New Models For Real-Time Tracking Using Particle Filtering

Ng Ka Ki and Edward J. Delp
Video and Image Processing Laboratories (VIPER)
School of Electrical and Computer Engineering
Purdue University
West Lafayette, Indiana USA

ABSTRACT

This paper presents new methods for efficient object tracking in video sequences using multiple features and particle filtering. A histogram-based framework is used to describe the features. Histograms are useful because have the property that they allow changes in the object appearance while the histograms remain the same. Particle filtering is used because it is very robust for non-linear and non-Gaussian dynamic state estimation problems and performs well when clutter and occlusions are present. Color histogram based particle filtering is the most common method used for object tracking. However, a single feature tracker loses track easily and can track the wrong object. One popular remedy for this problem is using multiple features. It has been shown that using multiple features for tracking provides more accurate results while increasing the computational complexity. In this paper we address these problems by describing an efficient method for histogram computation. For better tracking performance we also introduce a new observation likelihood model with dynamic parameter setting. Experiments show our proposed method is more accurate and more efficient then the traditional color histogram based particle filtering.

Keywords: object tracking, particle filtering, color features, edge features, motion estimation, histograms

1. INTRODUCTION

Visual object tracking is an important step in many applications such as human-computer interaction, medical imaging, video compression, video surveillance, and gesture recognition. In recent years, tracking algorithms based on particle filtering has been extensively studied [1]. Experimental results show that particle filtering outperforms other conventional state estimation method such as Kalman filter for non-linear and non-Gaussian systems. Particle filters generalize the traditional Kalman filtering methods which provide the optimal solution only when the models have linear functions and Gaussian noise [1].

Many algorithms have been proposed for object tracking that rely on a single feature, which is chosen according to the application. In [2], a color feature is used in conjunction with a particle filter which results in tracking performance that is better than the color based mean-shift tracker introduced in [3]. However, this method requires more computational time. Color histograms have been widely used for tracking problems [2–4] because they are robust to partial occlusion and are rotation and scale invariant. They have limitations in areas where the background has a similar color as the target object in that the tracker can be confused and lose track of the object. Color histograms also have poor performance when the illumination varies.

Other features have been proposed for tracking including shape [5] and gradient direction [6] or a combination of shape and color [7]. A single feature does not provide enough information about the object being tracked and hence using multiple features can provide more information about the object. For our algorithm described in this paper we use color and also introduce an edge feature to the observation likelihood model. It is useful for modeling the structure of the object to be tracked. The edges are described using a histogram based on the estimated strength and orientation of the edges.

Each iteration of our particle filtering based tracking system consists of three steps: state transition using the motion model, computation of particle weights (likelihood estimation) by observing the features using the

observation likelihood model, and resampling. Generally likelihood estimation requires the most computational cost among all these operations [8]. For example, it has been shown that computation of the color histograms at each iteration is a major bottleneck of the color-based particle filter [4].

Due to the high computational cost of likelihood estimation, tracking with multiple features is more computational expensive than single feature tracking. To deal with this, a more efficient algorithm for the computation of the color histogram is proposed. The traditional way to compute the color histogram is to construct the color histogram using every pixel in the tracker. In our work, we use a subset of the pixels inside the tracker e.g. a subset of the pixels that forms a checkerboard pattern. It is shown in our experiments that this approach works well and takes advantage of the spatial redundancy of natural images in that their neighboring pixels are highly correlated.

The observation likelihood function plays an important role in tracking performance using particle filtering. A particle filter tracks multiple hypotheses simultaneously and each of them is weighted according to the observation likelihood function. There are two major reasons for why the observation likelihood function is important. One reason is that this function determines the weights of the particles and the weights determine how the particles are resampled. Another reason is that the estimated state is the weighted average of all particles, hence it affects the estimates directly. Besides the color and edge features we use to compute the Bhattacharyya distance in the observation likelihood model, there is another parameter (σ in Equation 7 as described later) that controls how a particle is weighted. In this paper we present an algorithm to choose this parameter dynamically.

We will also present experimental results for the new observation likelihood functions and illustrate their impact on tracking performance.

2. AN INTRODUCTION TO PARTICLE FILTERING

2.1. Filtering

The "filtering" problem is the process of estimating a system's current state which is hidden, based on past and current observations [9]. This is represented by the probability density function $p(x_t|x_{t-1}, z_{0:t})$. Depending on the application, the state at time t, x_t and observation at time t, z_t can be different entities. For the case of visual tracking, the state can be position, velocity, orientation, or scale of a object, and the observation can be the color histogram, edge orientation histogram, or contours of the image data.

In general, the object tracking problem can be modeled by the state-space representation

1. State Equation:

$$x_t = f_t(x_{t-1}, v_t) \tag{1}$$

2. Meaurement Equation:

$$z_t = h_t(x_t, n_t) \tag{2}$$

where f_t is the state equation (transition equation) at time t, and h_t is the measurement equation (observation equation) at time t. These are non-linear and time varying in general. The system noise and the measurement noise are denoted by v_t and n_t respectively.

2.2. Bayesian Recursive Filtering

The optimal estimator $\hat{x}_t$ (optimal in the minimum variance sense) of the state x_t is provided by the conditional expectation $E[x_t|z_{1:t}]$. To obtain $E[x_t|z_{1:t}]$, we need to estimate the posterior probability density function $p(x_t|z_{1:t})$ at each time instant t, and this estimation can be done by using Bayesian filtering.

The goal of Bayesian filtering is to construct the posterior probability density function $p(x_t|z_{1:t})$, of the state x_t at time t given all available observations $z_{1:t}$ up to time t. It is assumed that the initial probability density function of the state (prior probability density function) p_{x0} is known. Then the posterior probability density function $p(x_t|z_{1:t})$ can be obtained recursively with a two-stage process: prediction and update.

Recursive filtering means that the received or observed data can be processed sequentially rather than as a batch:
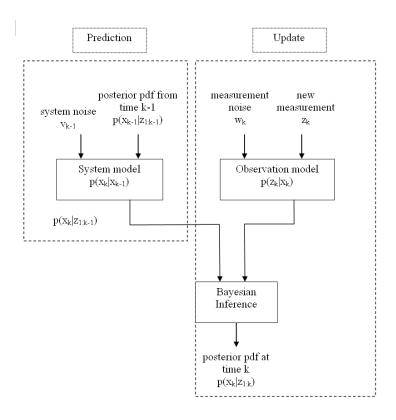
Figure 1. One iteration of the prediction and update. The goal is to find the posterior probability density funstion at time k.

- Prediction: Assume that the posterior probability density function $p(x_{t-1}|z_{1:t-1})$ is available, the prediction step yields $p(x_t|z_{1:t-1})$ using the Chapman-Kolmogorov integral equation:

$$p(x_t|z_{1:t-1}) = \int p(x_t|x_{t-1})p(x_{t-1}|z_{1:t-1})dx_{t-1} \tag{3}$$

where x_t is the state at time t and $z_{1:t-1}$ is the observation sequence from time 1 to time $t-1$. The probability density function $p(x_t|x_{t-1})$ expresses the state model (transition model).

- Update: At time t, the observation z_t is available, the prior probability density function $p(x_t|z_{1:t-1})$ is updated via Bayes' rule:

$$p(x_t|z_{1:t}) = \frac{p(z_t|x_t)p(x_t|z_{1:t-1})}{\int p(z_t|x_t)p(x_t|z_{1:t-1})dx_t} \tag{4}$$

where x_t is the state at time t and $z_{1:t}$ is the observation sequence from time 1 to time t. The probability density function $p(z_t|x_t)$ expresses the likelihood function that describes the measurement model. One iteration of this recursive process is shown in Figure 1.

In general Equations (3) and (4) cannot be evaluated in closed form. In some special cases, such as linear and Gaussian state-space models which can be analyzed using a Kalman filter, and hidden finite-state space Markov chains closed form solutions are available. To solve this problem particle filtering is used to approximate Bayesian filtering when a closed-form solution cannot be obtained (i.e. when the transition and observation models are non-linear and non-Gaussian).

2.3. Particle Filtering

The key idea of particle filtering is to represent the posterior probability density function by a set of discrete samples know as particles. A sample is also referred to as a particle because of its discrete nature and its discrete representation by the probability density function. Each particle represents a hypothesis of the state and it is randomly drawn from the prior density. After a particle is drawn, it is then propagated according to the transition model. Each propagated particle is verified by a weight assignment using the likelihood model. The weight characterizes the quality of a specific particle. A large weight will be assigned to a good particle, and a small weight will be assigned to a bad particle.

The posterior probability density function is constructed recursively by the set of weighted random samples $\{x_t^{(i)}, w_t^{(i)}; i = 1, \ldots, N\}$ where N is the total number of particles. At each time t, the particle filtering algorithm repeats a two-stage procedure: prediction and upate:

- Prediction: Each particle $x_t^{(i)}$ evolves independently according to the state model (1), including the addition of random noise in order to simulate the unknown disturbance. The step yields an approximation of the prior probability density function:

$$p(x_t) \approx \frac{1}{N} \sum_{i=1}^{N} \delta(x_t - x_t^{(i)}) \tag{5}$$

- Update: Each particle's weight is evaluated based on the latest measurement according to the measurement model (likelihood model) (2). The posterior probability density function at time t in the form of a discrete approximation can be written as

$$p(x_t|z_{1:t}) \approx \sum_{i=1}^{N} w_t^{(i)} \delta(x_t - x_t^{(i)}) \tag{6}$$

where

$$w_t^{(i)} = \frac{\mathcal{L}(z_t|x_t^{(i)}) p(x_t^{(i)}|x_{t-1}^{(i)})}{q(x_t^{(i)}|x_{t-1,z_t})} \tag{7}$$

and weight set satisfies

$$\sum_{i=1}^{N} w_t^{(i)} = 1. \tag{8}$$

3. OBSERVATION LIKELIHOOD MODELS

3.1. Color Features

This section describes how we model color features of an elliptical region R, where R can be a region surrounding the object to be tracked or region surrounding one of the hypothetical regions. A color histogram is commonly used for object tracking because they are robust to partial occlusion and are rotation and scale invariant. They are also flexible in the types of object that they can be used to track, including rigid (e.g. vehicles) and non-rigid object (e.g. people).

The color distribution is expressed by a m-bin histogram, whose components are normalized so that its sum of all bins equals one. For a region R in an image, given a set of n samples in R, denoted by $\mathbf{S} = \{x_i, i = 1, 2, \ldots, n\} \in R$, the m-bin color histogram $H(R) = \{h_j, j = 1, 2, \ldots, m\}$ can be obtained by assigning each pixel, x_i to a bin, by the following equation:

$$h_j = \frac{1}{n} \sum_{(x_i) \in S} \delta_j[b(x_i)] \tag{9}$$

where $b(x_i)$ is the bin index where the color component at x_i falls into, and δ is the Kronecker delta function.

In our experiment, a $8 \times 8 \times 4$-bin histogram is constructed for each region R in HSV color space. HSV color space is used to reduce the sensitivity to illumination [2].

3.2. Edge Features

Color is a powerful feature for target tracking, using only color features is sufficient for most tracking tasks, including cases such as occlusion and rotation. However, problems arise when the target object and background have similar color distribution. It may be difficult to distinguish the object from background. To solve this problem, multiple-feature tracking is used because it provides more information about the object. Edge features are introduced as a complementary feature to our observation model because it is useful for modeling the structure of the object to be tracked. The edge features are used with a histogram based on the estimated strength and orientation of the edges.

Edges are detected using a simple horizontal and vertical Sobel operator K_x and K_y on the grayscale image. The gradient component in each orientation G_x and G_y has magnitude given by

$$G_x(x,y) = K_x * I(x,y), G_y(x,y) = K_y * I(x,y) \tag{10}$$

where

$$K_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}, K_y = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix} \tag{11}$$

The strength and the orientation of the edges are

1. Strength
$$S(x,y) = \sqrt{G_x^2(x,y) + G_y^2(x,y)} \tag{12}$$

2. Orientation
$$\theta = arctan(G_y(x,y)/G_x(x,y)) \tag{13}$$

A predefined threshold is used to remove noise, any edge orientation with strength less then the threshold is discarded from the edge orientation histogram. The threshold is set to 100 in our experiments, hence:

$$\theta'(x,y) = \begin{cases} \theta(x,y) & \text{if } S(x,y) > threshold \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

Variations on this scheme have included the Roberts, Prewitt [10] and Sobel [11] operators.

For a region R in an image, given a set of n pixels in R, denoted by $\mathbf{S} = \{(x_i, y_i), i = 1, 2, \ldots, n\} \in R$, the k-bin edge histogram $E(R) = \{e_j, j = 1, 2, \ldots, k\}$ is computed by assigning each pixel, (x_i, y_i) to a bin, by the following equation:

$$e_j = \frac{1}{k} \sum_{(x_i, y_i) \in S} \delta_j [b(x_i, y_i)] \tag{15}$$

where $b(x_i, y_i)$ is the bin where $\theta'(x_i, y_i)$ falls into.

3.3. Weighted Histograms

The probability density function can be estimated using the sample set $\mathbf{S} = \{x_i, i = 1, 2, \ldots, n\} \in R$ by kernel density estimation. Kernel density estimation is one of the most popular non-parametric methods for probability density function estimation [3, 12].

For the point x, a density estimator with kernel $K(x)$ with bandwidth h is defined as

$$p_k(x) = \frac{1}{nh^d} \sum_{i=0}^{n} K\left(\frac{x - x_i}{h}\right) \tag{16}$$

where d is the dimension of the sample x_i, and the kernel K we use is the Epanechnikov kernel K_e, and it is defined as

$$K_e(x) = \begin{cases} \frac{1}{2} c_d^{-1}(d+2)(1 - \|x\|^2) & \text{if } \|x\|^2 < 0 \\ 0 & \text{otherwise} \end{cases} \tag{17}$$

where $d(x_i)$ is the distance from x_i to the center of the ellipse and c_d is the volume of the unit d-dimensional sphere. The Epanechnikov kernel is a convex and monotonic decreasing function.

The classical way of estimating the probability density function is to use every pixel in the region R as a sample for the histogram (color or edge histogram for our experiments) followed by using a kernel on that sample. That is, the set of samples, $\mathbf{S} = \{x_i, i = 1, 2, \ldots, n\} \in R$ has x_i's equal all pixels in the region. The probability of the quantized color vector u in the object model is given by

$$q_u(x) = C \sum_{i=1}^{n} K(\|d(x_i)\|^2)\delta[b(x_i) - u] \tag{18}$$

where C is the normalization factor

$$C = \frac{1}{\displaystyle\sum_{i=1}^{n} K(\|d(x_i)\|^2)} \tag{19}$$

that ensures $\sum q_u(x) = 1$

The observed information will be unstable at peripheral points in the region and at its center when the object is undergoing partial occlusions and background changes. To avoid this, assigning larger weights for the central points of the region is achieved by using the Epanechnikov kernel $K(x)$ defined earlier in Equation (17) [2, 13, 14].

Comparing the estimated probability density function in Equation (18) to the histogram computation in Equation (9), they are very similar except that the estimated probability density function is a weighted histogram.

3.4. Distance Measure

A model histogram is the weighted color or edge orientation histogram (described in section 3.1 and 3.2) of the object to be tracked. The model histogram is constructed during the initialization (first frame at time $t = 1$) of the system. In subsequent frames, at every time t, there are N particles that represent N hypothetical states need to be evaluated. The observation likelihood model is used to assign a weight associated to a specific particle (new observation) depending on how similar the object histogram q and the histogram $p(s_t)$ of the region described by the ith particle $s_t^{(i)}$ are.

To evaluate the similarity between the model histogram, q and the particle's histogram, $p(s_t^{(i)})$, where $s_t^{(i)}$ is the ith particle at time t, we employ the Bhattacharyya coefficient ρ.

$$\rho[p(s_t), q] = \sum_{u=1}^{m} \sqrt{p_u(s_t)q_u} \tag{20}$$

Figure 2. The pixels marked by a cross are used to construct the histogram. Left: One half of the pixels are used. Right: One quarter of the pixels are used.

where u is the histogram bin index. The larger ρ is, the more similar the two distributions are. For two identical normalized histograms we obtain $\rho = 1$, indicating a perfect match. To quantify the distance between two distributions, the distance d is defined as

$$d = \sqrt{1 - \rho[p, q]} \tag{21}$$

which is known as the Bhattacharyya distance. The observation likelihood function uses this distance to assign a weight to each particle. A sample with small Bhattacharyya distance corresponds to a large weight; similarly, a sample with large Bhattacharyya distance corresponds to a small weight.

3.5. A New Method For Efficient Histogram Computation

In tracking using classical color histogram based particle filtering, the construction of the histograms are always a bottleneck [4, 8]. The traditional way to compute the color histogram is to construct the color histogram using every pixel in the tracker. An efficient method for histogram construction is proposed here. In this method, we use a subset of the pixels inside the tracker for the color histogram construction e.g. subset of pixels that forms a checkerboard pattern. This is shown in Figure 2. Only the pixels marked by crosses are used to construct the histogram. It is shown in our experiments that this approach for histogram construction performs as well as the traditional histogram computation. This method takes advantage of the spatial redundancy of natural images since neighboring pixels are highly correlated.

3.6. Observation Likelihood Function

A particle filter tracks multiple hypotheses simultaneously and each hypothesis is represented by a particle. Each particle is weighted according to the observation likelihood function. If a particle has features similar to the features of the target, this particle has a smaller distance from the object model, and it will be assigned with a larger weight according to the observation model.

The observation likelihood function is very important for the tracking performance of particle filters. One reason is that the observation likelihood function determines the weights of the particles and they determine how the particles are resampled. Resampling is used to decrease the number of low-weighted particles and increase the number of particles with high weights. Another reason is that the estimated state is the weighted average of all particles, hence this function affects the estimates directly.

The observation likelihood function is expressed as $\mathcal{L}(z_t|x_t^{(i)})$ in Equation 7. In the case of standard implementation, the importance density $q(x_t^{(i)}|x_{t-1,z_t})$ is chosen to be the dynamic transition model, $p(x_t^{(i)}|x_{t-1}^{(i)})$. In this case, the observation likelihood function is the only function that contributes to a particle's weight.

A classical observation likelihood function that is typically used is

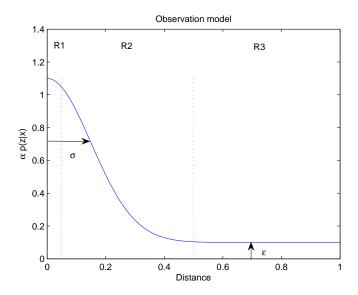$$\mathcal{L}(z|x) \propto e^{-\frac{d^2}{2\sigma^2}} \tag{22}$$

Figure 3. Observation likelihood function with $\epsilon = 0.1$ and $\sigma = 0.15$.

The only parameter in this observation likelihood function is σ. This parameter determines the steepness of the function, i.e. how fast the function will decrease when the distance is large (bad particles). Another likelihood function was proposed in [6]:

$$\mathcal{L}(z|x) \propto e^{-\frac{d^2}{2\sigma^2}} + \epsilon \tag{23}$$

where σ and ϵ are the model parameters. σ determines the steepness of the function, and ϵ weights all the bad particles equally, this parameter also prevents the particles from getting stuck at the local maxima when the object is lost. To improve the ability to recover from a lost object, ϵ should be small but non-zero. The model is shown in Figure 3, with $\epsilon = 0.1$ and $\sigma = 0.15$. For the good particles with distances close to zero (R1 in the figure), the model introduces some tolerance for the difference due to distortion. So particles fall onto this region are assigned to very high weights. The plummet in R2 discards the bad samples when there are good particles present in R1. The worst particles that fall onto R3 are weighted equally and are determined by ϵ.

In [6], the importance of choosing the right model parameters of the observation likelihood function is investigated. The effects on the tracking performance when the model parameters are too large or too small are described. However, complete details about how to determine the model parameters were not presented. In this paper, we combine the work in [10] for setting the dynamic parameter of the observation likelihood function with the observation likelihood function proposed in [6].

From equation (22) we solve for σ

$$\sigma = \frac{\sqrt{2}}{2}\sqrt{\frac{-d_{min}^2}{logL}} \tag{24}$$

Where d_{min} is the distance of the most reliable particle (the particle with the minimum distance), and $log(L) = -1$.

An adaptive σ always ensures the best particles are assigned with highest weights and the worst ones are assigned with ϵ. For example, for the case of a fixed $\sigma = 0.15$ is used, as shown in Figure 3, if all particles have distances greater than 0.4, then all particles fall onto R3 and they are all assigned with weights equal σ. As a result the likelihood function will not differentiate the difference between particle with distance equals 0.4 and particle with distance equals 1. However, by adaptively setting σ according to d_{min}, there will be at least one particle that falls onto R1, and the worst particles will still fall onto R3, making the observation likelihood function more discriminative.

	All Pixels	Half of the Pixels	One Quarter
computational time(sec)	0.1304	0.0498	0.0252
ρ	1	0.9998	0.9993
distance	0	0.0157	0.0258
$\mathcal{L}(z|x)$	1	0.9946	0.9853

Table 1. Similarity between histograms with $\sigma = 0.15$.

4. EXPERIMENTAL RESULTS

In order to test the effect of using only a subset of pixels in color histogram construction, we used a still image from a video sequence that has an elliptical region with a human body is enclosed. A color histogram constructed using all pixels in the ellipse is computed and is used as a control for comparison. Two histograms using only one half and one quarter of the pixels are constructed and are compared with the control histogram.

The comparison is done in terms of the Bhattacharyya coefficient ρ using Equation (20), Bhattacharyya distance using Equation (21), the observation likelihood using Equation (22), and computational time.

The results is shown in Table 1. For the case with only half of the pixels being used, the computational complexity was decreased by about 50%. With ρ and distance between the histogram with all pixels being used and histogram with only one half of the pixels being used were very close to 1 and 0 respectively. For the case with only one quarter of the pixels being used, the computational complexity was decreased by about 25%, with ρ and distance very close to 1 and 0 as well.

We also performed video object tracking using our efficient histogram computation, the experiment results show that our algorithm works successfully. Figure 4 shows some tracked results of a person walking in a low-illumination room. The size of the input frames is 352×240, the size of the ellipse is about 10×30. The blue dot represents the estimated position of the object, and the red ellipse represents the size of the object. Figure 5 shows the results of a tracked bus. The size of the input frames is 384×288.

5. IMPLEMENTATION ON NOKIA N810

Using our proposed algorithm, we implemented the tracking system on a hand held device, the Nokia N810 Internet Tablet. The Nokia N810 is a portable internet tablet with a 400 MHz ARM processor. It runs a Linux-base operating system.The system executes the object tracking at about 1 frame/second. A picture of the internet tablet is shown in Figure 6. We are in the process of optimizing the implementation of this device and plan on testing it in the field for real-time tracking of people in surveillance videos.

6. CONCLUSIONS

In this paper we introduced two new concepts for using particle filtering for tracking. An efficient algorithm for computation of the color histograms was described and a new observation likelihood function was presented. The new approaches decreased the computational complexity by 50% when the checkerboard pattern was used for histogram construction. The method was more robust to cases such as occlusion and illumination change when multiple features and dynamic parameter setting were used.

REFERENCES

1. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking," *IEEE Transactions of Signal Processing*, 2002, pp. 174–188.
2. K. Nummiaro, E. Koller-Meier, and L. V. Gool, "Color features for tracking non-rigid objects," *Special Issue on Visual Surveillance Chinese Journal of Automation*, 2003, pp. 345–355.
3. D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," *Computer Vision and Pattern Recognition*, vol. 2, 2000, pp. 142–149.
4. P. Pérez, C. Hue, J. Vermaak, and M. Gangnet, "Color-based probabilistic tracking," *Proceedings of the 7th European Conference on Computer Vision-Part I.* London, UK: Springer-Verlag, 2002, pp. 661–675.
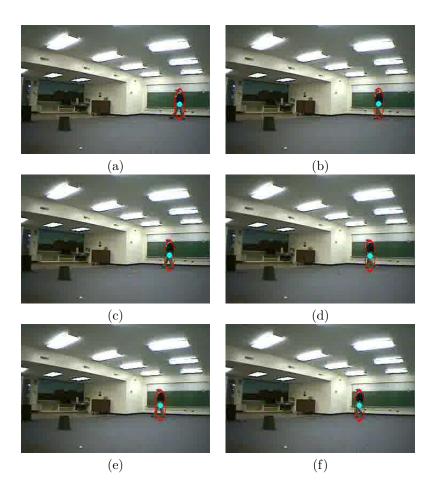
Figure 4. Tracking results of a low-illumination sequence with a person walking, the tracked object is circled: frame number (a)45 (b)55 (c)65 (d)75 (e)85 (f)95.

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

6. J. Lichtenauer, M. Reinders, and E. Hendriks, "Influence of the observation likelihood function on particle filtering performance in tracking applications," *IEEE International Conference on Automatic Face and Gesture Recognition*, 2005, pp. 767–772.

7. T. Xiong and C. Debrunner, "Stochastic car tracking with line- and color-based features," *IEEE Transactions on Intelligent Transportation Systems*, vol. 5, no. 4, Dec 2004, pp. 324–328.

8. H. Sugano and R. Miyamoto, "A real-time object recognition system on cell broadband engine," *Proceedings of Pacific-Rim Symposium on Image and Video Technology*, 2007, pp. 932–943.

9. Y. Rui and Y. Chen, "Better proposal distributions: object tracking using unscented particle filter," *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, 2001, pp. 786–793.

10. P. Brasnett, L. Mihaylova, D. Bull, and N. Canagarajah, "Sequential monte carlo tracking by fusing multiple cues in video sequences," *Image and Vision Computing*, vol. 25, no. 8, 2007, pp. 1217–1227.

11. C. Yang, R. Duraiswami, and L. Davis, "Fast multiple object tracking via a hierarchical particle filter," *International Conference on Computer Vision*, vol. 1, 2005, pp. 212–219.

12. Q. Wang and J. Liu, "Visual tracking using the kernel based particle filter and color distribution," *International Conference on Neural Networks and Brain*, vol. 3, 2005, pp. 1730–1733.
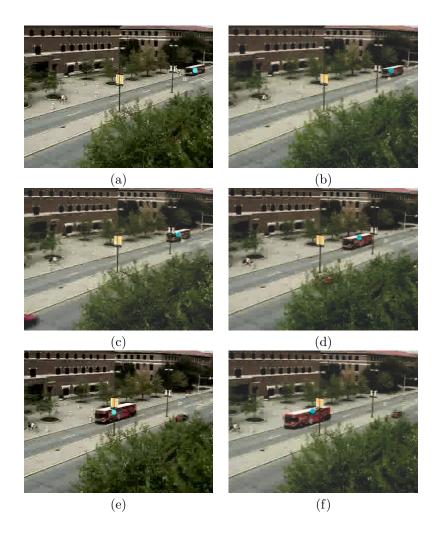
Figure 5. Real-time vehicle tracking results of a bus; frame number (a)30 (b)40 (c)50 (d)75 (e)102 (f)110.

Figure 6. Nokia N810 Internet Tablet mobile device.

13. D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 5, pp. 564–575, 2003.
14. K. Deguchi, O. Kawanaka, and T. Okatani, "Object tracking by the mean-shift of regional color distribution combined with the particle-filter algorithm," *Proceedings of the Pattern Recognition, 17th International Conference*, vol. 3.   Washington, DC, USA: IEEE Computer Society, 2004, pp. 506–509.