{z}_{1:t})}. \tag{2}$$

The last step is to re-sample the particles to ensure the efficiency of the evolution [16]. The general particle filter algorithm is displayed in Fig. 1. Note that when

1. **Initialization**:
   For $i = 1, ..., N$, sample particles from the prior $\mathbf{x}_0^{(i)} \sim p(\mathbf{x_0})$ and set $t = 1$.
2. **Importance sampling step**:
   For $i = 1, ..., N$, sample $\mathbf{x}_t^{(i)} \sim f(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)}, \mathbf{z}_{1:t})$ and evaluate the weights according to Eq. (2). Then normalize the weights.
3. **Selection(re-sampling) step**:
   Re-sample to obtain N replacement particles according to the probabilities $\omega_t^{(i)}$. Set $t = t + 1$, go to step 2.

Fig. 1. The standard particle filter algorithm.

the proposed importance distribution is chosen as the distribution conditional on the state at the previous discrete time, i.e., in Eq. (2), the importance function becomes $f(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(i)}, \mathbf{z}_{1:t}) = p(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(i)})$, then the weighting equation reduces to $\omega_t^{(i)} \propto \mathcal{L}(\mathbf{z}_t|\mathbf{x}_t^{(i)})$. This simplification produces a variant of a well known particle filter in computer vision, CONDENSATION [5].

2.2 State Space and the Dynamical Model

We want to track a patch of interest in the image plane. The patch can be an upright ellipse or rectangle in shape and is parameterized by

$$\mathbf{x} = \{x, y, \dot{x}, \dot{y}, s_x, s_y, \dot{s}_x, \dot{s}_y\} \tag{3}$$

where x and y denote the centroid of the ellipse or rectangle, $\dot{x}$ and $\dot{y}$ the velocities of the centroid, s_x and s_y the length of the half axes or sides and $\dot{s}_x$ and $\dot{s}_y$ the velocities of s_x and s_y (see [13, 14]). We use a first order auto-regression (AR) equation to model the dynamics. This has the form

$$\mathbf{x}_t = \mathbf{A}\mathbf{x}_{t-1} + \mathbf{B}\mathbf{v}_t \tag{4}$$

where $\mathbf{v}_t$ is a multivariate normal distribution and the matrices $\mathbf{A}$ defines the deterministic component and $\mathbf{B}$, the stochastic component. It is straightforward to extend this model to second order if it is required to capture more complex dynamical motions.

3 Individual Cue

While the particle filter is a powerful tool for robust tracking, its success depends largely on the choice of appropriate features to track. In this paper, we use color and shape information. Our choice of features is based on their feasibility for robust visual tracking and their adaptability to our probabilistic framework. In contrast to the implementation methods in [3, 4], we use these cues in a probabilistic way so that they can be integrated into the particle filtering framework more naturally.

3.1 Color Cue

We follow the model presented in [13, 14] to encode the color information. In RGB space, color histograms are calculated with m bins. In our experiments, $8 \times 8 \times 8$ bins are sufficient to represent the color distribution for pixels with 8-bit color depth in each channel. After obtaining the color histogram, with the mean shift algorithm, we calculate the color distribution density $\boldsymbol{\phi}(\mathbf{x}) = \{\phi^{(u)}(\mathbf{x})\}_{u=1...m}$ at location $\mathbf{x}$ (for details, refer to [12]).

At frame t, the color model $\boldsymbol{\phi}_t(\mathbf{x})$ whose corresponding state is $\mathbf{x}$ [1] will be compared to the target color model $\boldsymbol{\phi}_0(\mathbf{x}_0)$ which is manually selected or generated by an automatic detection algorithm.

The similarity is measured by the Bhattacharyya distance,

$$\rho[\boldsymbol{\phi}_t(\mathbf{x}), \boldsymbol{\phi}_0(\mathbf{x}_0)] = \sum_{u=1}^{m} \sqrt{\phi_t^{(u)}(\mathbf{x}_0)\phi_0^{(u)}(\mathbf{x}_0)}, \tag{5}$$

$$D[\boldsymbol{\phi}_t(\mathbf{x}), \boldsymbol{\phi}_0(\mathbf{x}_0)] = \sqrt{1 - \rho[\boldsymbol{\phi}_t(\mathbf{x}), \boldsymbol{\phi}_0(\mathbf{x}_0)]}. \tag{6}$$

Note that both of these distributions are required to be normalized first. In the tracking procedure, the estimate will be updated continuously.

As in [13, 14], the likelihood function with respect to the similarity distances is modelled as a Gaussian distribution

$$\mathcal{L}_{\texttt{color}}(\mathbf{z}_{\texttt{color},t}|\mathbf{x}_t) \propto \exp\left(-\kappa D[\boldsymbol{\phi}_t(\mathbf{x}_t), \boldsymbol{\phi}_0(\mathbf{x}_0)]^2\right). \tag{7}$$

where κ is a constant determined by the Gaussian variance.

3.2 Shape Cue

To complement the color cue, which contains no information about shape, a shape contour is included in our object model. Instead of using B-splines to model a detailed contour, a parametric ellipse or rectangle is used as the shape model. In our experiment, we use an ellipse to model the contour of the human face. Therefore calculating the likelihood of the shape cue is based on an ellipse.

The likelihood of the ellipse is computed at a point p on the ellipse as follows [1, 5, 19]. A measurement line is constructed passing through the point p and the centre of the ellipse. n sample points are generated at fixed intervals along this line, centered about p. A Canny edge detector is then applied at each sample point.

Under the assumption that the true edge point is normally distributed with zero mean and variance σ^2, and the clutter is a Poisson process with density λ, the likelihood of the observation at a sample point is

$$\mathcal{L}_{\texttt{shape}}(\mathbf{z}^{(l)}_{\texttt{shape},t}|\mathbf{x}_t) \propto 1 + \frac{1}{\sqrt{2\pi}\sigma h_0 \lambda} \sum_{j=1}^{n_l} \exp\left[-\frac{(z_j - x)^2}{2\sigma^2}\right], \tag{8}$$

[1] For clarity, we denote the pixel location $\{x, y\}$ by the corresponding state vector $\mathbf{x} = \{x, y, ...\}$.

where h_0 is the prior probability that no true contour edge is detected and z_j is the distance of the detected feature point from the ellipse. See [1, 20] for details. If it is assumed that the likelihoods on each measurement line are statistically independent, then the overall likelihood for m lines evenly spaced around the ellipse is

$$\mathcal{L}_{\texttt{shape}}(\mathbf{z}_{\texttt{shape},t}|\mathbf{x}_t) = \prod_{l=1}^{m} \mathcal{L}_{\texttt{shape}}(\mathbf{z}^{(l)}_{\texttt{shape},t}|\mathbf{x}_t). \tag{9}$$

4 Integration of Multiple Cues into Particle Filtering

Under the assumption that the observation from each cue is statistically independent, the entire likelihood given the state $\mathbf{x}_t$ is written

$$\mathcal{L}(\mathbf{z}_t|\mathbf{x}_t) = \mathcal{L}_{\texttt{color}}(\mathbf{z}_{\texttt{color},t}|\mathbf{x}_t) \cdot \mathcal{L}_{\texttt{shape}}(\mathbf{z}_{\texttt{shape},t}|\mathbf{x}_t). \tag{10}$$

To adapt to the reliability of each cue, the likelihood function is slightly adjusted [6]:

$$\mathcal{L}(\mathbf{z}_t|\mathbf{x}_t) = [\mathcal{L}_{\texttt{color}}(\mathbf{z}_{\texttt{color},t}|\mathbf{x}_t)]^{\alpha_1} \cdot [\mathcal{L}_{\texttt{shape}}(\mathbf{z}_{\texttt{shape},t}|\mathbf{x}_t)]^{\alpha_2}, \tag{11}$$

where $0 \le \alpha_1, \alpha_2 \le 1$ are defined as the reliability factors for color and shape cues. The weights need not sum to unity as for democratic integration. If a cue is completely unreliable then the weight α_k for that cue should be set to zero so that $[\mathcal{L}(\mathbf{z}_t|\mathbf{x}_t)]^{\alpha_k} = 1$ ($k = 1, 2$ in our case) which implies that the measurement for that cue has no effect. Conversely if a cue is totally reliable then $\alpha_k = 1$. In this paper, we take only color and shape cues into consideration; however, it is straightforward to include extra cues into this framework.

The multi-cue likelihood Eq. (11) can be substituted into Eq. (2), and the re-weighting step, to obtain a particle filter that includes information from multiple cues. The weighting parameters can be fixed and determined by some prior knowledge [4]. However, it is beneficial to introduce an on-line adaptation scheme to weight the cues with a probabilistic technique. The following section shows how to introduce an adaptation scheme within the particle filtering framework.

4.1 Adaptation of Cue Weights

The democratic integration [3] method includes a re-weighting scheme in which each cue is associated with a score based on the error between the individual cue's saliency and the average saliency.

We extend this idea to the particle filtering framework. A score $\gamma_{k,t}$ at frame t ($k = 1, 2$ in our case) is computed for each cue, which measures how well it agrees with the result. The detailed procedure to compute the score $\gamma_{k,t}$ is as follows. Suppose that the image observation changes slowly such that we can predict $\gamma_{k,t}$ from the previous $\gamma_{k,t-1}$ and $\alpha_{k,t-1}$. We compute the difference between the tracking result obtained by that cue alone and the result obtained by fusing the likelihood from all cues. The measurement of the difference can be an L_2-norm distance between the centroids of the tracked regions. Note that in this procedure, no additional heavy computation is required to calculate the tracking result for each cue because the bottle-neck is the calculation of the likelihood for each cue, which only needs to be computed once at each frame.

Denoting the L_2-norm distance by $\bar{E}_{k,t}$ for each cue at frame t, the scores can be computed as

$$\gamma_{k,t} = \frac{\tanh(-a\bar{E}_{k,t} + b) + 1}{2} \tag{12}$$

where a, b are pre-defined constants and $\tanh(\cdot)$ is the hyperbolic tangent function[2]. In our experiments, $a = 0.4\ pixel^{-1}$ and $b = 3$.

The weights are then adapted according to a leaking integrator,

$$\xi \dot{\alpha}_{k,t} = \gamma_{k,t} - \alpha_{k,t} \tag{13}$$

where ξ is a parameter which determines the changing rate of the weights [3,7].

Different from our criterion determining the reliability of tracking, Shearer *et al.* [21] measured the success of tracking based on the consistency of object velocity, which can also be used in our framework.

5 Target Model Updating

In our work, both *bottom-up* and *top-down* methodologies are used. Because the proposed color cue tracking is a *top-down* template matching approach and the target we are interested in is time-varying, it is necessary to update the template during the tracking procedure. After we track one frame successfully, we can use the tracked region of this frame to update the target model. For parametric models, *e.g.* GMMs, some work has been done on updating the model parameters on-line [9]. For nonparametric models, as proposed in [13], we put more weight on recent tracking results than on older tracking results by using a *forgetting* parameter. The question then arises of how to decide whether we should believe the current tracking result is reliable and use this result to update the target model. The average likelihood before normalization over all particles, $\bar{\mathcal{L}}$, could be a good criteria. The higher the average likelihood $\bar{\mathcal{L}}$, the more reliable the current result is. Therefore we compare the average likelihood with a pre-defined threshold and update the model only when $\bar{\mathcal{L}}$ is larger than the threshold. A similar approach is used for model updates in [13].

The target model can be initialized with a detector algorithm (*e.g.*, for face tracking, a face detector can be used). Alternatively we can specify an image patch as the target model in the first frame by hand.

[2] In the field of neural networks, such functions are usually adopted as stimuli functions.

6 Evaluation

This section evaluates the performance of the proposed integration and adaptation algorithm. The performance of both single cue and multiple cues are compared[3].

Our experiments show that the tracking algorithm with multiple cues performs much more robustly than those with a single cue. The tracking results are presented in Fig. 2. In the experiments, the resolution of the image sequences is

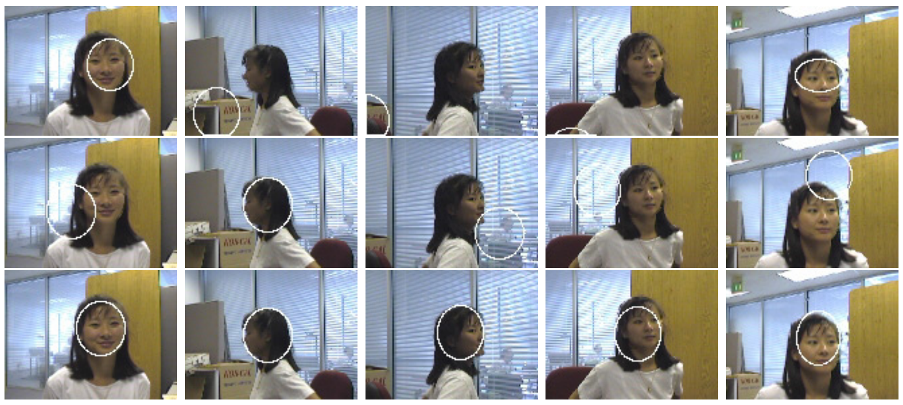

Fig. 2. Tracking results with color cue alone (top), shape cue alone (middle) and multiple cue integration (bottom). From left to right, the corresponding frame number is frame 17, 87, 116, 283 and 390 out of total 500 frames. At frame 87, the color based particle filter is trapped in a false region when the subject's face turns around, drastically changing the target model. The color tracker fails for most of the remaining frames. The color based mean shift tracker [12] fails at the same position. The single shape cue based particle filter is distracted severely by the clutter at frame 17, 116, 283 and 390, as the window blind beside the head contains many edges. The multiple cue particle filter, however, tracks the face accurately through the whole sequence.

128×96 and the frame rate is 10 fps. The human head is modelled as an upright ellipse as discussed in Section 2.2. For the color cue, $8 \times 8 \times 8$ bins are used to represent the color histogram in the RGB space. For the shape cue, for each ellipse, 25 equally spaced rays are used and $\sigma = 6\,pixels$. Each line has $2\sigma + 1$ observation locations. For particle filters in all the cases, the number of particles is $N = 100$.

The sequence is set in a office environment with a cluttered background which contains many clutter edges to distract the contour tracker. The color of the door is very similar to the skin color, which can confuse the color tracker. Additionally, the tracked subject turns her head around which creates sudden color changes

[3] The tracking result described in this paper can be accessed at `http://www.cs.adelaide.edu.au/~vision/demo/`.
The test image sequences (courtesy of Dr. Birchfield) are available at `http://robotics.stanford.edu/~birch/headtracker/seq/`.

of the visible side of her head. At many frames the face is totally invisible, which can make color based tracking algorithm lose the target completely. In the experiment, neither single cue can track the subject well. Our algorithm succeeds because it monitors the saliency of each cue and updates its influence on the result accordingly.

Fig. 3 depicts the evolution of the color and shape cue. We can see that at some certain frames, the proposed integration scheme successfully suppresses the cue which is unreliable for tracking.

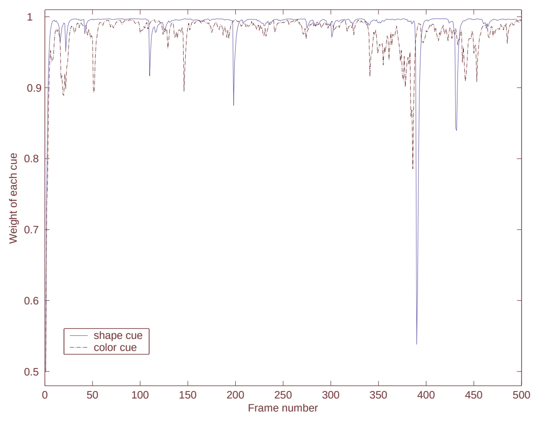

Fig. 3. The evolution of the cue weights for Bayesian multiple cues integration. At frame 390, the shape cue is quite unliable (see Fig. 2) while the color cue works well, in this figure a minimum appears for the shape cue curve, which shows it is suppressed.

7 Discussion and Conclusion

This paper has presented a probabilistic multi-cue integration framework for robust visual object tracking. In contrast to democratic integration, each cue's likelihood and weight is calculated, and then Bayes' rule is applied to obtain the resulting posterior. This algorithm is well suited to incorporation into a particle filter in the procedure of particle weighting. Also presented was a criterion to measure the reliability associated with each cue such that each cue's weight can be adapted on-line during the tracking procedure. Experimental data show such an adaptation scheme is effective.

Future work includes exploring new criteria to measure the reliability of each cue and extending this framework to multiple object tracking. As the number of cues increases, further work will also be required to minimize the number of tuning parameters required.

References

1. Blake, A., Isard, M.: Active Contours. Spinger, Berlin (1998)

2. Birchfield, S.: Elliptical head tracking using intensity gradients and color histograms. In: IEEE International Conference on Computer Vision and Pattern Recognition, Santa Barbara, CA (1998) 232–237
3. Triesch, J., von der Malsburg, C.: Democratic integration: Self-organized integration of adaptive cues. Neural Computation **13** (2001) 2049–2074
4. Spengler, M., Schiele, B.: Towards robust multi-cue integration for visual tracking. Machine Vision and Applications **14** (2003) 50–58
5. Isard, M., Blake, A.: CONDENSATION – conditional density propagation for visual tracking. International Journal of Computer Vision **29** (1998) 5–28
6. Khan, S., Shah, M.: Object based segmentation of video using color, motion and spatial information. In: IEEE Conference on Computer Vision and Pattern Recognition. Volume 2., Kauai, Hawaii (2001) 746–751
7. Hayman, E., Eklundh, J.: Probabilistic and voting approaches to cue integration for figure-ground segmentation. In: 7th European Conference on Computer Vision. Volume 2352 of Lecture Notes in Computer Science., Copenhagen, Denmark, Springer (2002) 469–486
8. Zhu, Y., Fujimura, K.: Driver face tracking using Gaussian mixture model (GMM). In: IEEE Intelligent Vehicles Symposium, Columbus, OH, USA (2003) 587–592
9. McKenna, S.J., Raja, Y., Gong, S.: Object tracking using adaptive colour mixture models. In: Asian Conference on Computer Vision. Volume 1351 of Lecture Notes in Computer Science., Hong Kong (1998) 607–614
10. Toyama, K., Horvitz, E.: Bayesian modality fusion: Probabilistic integration of multiple vision algorithms for head tracking. In: Asian Conference on Computer Vision, Tapei, Taiwan (2000)
11. Wu, Y., Huang, T.: A co-inference approach to robust visual tracking. In: IEEE International Conference on Computer Vision. Volume 2., Vancouver, Canada (2001) 26–33
12. Comaniciu, D., Ramesh, V., Meer, P.: Kernel-based object tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence **25** (2003) 564–577
13. Nummiaro, K., Koller-Meier, E., Gool, L.V.: An adaptive color-based particle filter. Image and Vision Computing **21** (2003) 99–110
14. Pérez, P., Hue, C., Vermaak, J., Gangnet, M.: Color-based probabilistic tracking. In: 7th European Conference on Computer Vision. Volume 2350 of Lecture Notes in Computer Science., Copenhagen, Denmark, Springer (2002) 661–675
15. Arulampalam, S., Maskell, S., Gordon, N., Clapp, T.: A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking. IEEE Transactions on Signal Processing **50** (2002) 174–188
16. Doucet, A., de Freitas, N., Gordon, N., eds.: Sequential Monte Carlo Methods in Practice, New York, Springer-Verlag (2001)
17. van der Merwe, R., Doucet, A., de Freitas, N., Wan, E.: The unscented particle filter. Technical Report CUED/F-INFENG/TR 380, Cambridge University Department of Engineering, UK (2000)
18. Rui, Y., Chen, Y.: Better proposal distributions: Object tracking using unscented particle filter. In: IEEE Conference on Computer Vision and Pattern Recognition. Volume 2., Kauai, Hawaii (2001) 786–793
19. Li, P., Zhang, T., Pece, A.E.C.: Visual contour tracking based on particle filters. Image and Vision Computing **21** (2003) 111–123
20. MacCormick, J.: Probabilistic modelling and stochastic algorithms for visual localisation and tracking. PhD thesis, University of Oxford, UK (2000)
21. Shearer, K., Wong, K.D., Venkatesh, S.: Combining multiple tracking algorithms for improved general performance. Pattern Recognition **34** (2001) 1257–1269