A Robust Particle Filter-based Face Tracker Using Combination of Color and Geometric Information

Bogdan Raducanu[1] y Jordi Vitrià[1,2]

[1] Centre de Visió per Computador, Edifici O - Campus UAB
[2] Departament de Cienciès de la Computació, Universitat Autònoma de Barcelona
08193 Bellaterra, Barcelona
Spain
E-mail: {bogdan, jordi}@cvc.uab.es

Abstract. Particle filtering is one of the most successful approaches for visual tracking. However, so far, most particle-filter trackers are limited to a single cue. This can be a serious limitation, since it can reduce the tracker's robustness. In the current work, we present a multiple cue integration approach applied for face tracking, based on color and geometric properties. We tested it over several video sequences and we show it is very robust against changes in face appearance, scale and pose. Moreover, our technique is proposed as a contextual information for human presence detection.

1 Introduction

Face tracking is required for a large number of computer applications: HCI, surveillance, biometry, video compression, human-robot communication, etc. There are many approaches proposed to solve this problem, but among them a very common technique is represented by particle filter. Particle filter has been introduced in [7] and its basic idea consists of propagating a probability distribution through time, based on sample sets. Its powerfulness is represented by its ability to track simultaneously non-linear/non-gaussian multi-modal distributions, solving in this way the limitations imposed by Kalman filter [24]. Since it was introduced, it was used mainly for object-tracking purposes, but it also has been proposed in robotics for the problem of self-localization for mobile robots [20].

Particle filtering has been mainly used to track objects based on their color histograms [26], [12], [15], [13] or geometrical shape [7], [6]. The use of single cue proved to offer good results, but since the most applications are intended for dynamic environments, with continuous changes in illumination conditions, shape, partial or total occlusion of the object being tracked, this could pose a limitation in the robustness of these methods. The solution is represented by the integration of several cues, like in [16], [21], [14].

In the current paper, we propose a novel method for face tracking based on the fusion of color and geometrical structure of the face. In other words, the likelihood between the model and the hypotheses ("particles") is expressed in terms of a linear combination between color similarity, difference in size and euclidean distance. The novelty is represented by the fact that in order to initialize the tracker and to update the confidence, a model-based approach for face detection is used. The face evidence is confirmed using Haar-like features trained with the Ada-Boost algorithm. In case of self-occlusions (head is rotated), when the detector fails despite a person continues to be present in front of the camera, a "CamShift" algorithm is employed, in order to confirm the face presence based on skin color. This could represent a great advantage, since some approaches are using either manual initialization or a learned histogram of the object to be tracked. We try to argument the relevance of our method and the improvement it introduces, by comparing it with other recent publications:

- instead of working with the RGB color model (like in [12]), we prefer HSV, since it is less sensitive to changes in illumination conditions

- the "mean-shift" algorithm from [17] was replaced by the "CamShift" algorithm. "Camshift" [3] stays for "continuous adaptive mean-shift", and unlike the "mean-shift" , which is designed for static distributions, "CamShift" can deal with dynamically changing distributions.

The paper is structured as follows: section 2 contains a brief recall to particle filter; section 3 focuses upon observation model; in section 4, the whole system is presented and the integration of several cues is explained; some experimental results are reported in section 5; finally, section 6 contains our conclusions and the guiding lines for future work.

2 Particle filtering

The tracking problem can be formulated in the framework of partially observable Markov chains [5] that considers the evolution of a state vector sequence x_k over time. This implies the estimation of the *a posteriori* density over the state x_k from all available sensor measurements $z_{0:k} = z_0 ... z_k$. A solution to this problem is given by Bayes filters [8], which computes this *a posteriori* density recursively (we make here a first-order Markov assumption, i.e. the state x_k depends only on x_{k-1}):

$$p\left(x_k|z_{0:k}\right) = K \cdot p\left(z_k|x_k\right) \cdot \int p\left(x_k|x_{k-1}\right) \cdot p\left(x_{k-1}|z_{0:k-1}\right) dx_{k-1} \tag{1}$$

If the a posteriori distribution at any moment is Gaussian and the distributions $p\left(x_k|x_{k-1}\right)$ and $p\left(z_k|x_k\right)$ are linear in its arguments, than the equation (1) reduces to the Kalman filter. If one of these conditions cannot be met, than a possible alternative is to use some approximation methods in order to linearize the actuation and measurements models. There are two extensions of the Kalman filter that can be used in these situations, known as Extended Kalman

Filter [10] and Unscented Kalman Filter [9]. However, most of the real-world processes are non-linear and these approximations don't hold. The solution to the general case is given by Particle Filtering.

The basic idea of a Particle Filter is to maintain a multiple hypothesis (*particles*) about the object being tracked. This means that the distribution $p\left(x_k|z_{0:k}\right)$ is represented using a set $\{s_k^n, w_k^n\}$, where $n = 1..N$ (N is the number of particles). The weights should be normalized such that $\sum\limits_n w_n = 1$. In other words, each particle has associated a weight that reflects its relevance in representing the object. The initial set of particles can be randomly generated or using the output of a detector. The *a posteriori* probability is updated in a recursive way, according to the following steps:

- from the previous set of particles $\left\{s_{k-1}^n, w_{k-1}^n\right\}_{n=1..N}$, draw one particle through importance sampling, so s_k^m will correspond to some s_{k-1}^j:

$$s_k^j \sim w_{k-1}^j \tag{2}$$

- the chosen particle is propagated according to the model's dynamics:

$$s_k^n \sim p\left(x_k|x_{k-1} = s_k^m\right) \tag{3}$$

- assign a new weight to the recent generated particle equal to the likelihood of the observation, i.e.:

$$w_k^n \sim p\left(z_k|x_k = s_k^n\right) \tag{4}$$

then normalize again the resulting weights and store the new obtained set of particles.

Once the N particles have been constructed, an estimate for the current object state can be obtained from the following formula:

$$\xi\left[f\left(x_k\right)\right] = \sum_{n=1}^{N} w_k^n f\left(s_k^n\right) \tag{5}$$

where $f\left(x\right) = x$.

From time to time, it is necessary to resample the particles, otherwise could appear degeneracy problems, which is the concentration of the whole weight on a single particle. The resampling procedure duplicates the particles with higher weights, while discards those with lower weights. In consequence, without resampling the performance of the algorithm will be considerably degraded. A more complete discussion about the degeneracy problem can be found in [2].

In regard with the dynamical model, we chose a simple one, described by a linear stochastic differential equation:

$$x_k = Ax_{k-1} + Bu_k \tag{6}$$

where A is the state transition matrix, u_k is a vector of standard normal random variates and BB^T is the process noise covariance.

3 Model-based approach

In order to estimate and improve the accuracy in the prediction of the particle filter, two other independent methods have been used. The integration between a motion-based approach and a model-based approach for tracking is not new and was previously proposed in [11] and more recently in [25]. The purpose of integrating these two approaches is twofold. First, the face detector is used to initialize the face tracker. There are other methods to initialize the tracker (either automatically, starting with a random position and expecting that after a number of frames it will converge towards the solution, or manually). In our case we opted for an automatic method which guarantees the best initialization for the tracker. On the other hand, our system is aimed for long-term use (i.e. hours, maybe even days). We plan later on to use it for automatic acquisition of faces under different poses. In this situation, the reason for using a face detector would be to reinforce and to help the tracker to recover from possible failures when it gets distracted. Without this confirmation from the detector, the system won't run properly for more than a few hundred frames. The consequence will be that the system will fail quite often and will require repetitive initialization.

As first method, we employed a face detector based on Haar-like features trained with the Ada-Boost. This works very well for frontal faces. In case of self-occlusion (head is rotated), which is the most common source of failures for the face detector, we substitute this evidence with a, color-exclusive method, represented by the "CamShift" algorithm. The use of these two approaches is complementary, since the use of a color-exclusive method (without having any other independent information of facial features) could degrade the performance of the tracker. Passing a hand in front of the face (which has the same color distribution), could fool the tracker and thus loose the real target. We always prefer the use of the first method, over the second.

3.1 Face detection using Haar-like features

Ideally, techniques for face detection are desired to show robustness against changes in illumination, scale and head poses. They should be able to detect faces despite variation in facial expression and the presence or absence of natural or artificial accessories like beards and glasses. The face detection techniques can be divided in two main classes: holistic and local ones. The weakness of the holistic approaches is that they represent the images as raster vectors without any information about the 2D topology of facial features. On the other hand, face detectors based on local information have the advantage of providing more accurate information about the presence of face parts, by classifying image patches in 'face' or 'non-face' vectors.

Following this second path, we implemented a boosted face detector similar to those proposed in [22]. The idea is to use a cascade of weak classifiers in order to produce a stronger one. Each classifier is trained with a few hundreds samples of a particular object (a face), using Haar-like wavelet features, depicted in figure 1.

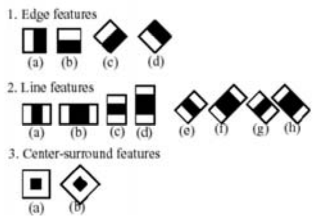


Fig. 1. Haar-like wavelet features used to train the weak classifiers.

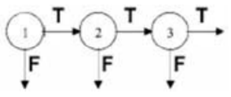


Fig. 2. The cascade representation of weak classifiers. The features that are wrong classified are rejected through the 'F' output.

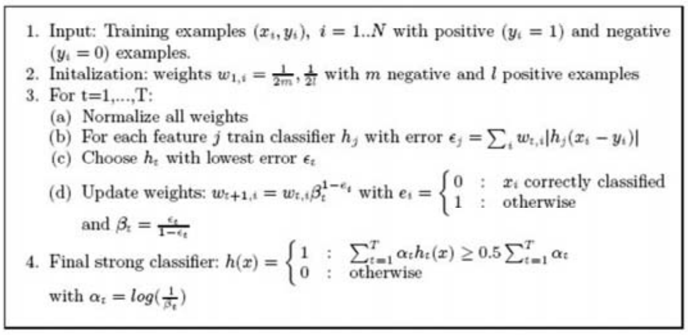
After each training epoch, the wrong classified features are retrained. Each classifier outputs "1" if the corresponding sample corresponds to a face and "0" otherwise. The final strong classifier is obtained as a linear combination of weak classifiers, followed by a threshold. This is depicted in figure 2.

The classifier is designed in such a way that allows detection of faces at different scales. This is a more convenient technique than downsampling the image itself. A very popular algorithm to train the weak classifiers in order to obtain the strong one is represented by Ada-Boost. The algorithm is summarized in figure 3.

3.2 "CamShift" algorithm

In this section we give a brief review of this algorithm. It has been proposed as an extension of the mean-shift algorithm. Mean-shift works at one distribution scale, but is not suitable for another scale as the object moves towards and away from the camera. Small fixed-sized windows may get lost entirely for large object translation in the scene. Large fixed-sized windows may include distractors (other people or hands) and too much noise. "CamShift" uses continuously adaptive probability distributions (that is, distributions that are updated for each frame) while mean-shift is based on static distributions, which are not updated. Since "CamShift" does not maintain static distributions, spatial moments are used to iterate towards the mode of the distribution. This is in contrast to the conventional implementation of the mean-shift algorithm where target and candidate distributions are used to iterate towards the maximum increase in density using the ratio of the current (candidate) distribution over the target.

The probability distribution image may be determined using any method that associates a pixel value with a probability that the given pixel belongs to the target. A common method is known as histogram back-projection [19] which associates the pixel values in the image with the value of the corresponding



1. Input: Training examples (x_i, y_i), $i = 1..N$ with positive $(y_i = 1)$ and negative $(y_i = 0)$ examples.
2. Initalization: weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ with m negative and l positive examples
3. For t=1,...,T:
   (a) Normalize all weights
   (b) For each feature j train classifier h_j with error $\epsilon_j = \sum_i w_{t,i}|h_j(x_i - y_i)|$
   (c) Choose h_t with lowest error ϵ_t
   (d) Update weights: $w_{t+1,i} = w_{t,i}\beta_t^{1-e_i}$ with $e_i = \begin{cases} 0 & : & x_i \text{ correctly classified} \\ 1 & : & \text{otherwise} \end{cases}$
   and $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$
4. Final strong classifier: $h(x) = \begin{cases} 1 & : & \sum_{t=1}^{T} \alpha_t h_t(x) \geq 0.5 \sum_{t=1}^{T} \alpha_t \\ 0 & : & \text{otherwise} \end{cases}$
   with $\alpha_t = log(\frac{1}{\beta_t})$

Fig. 3. The Ada-Boost algoritm.

histogram bin. The back-projection of the target histogram with any consecutive frame generates a probability image where the value of each pixel characterizes probability that the input pixel belongs to the histogram that was used. Formally, the method can be described as follows. The ratio histogram, for a given color c is given by:

$$r_c(I, Q) = \min\left\{\frac{H(Q)}{H(I)}, 1\right\} \tag{7}$$

where $H(Q)$ represents the histogram of the model and $H(I)$ is the histogram of the image. It also represents the likelihood of a pixel $p \in I$ to belong to the target. The contribution of a subimage $I_p \subset I$ is given by:

$$\Pi_p(I, Q) = \sum_{q \in I_p} r_{I(q)}(I, Q) \tag{8}$$

Then the location of the target is given by:

$$\arg\max_p \Pi_p(I, Q) \tag{9}$$

4 Face tracking system

4.1 Dynamic model

In our problem about face tracking, a particle is represented by a combination of spatial and color information: $s_k = \{p_x, p_y, p_\sigma, H(p\Phi)\}$. The first three parameters represent the state vector $\{x_k = p_x, p_y, p_\sigma\}$ where (p_x, p_y) is the center of the particle and p_σ - the size of the particle. The last term $H(p\Phi)$ is the color histogram of the area 'under' the particle. We use the HSV color space, because it is less sensitive to illumination conditions. It separates the original RGB channels into H - hue (color), S - saturation and V - brightness. The color histogram is created by taking $1D$ histograms from the H (hue) channel in HSV space.

The tracker is initialized using the face detector based on Haar-like features. The face measurement is represented also as a feature vector $z_k = \{f_x, f_y, f_\sigma, H(f\Phi)\}$, where (f_x, f_y) is the center of the face, f_σ - the size of the face and $H(p\Phi)$ is the color histogram of the face. At each frame, we reaffirm the confidence in our prediction, by invoking the feature-based face detector (in case of failure, face evidence is searched using the color-based detector implementing the "CamShift" algorithm). For this reason, we have to define a likelihood measure for our observations.

4.2 Observation likelihood

In the particle filter, we have to update at each frame, the weight of the particles. This is achieved by computing the likelihood of the predicted parameters. Our observation model consists of a combination between color and geometrical information, thus we will use a scheme to fuse these cues into a global likelihood.

Color cue A common similarity measure between distributions is given by the Bhattacharyya distance [1]. When representing a color object through its histogram, we can express it as a discrete density, i.e $H(Q) = \{q^u\}_{u=1..n}$, where q_i represent the normalized bins of the histogram. Considering we have two discrete densities, $H(Q) = \{q^u\}_{u=1..n}$ and $H(R) = \{r^u\}_{u=1..n}$, we define the following coefficient:

$$\rho[Q,R] = \sum_{u=1}^{m} \sqrt{q^u r^u} \tag{10}$$

The larger ρ is, the more similar the color distributions are. Finally, the Bhattacharyya distance is defined as:

$$d = \sqrt{1 - \rho[Q,R]} \tag{11}$$

Based on this similarity measure, the likelihood between a candidate particle and the actual face model is given by:

$$p\left(z_k^c | x_k^i\right) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{d^2}{2\sigma^2}} \tag{12}$$

where the c superscript designates the color cue. The interpretation of the above equation is that large weights correspond to small Bhattacharyya values.

In our approach we don't use a previously learned histogram for skin color distribution. We prefer instead to use an adaptive strategy, extracting the initial histogram from the very first instance of the detected face from the video stream. The update of the target model is done with the following equation:

$$H(m_t^u) = (1-\alpha) H\left(m_{t-1}^u\right) + \alpha H(f\Phi^u) \tag{13}$$

for each histogram bin u of the model $H(M)$ and α represents the 'forgetting' term. As time passes, the relevance of the previously learned histograms tend to decrease and they are substituted by the most recent instances of face histograms.

Geometric cues The color cue is only one element to be taken into consideration when we appreciate the likelihood between the target and the model. Sometimes, color only cannot be sufficient. For this reason, we also make use of other elements, like size and position to the detected face. In other words, we want to be sure that not only the particles which are similar in color terms, but also has similar size and are situated in the neighborhood of the real face are the most confident candidates. The likelihood in size is expressed like:

$$p\left(z_k^s|x_k^i\right) = \frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{d^2}{2\sigma^2}} \tag{14}$$

where $d = p_\sigma - f_\sigma$. The s superscript designates the size cue.

Meanwhile, the likelihood for position is given by the euclidean distance between the center of the predicted face and the face center provided by face detector in the previous frame. Thus,

$$p\left(z_k^p|x_k^i\right) = \frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{d^2}{2\sigma^2}} \tag{15}$$

where $d = \sqrt{\left(p_x - f_x\right)^2 + \left(p_y - f_y\right)^2}$. The p superscript stays for the position cue.

Multiple cue integration Several methods for multiple cue integration have been proposed. In [21] they propose a novel architecture for adaptively integrating different cues in a self-organized manner. They argue that this approach is sustained with results from psychophysical experiments. In Democratic Integration different cues agree on a result, and each cue adapts toward the result agreed on. In particular, discordant cues are quickly suppressed and recalibrated, while cues having been consistent with the result in the recent past are given a higher weight in the future. The same idea is further developed in [18] in order to allow the tracker to deal with multiple objects. Their idea exploits two methodologies for the adaptation to different conditions: in the first place, the integration scheme can be changed and in the second place, the visual cues themselves can be adapted. Adapting the visual cues allows to adapt to different environmental changes directly. Changing the integration scheme reflects the underlying assumption that different cues are suitable for different conditions.

In our case, we propose a simpler solution, based on a linear combination of the computed likelihoods for each cue. Thus, the global likelihood will be given by:

$$p\left(z_k|x_k^i\right) = w_k^i = \beta_1 p\left(z_k^c|x_k^i\right) + \beta_2 p\left(z_k^s|x_k^i\right) + \beta_3 p\left(z_k^p|x_k^i\right) \tag{16}$$

where $\beta_j, j = 1, 2, 3$ are some constants. The final estimation position of the face is computed by maximizing the total likelihood:

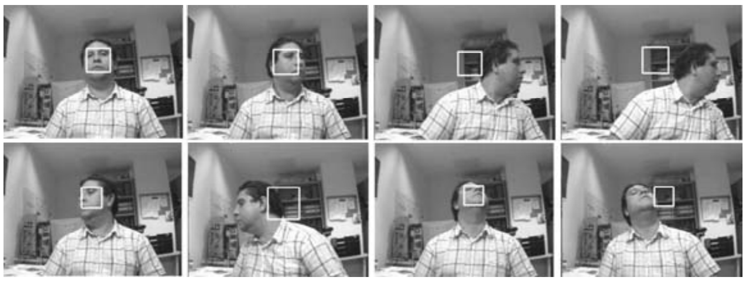


Fig. 4. Face tracking results obtained using only the particle filter. When the detector misses the face, the predicted face is shown at the last detected position.

$$x_{k\,\max} = \arg\max_{i} w_k^i \tag{17}$$

5 Experimental Results

The experimental results intend to show the improvements obtained by the introduction of the "CamShift" approach. In figure 4 we depict the results obtained using the particle filter method only. When the face evidence from the face detector is missing, the predicted position remains the same as in the last frame (we made the assumption that the proposed method is suitable for human-computer interaction, so in this case, the only possible occlusions are the ones due to head rotation - "self-occlusions"). Thus, the tracker is unable to cope with small head movements. On the other hand, when we introduced the "CamShift" approach, the tracker is able to robustly follow the face region, despite of the lack of face evidence provided by the Viola-Jones detector. This case is illustrated in figure 5.

In order to confirm the robustness of our approach, we tested it on several video sequences and compared the results with two others methods: particle filter and "CamShift" algorithm. For this purpose, we compared the results obtained from each of these algorithms with ground-truth values. As ground-truth we considered the hand-labelled position and size of the face in a video sequence consisting of almost 700 frames (we marked the upper-left and bottom-right corners of the face region). The video sequence considered for analysis was a more complex one, in which the person was moving without restriction (in front of a cluttered background) in the camera's field of view. In figure 6 we plotted the error both in terms of position (left graphic) and size (right graphic) between the tracked face and the manual picked one. The error in position was expressed as the euclidean distance between the center of the tracked-face and the center of the manual-labelled one. From the left graphic, it can be appreciated that

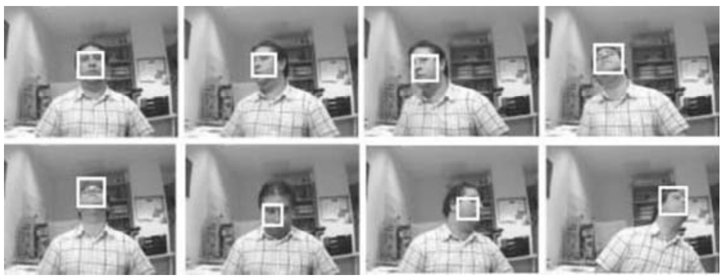


Fig. 5. Face tracking results obtained through the combination of the particle filter and the "CamShift". In this case, no face is missed. The improvements obtained by the use of "CamShift" are obvious.

when the person is moving in a transversal plane (the camera perceives the face from profile) with respect to the camera's optical axes (that corresponds to the interval between the frames 100-300 and again between the frames 400-480), the particle filter alone is getting stuck, because no evidence from the face detector is provided. That's the interpretation of the high peaks in the graph. However, "CamShift" and our approach perform well. Another situation can be found when the person passes his hand in front of the face (between the frames 530-580). This situation can be very well observed on the both graphs. Our system, despite its brief distraction, its able to recover after this occlusion. The particle filter is the most robust one in this case, because is tracking the real position of the face. On the contrary, "CamShift" has lost the track forever.

6 Conclusions and Future Work

In this paper we proposed a solution for multiple cue integration in the framework of particle filters, with application to face tracking. We saw that the use of a single cue (either color or shape) is not sufficient to cope with dynamic environments. Thus, our method will offer an increased robustness against changes in appearance, scale and pose. The successful framework of particle filter was combined with evidence about face presence obtained from a model-based approach. We tested our method on several video sequences and the results obtained are highly encouraging. In the near future, we plan to adapt our solution and implement it on an AIBO robot. Its primary task will be thus to use facial information as a contextual information for human-presence. For the beginning, the contextual information will be limited only to assess the user presence and distance to the robot. Later on, we will expand the contextual information, by including information about person's gender and identity. Regarding person identity, we

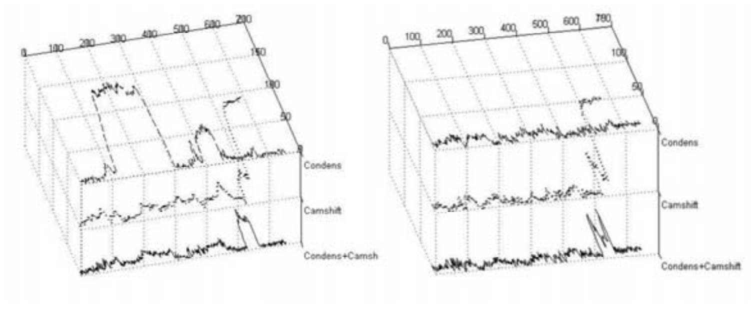


Fig. 6. Comparison between our approach and two others: particle filter and "CamShift". The plots represent the error in distance (left) and size (right) between the face reported by each of the three methods and ground truth. 'Condens' label states for particle filter.

are currently investigating a novel technique known as "unsupervised face recognition". That's it, the robot starts with an empty database, and will gradually 'get to now' new people.

Acknowledgements

This work is supported by MCYT Grant TIC2003-00654, Ministerio de Ciencia y Tecnología, Spain. Bogdan Raducanu is supported by the Ramon y Cajal research program, Ministerio de Educación y Ciencia, Spain.

References

1. Aherne, F., Thacker, N., Rockett, P.: The Bhattacharya Metric as an Absolute Similarity Measure for Frequency Coded Data. Kybernetyka **32** (1997) 1-7
2. Arulampalam, M.S., Maskell, S., Gordon, N., Clapp, T.: A Tutorial on Particle Filters for Online Nonlinear/Non-gaussian Bayesian Tracking. IEEE Transactions on Signal Processing **50** (2002) 174–188
3. Bradsky, G.R.: Computer Vision Face Tracking for Use in a Perceptual User Interface. Technical Report 2001.2, Microcomputer Research Lab, Intel Corporation (2001)
4. Dey, A.K.: Understanding and Using Context. Personal and Ubiquitous Computing Journal **5** (2001) 4-7
5. Doucet, A., de Freitas, J.F.D., Gordons, N.J.: Sequential Monte Carlo Methods in Practice. Springer (2001)
6. Gatica-Perez, D., Lathoud, G., McCowan, I., Odobez, J.-M., Moore, D.: Audio-Visual Speaker Tracking with Importance Particle Filters. Proceedings of IEEE International Conference on Image Processing (ICIP), Barcelona, Spain (2003) pp.N/A

7. Isard, M., Blake, A.: CONDENSATION - Conditional Density Propagation for Visual Tracking. International Journal of Computer Vision **29** (1998) 5-28
8. Jazwinsky, A.M.: Stochastic Processes and Filtering Theory. Academic Press (1970)
9. Julier, S.J., Uhlmann, J.K.: A New Extension of the Kalman Filter to Nonlinear Systems. Proceedings of SPIE on Signal Processing, Sensor Fusion and Target Recognition VI **3068** (1997) 182-193
10. Maybeck, P.: Stochastic Models, Estimation and Control (I). Academic Press (1979)
11. McKenna, S., Gong, S.: Tracking Faces. Proceedings of 2nd International Conference on Automatic Face and Gesture Recognition (AFGR), Vermont, USA (1996) 271-276
12. Nummiaro, K., Meier, E.K., Gool, L.v.: An Adaptive Color-Based Particle Filter. Image and Vision Computing **21** (2003) 99-110
13. Pantrigo, J.J., Montemayor, A.S., Cabido, R.: Scatter Search Particle Filter for 2D Real-Time Hands and Face Tracking. Proceedings of International Conference on Image Analysis and Processing, Lecture Notes in Computer Science, Springer, Berlin Heidelberg, **3617** (2005) 953-960
14. Peihua, L., Chaumette, F.: Image Cues Fusion for Contour Tracking Based on Particle Filter. Proceedings of International Workshop on Articulated Motion and Deformable Objects. Lecture Notes in Computer Science, Springer, New York, **3332** (2004) 99-107
15. Pérez, P., Hue, C., Vermaak, J., Cangnet, M.: Color-based Probabilistic Tracking. Proceedings of the 7th European Conference on Computer Vision, Lecture Notes in Computer Science, Springer, New York, **2350** (2002) 661-675
16. Pérez, P., Vermaak, J., Blake, A.: Data Fusion for Visual Tracking with Particles. Proceedings of IEEE **92** (2004) 495-513
17. Shan, C., Wei, Y., Tan, Y.T., Ojardias, F.: Real Time Hand Tracking by Combining Particle Filtering and Mean Shift. Proceedings of the 6th Int. Conf. on Automatic Face and Gesture Recognition (AFGR), Seoul, Korea (2004) pp.N/A
18. Spengler, M., Schiele, N.: Towards Robust Multi-Cue Integration for Visual Tracking. Proceedings of International Conference on Computer Vision Systems, Lecture Notes in Computer Science, Springer, Berlin Heidelberg, **2095** (2001) 93-106
19. Swain, M., Ballard, D.: Color Indexing. International Journal of Computer Vision **7** (1991) 11–32
20. Thrun, S.: Particle Filters in Robotics. Proceedings of Uncertainty in AI (2002) pp.N/A
21. Triesch, J., Malsburg, C.v.d.: Democratic Integration: Self-Organized Integration of Adaptive Cues. Neural Computation **13** (2001) 2049-2074
22. Viola, P., Jones, M.J.: Robust Real-Time Face Detection. International Journal of Computer Vision, **57** (2004) 137-154
23. Weiser, M.: The Computer for the Twenty-First Century. Scientific American (1991) 94-10
24. Welch, G., Bishop, G.: An Introduction to Kalman filter. Technical Report TR95-041, Dept. of Comp. Science, Univ. of North Carolina (2002)
25. Williams, O., Blake, A., Cipolla, R.: Sparse Bayesian Learning for Efficient Visual Tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence **27** (2005) 1292-1304
26. Zajdel, W., Zivkovic, Z.,Kröse, B.J.A.: Keeping Track of Humans: Have I Seen This Person Before?. Proceedings of International Conference on Robotics and Automation (ICRA), Barcelona, Spain (2005) 2093-2098