4

Object Tracking Based on Color Information Employing Particle Filter Algorithm

Budi Sugandi, Hyoungseop Kim, Joo Kooi Tan and Seiji Ishikawa
Graduate School of Engineering, Kyushu Institute of Technology
Japan

1. Introduction

The increasing interest in the object tracking is motivated by a huge number of promising applications that can now be tackled in real-time applications. These applications include performance analysis, surveillance, video-indexing, smart interfaces, teleconferencing and video compression and so on. However, object tracking can be extremely complex and time-consuming especially when it is done in outdoor environments. Here, we can mention some problems of object tracking in outdoor environments such as fake-motion background, illumination changes, shadows and presence of clutter. A variety of tracking algorithms have been proposed and implemented to overcome these difficulties. They can be roughly classified into two categories: deterministic methods and stochastic methods.
Deterministic methods typically track the object by performing an iterative search for a similarity between the template image and the current one. The algorithms which utilize the deterministic method are background subtraction (Heikkila & Silven, 1999; Stauffer & Grimson, 1999; McIvor, 2000; Liu et al., 2001), inter-frame difference (Lipton et al., 1998; Collins et al., 2000), optical flow (Meyer et al., 1998), skin color extraction (Cho et al., 2001; Phung et al., 2003) and so on. On the other hand, the stochastic methods use the state space to model the underlying dynamics of the tracking system such as Kalman filter (Broida & Chellappa, 1986; Arulampalam et al., 2002) and particle filter (Isard & Black, 1998; Kitagawa, 1996; Gordon et al., 1993; Ristic et al., 2004). Kalman filter is a common approach for dealing with target tracking in the probabilistic framework. In a linear-Gaussian model with linear measurement, there is always only one mode in the posterior probability density function (pdf). The Kalman filter can be used to propagate and update the mean and covariance of the distribution of this model (Arulampalam et al., 2002). But it cannot resolve the tracking problem when the model is nonlinear and non-Gaussian (Tanizaki, 1987). For nonlinear or non-Gaussian problems, it is impossible to evaluate the distributions analytically and many algorithms have been proposed to approximate them. The extended Kalman filter can deal to this problem, but still has a problem when the nonlinearity and non-Gaussian cannot be approximated accurately. To overcome those problems, particle filter has been introduced by many researchers and become popular algorithm to estimate the problem of nonlinear and non-Gaussian estimation framework. The particle filter, also known as sequential Monte Carlo (Kitagawa, 1996), is the most popular approach which recursively constructs the posterior pdf of the state space using Monte Carlo integration. It approximates a posterior probability density of the state such as the object position by using samples or

particles. The probability distribution of the state of the tracked object is approximated by a set of particles, which each state is denoted as the hypothetical state of the tracked object and its weight. It has been developed in the computer vision community and applied to tracking problem and is also known as the Condensation algorithm (Isard & Black, 1998). For another particle filter, Bayesian bootstrap filter was introduced (Gordon et al., 1993).
Although particle filters have been widely used in recent years, they have important drawbacks (King & Forsyth, 2000) such as samples are spread around several modes pointing out the different hypotheses in the state space. Many improvements have been introduced, but there is still much ground to cover. Different approaches have been taken in order to overcome these problems. (Nummiaro et al., 2002) used a particle filter based on color histograms features. Histograms are robust to partial occlusions and rotations but no shape analysis is taken into account. Moreover, no multiple-target tracking is considered and complete occlusions are not handled. (Perez et al., 2002) proposed also a particle filter based on color histogram. They introduced interesting extension in multiple-part modeling, incorporation of background information and multiple targets tracking. Nevertheless, it requires an extremely large number of samples and no appearance model updating is performed, what usually leads to target loss in dynamic scenes. (Comaniciu et al., 2000) proposed to use mean shift in order to track non-rigid object. His work has real time capabilities. However, the problem for object tracking with color occurs when the region around the object is cluttered and illumination is change. In this way, a color feature based tracking does not provide reliable performance because it fails to fully model the target especially when occlusion occurs. In another work, (Comaniciu et al., 2003) approach relied on gradient-based optimization and color-based histograms. In this case, no dynamic model is used therefore no occlusion can be predicted. (Deutscher & Reid, 2005) presented an interesting approach called annealing particle filter which aims to reduce the required number of samples. However, it could be inappropriate in a cluttered environment. They combine edge and intensity measures but they focused on motion analysis, and thus, no occlusion handling is explored.
To overcome the above problems, in this article, we made some improvements on color based object tracking and employed it to track a moving object. We proposed an object state not only object position but also speed, size, object size scale and appearance condition of the object. We applied also target model update condition and adaptive likelihood, ensuring the proper tracking object. By applying the appropriate initialization of the sample, we successfully track the moving object both known and unknown initial position of the object. We utilized the appearance condition and variance of the samples position for the tracking purpose without any other object detection methods. The color histogram is utilized as the observation model which is measured using Bhattacharyya distance as a similarity measurement between target model and the samples.
The outline of this article is as follows. In section 2, we describe briefly a particle filter algorithm. In section 3, our approach employing color particle filter is described in detail. We make an extension of our design method for robustness of the tracking method. In section 4, we show some experimental results in order to demonstrate the effectiveness of our method. Finally, section 5 presents the conclusions of the article.

2. Particle filter

2.1 Bayesian filtering framework

In this sub section, we briefly discuss the Bayesian approach for state estimation (Simon, 2006). Suppose we have a nonlinear system described by equations,

$$x_{k+1} = f_k(x_k, \omega_k)$$

$$y_k = h_k(x_k, v_k) \tag{1}$$

where, k is the time index, x_k is the state, ω_k is the process noise, y_k is the measurement, and v_k is the measurement noise, respectively. The function $f_k(.)$ and $h_k(.)$ are time-varying non-linear system and measurement equation. The noise sequences $\{\omega_k\}$ and$\{v_k\}$ are assumed to be independent and white with known pdf's. The goal of a Bayesian estimator is to approximate the conditional pdf of $\boldsymbol{x}_k$ based on measurements $y_1, y_2, \ldots, y_k$. This conditional pdf is denoted as $p(\boldsymbol{x}_k \mid \boldsymbol{y}_k)$. The first measurement is obtained at $k = 1$, so the initial condition of the estimator is the pdf of x_0, which can be written as $p(x_0) = p(x_0 \mid y_0)$ since y_0 is defined as the set of no measurements. Once we compute $p(\boldsymbol{x}_k \mid \boldsymbol{y}_k)$ then we can estimate $\boldsymbol{x}_k$.
The Bayesian estimator will find a recursive way to compute the conditional pdf $p(\boldsymbol{x}_k \mid \boldsymbol{y}_k)$. Before we find this conditional pdf, we will find the conditional pdf $p(\boldsymbol{x}_k \mid y_{1:k-1})$. This pdf can be calculated as

$$p(x_k \mid y_{1:k-1}) = \int p(x_k \mid x_{k-1}) p(x_{k-1} \mid y_{1:k-1}) dx_{k-1} \tag{2}$$

The second pdf on the right side of the above equation is not available yet, but it is available at the initial time ($p(x_0) = p(x_0 \mid y_0)$). The first pdf on the right side of the above equation is available. The pdf $p(x_k \mid x_{k-1})$ is simply the pdf of the state at time k given a specific state at time (k - 1). We know this pdf because we know the system equation $f_k(.)$ and we know the pdf of the noise ω_k. This pdf can be calculated according to dynamical model of the system $f_k(.)$
Next, we consider a *posteriori* conditional pdf of x_k . We can write this pdf as

$$p(x_k \mid y_{1:k}) = \frac{p(y_k \mid x_k) p(x_k \mid y_{1:k-1})}{p(y_k \mid y_{1:k-1})} \tag{3}$$

All of the pdf's on the right side of the above equation are available. The pdf $p(y_k \mid x_k)$ is available from our knowledge of the measurement equation $h_k(.)$ and our knowledge of the pdf of the measurement noise vk. The pdf $p(x_k \mid y_{1:k-1})$ is available from Eq. (2). Finally, the pdf $p(y_k \mid y_{1:k-1})$ is obtained as follows:

$$p(y_k \mid y_{1:k-1}) = \int p(y_k \mid x_k) p(x_k \mid y_{1:k-1}) dx_k \tag{4}$$

Both of the pdf's on the right side of the above equation are available. The pdf $p(y_k \mid x_k)$ is available from our knowledge of the measurement equation $h(.)$ and the pdf of vk, and $p(x_k \mid y_{1:k-1})$ is available from Eq. (2).
Summarizing the development of this section, the recursive equations of the Bayesian state estimation filter can be summarized as follows.

1. The system and measurement equations are given as follows:

$$x_{k+1} = f_k(x_k, \omega_k)$$

$$y_k = h_k(x_k, v_k)$$

where $\{\omega_k\}$ and $\{v_k\}$ are independent white noise processes with known pdf's.

2. Assuming that the pdf of the initial state $p(x_0)$ is known, initialize the estimator as follows:

$$p(x_0 \mid y_0) = p(x_0)$$

3. For k = 1, 2, …, perform the following.
   a. The a *priori* pdf is obtained from Eq. (2).

$$p(x_k \mid y_{1:k-1}) = \int p(x_k \mid x_{k-1}) p(x_{k-1} \mid y_{1:k-1}) dx_{k-1}$$

   b. The a *posteriori* pdf is obtained from Eq. (3) and (4).

$$p(x_k \mid y_{1:k}) = \frac{p(y_k \mid x_k) p(x_k \mid y_{1:k-1})}{\int p(y_k \mid x_k) p(x_k \mid y_{1:k-1}) dx_k}$$

Analytical solutions to these equations are available only for a few special cases. In particular, if $f(.)$ and $h(.)$ are linear, and x_0, $\{\omega_k\}$, and $\{v_k\}$ are additive, independent, and Gaussian, then the solution is the Kalman filter , otherwise it can be solved by particle filter.

2.2 Particle filter for object tracking

The particle filter was invented to numerically implement the Bayesian estimator which recursively approximates the posterior distribution using a finite set of weighted samples or particles. It has been introduced by many researchers to solve the estimation problem when the system is nonlinear and non-Gaussian. The basic idea behind the particle filter is Monte Carlo simulation, in which the posterior density is approximated by a set of particles with associated weights. As a Bayesian estimator, particle filter has two main steps: prediction and update. Prediction is done by propagating the samples based on the system model. The update step is done by measuring the weight of each samples based on the observation model. The implementation of particle filter can be described as follows.

1. Particle initialization.

Starting with a weighted set of samples at k-1 $\{\mathrm{x}_{k-1}^i, \pi_{k-1}^i; i = 1:N\}$ approximately distributed according to $p(\boldsymbol{x}_{k-1} \mid y_{1:k-1})$ as initial distribution $p(x_0)$, new samples are generated from a suitable proposal distribution, which may depend on the previous state and the new measurements.

2. Prediction step.

Using the probabilistic system transition model $p(\mathbf{x}_k \mid \mathbf{x}_{k-1})$, the particles are predicted at time k. It is done by propagating each particle based on the transition or system model.

$$x_{k+1} = f_k(x_k, \omega_k) = p(\boldsymbol{x}_k \mid \boldsymbol{x}_{k\text{-}1})$$

3. Update step.

To maintain a consistent sample, the new importance weights are set to

$$\pi_k^i = \pi_{k-1}^i \frac{p\left(\mathrm{y}_k \middle| \mathrm{x}_k^i\right) p\left(\mathrm{x}_k^i \middle| \mathrm{x}_{k-1}^i\right)}{q\left(\mathrm{x}_k \middle| \mathrm{x}_{1:k-1}, y_{1:k}\right)} \tag{5}$$

It is done by measuring the likelihood of each sample based on the observation model.

4. Resample.

This step is performed to generate a new samples set according to their weight for the next iteration. The resample step will decrease the number of the samples with low weight and will increase the number of high weight sample. The new particles set is re-sampled using normalized weights $\bar{\pi}_k^i$ as probabilities. This sample set represents the posterior at time k, $p(\mathrm{x}_k|y_{1:k})$.

5. Then, the expectations can be approximated as

$$\mathrm{E}p(\mathrm{x}_k|\mathrm{y}_{1:k}) \cong \sum_{i=1}^{N} \bar{\pi}_k^i \mathrm{x}_k^i \tag{6}$$

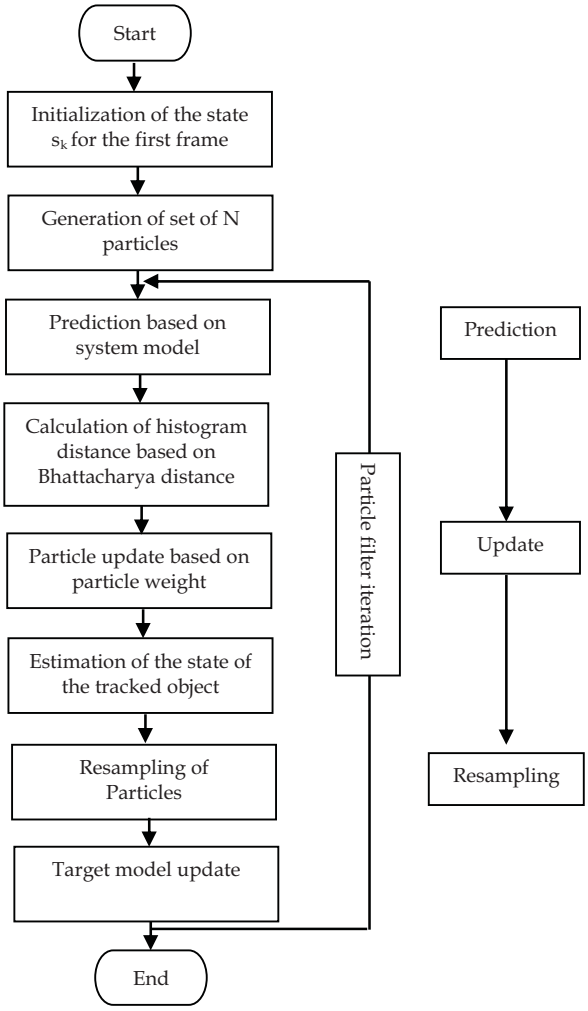


Fig. 1. Entired flow of procedure

3. An approach to robust tracking

In this article, we proposed an algorithm based on particle filter in conjunction with a color feature. The motion of the central point of a bounding box is modeled using first-order dynamics model. In this article, the state of the object is defined as $s_k = (x_k, \dot{x}_k, w_k, \dot{w}_k, A_k)$ where the components are position, speed, bounding-box size, bounding-box scale and pixel appearance, respectively. The observations y_k is given by input images I_k.

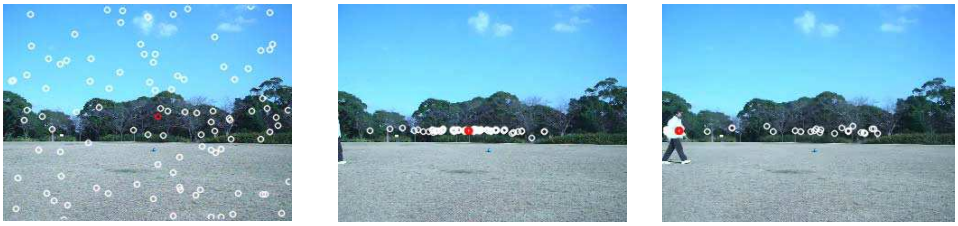
3.1 System flow diagram

In this article, we implement a particle filter in a color model-based framework to track the moving objects in outdoor environment. Firstly, the initialization of the samples done in the first frame is performed by drawing them randomly on entire scene or drawing them based on region where the object is expected to appear.

Next, the samples are predicted based on a system/transitional model by propagating each sample based on this model. The samples update is performed based on the observation model. In this article, we use color distribution of the object as the observation model. Then using the Bhattacharya distance, the similarity between the color distribution of the target and the samples can be measured. Based on the Bhattacharya distance, the weight of each sample is measured. The target sate estimation is performed based on the sample's weight. The resampling is performed for the next sample iteration to generate a new samples set. During the resample step, samples with a high weight may be chosen several times leading to identical copies, while others with relatively low weights may be ignored and deleted. And finally, the target model update is performed adaptively based on the best match of the target model. The overall flow diagram is shown in Fig. 1.

3.2 Initialization

In this article, the initialization of the particles is done on two approaches: (i) samples are initialized by putting the samples randomly on entire scene and (ii) by putting the samples around the region where the target is expected to appear such as edge of the scene or edge of the occluded object.



#1 #5 #10

Fig. 2. Initialization based on the variance of the samples

On the first approach (Fig. 2), the samples are initialized randomly on entire scene (frame #1). In this approach, we utilize the variance of the samples position to be a parameter of object tracking. We determine that the object is begun to track when the variance of the samples is below then certain threshold. Fig. 3 shows the variance of the samples position for each frame index. From that figure, we can understand that the object has not been

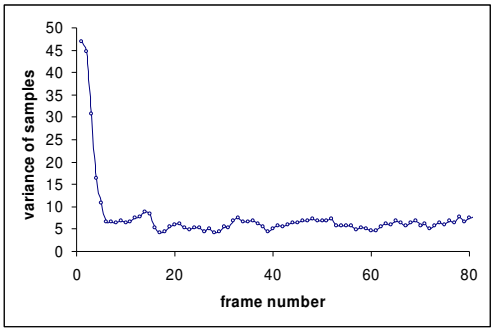


Fig. 3. Variance of the samples

tracked yet before the variance of the sample less than the threshold, although the object appeared on the scene. We can see from Fig. 2 that the object is begun to be tracked in frame #10 when the variance of the sample position is less than threshold.

#1 #9 #25

Fig. 4. Initialization based on expected region

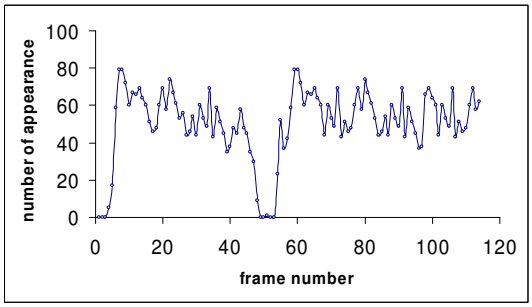


Fig. 5. Appearance of the samples

On the other hand on the second approach, the object tracking is begun when the samples satisfy some special appearance conditions. In this condition, the samples are put around the region where the target is expected to appear such as edge of the scene or edge of the occluded object. In this approach, if the target appears, the Bhattacharyya distances (later we will discuss it) of samples around the object position should be remarkable smaller than the average of sample set. Therefore, the mean value μ_b and the standard deviation σ_b of the Bhattacharyya distances of all initial samples are firstly calculated as,

$$\mu_b = \frac{1}{I}\sum_{i=1}^{I}\sqrt{1-\rho[p_{x_i},q]} \tag{7}$$

$$\sigma_b{}^2 = \frac{1}{I}\sum_{i=1}^{I}\left(\sqrt{1-\rho[p_{x_i},q]}-\mu\right)^2 \tag{8}$$

and then an appearance condition is defined as,

$$d = \sqrt{1-\rho[p_{x_i},q]} > \mu + 2\sigma \tag{9}$$

indicating a 95[%] confidence that a sample belongs to the object. Here, $\rho[p_{xi},q]$ is the Bhattacharya distance between the target model and the object. Then, an appearing and disappearing threshold T is defined to indicate the quantity of the samples fulfilling the appearance condition during initialization. More than T means the target being found and starting tracking, contrary situation means the tracker will enter the initialization mode. Fig. 4 shows the initialization approach (frame #1). The object tracking is begun when the appearance condition is fulfilled (frame #9). The appearance of the samples is shown in Fig. 5. As presented in that figure, the object is detected and started to be tracked when the number of samples appearance more than threshold T. From that figure also, we can understand that object disappear behind the tree. It is shown by the number of samples appearance become zero. Thus, using the appearance condition, we not only can detect the object but also can predict the occlusion of the object. The predicted occlusion is illustrated in Fig. 6. The occlusion is predicted occurs on frame #50.

#45 #50 #60

Fig. 6. Occlusion prediction based on appearance condition

3.3 Transition/system model

In this article, we consider the motion of the object as the discrete time 2-dimensional (2D) motion with constant velocity assumption. The state vector at a time step k is denoted by $\boldsymbol{s}_k$, including position, speed, size and bounding box scale of each sample. For this purpose, we model the system based on the following expression.

$$\begin{aligned}
\hat{\mathrm{x}}_k &= \mathrm{x}_{k-1} + \dot{\mathrm{x}}_{k-1}\Delta t + \xi_{\mathrm{x}}, \\
\hat{\dot{\mathrm{x}}}_k &= \dot{\mathrm{x}}_{k-1} + \xi_{\dot{\mathrm{x}}}, \\
\hat{\mathrm{w}}_k &= \mathrm{w}_{k-1} + \dot{\mathrm{w}}_{k-i}\Delta t + \xi_{\mathrm{w}}, \\
\hat{\dot{\mathrm{w}}}_k &= \dot{\mathrm{w}}_{k-1} + \xi_{\dot{\mathrm{w}}}
\end{aligned} \tag{10}$$

Here, $\hat{x}_k, \hat{\dot{x}}_k, \dot{w}_k, \hat{\dot{w}}_k$ are the sample position, speed, bounding box and bounding box scale of the state, respectively. The random vectors $\xi_x, \xi_{\dot{x}}, \xi_w, \xi_{\dot{w}}$ are Gaussian noise providing the system with a diversity of hypotheses. Then, each component of the samples is predicted by propagating the samples according to this transition model.

3.4 Observation model

The observation model is used to measure the observation likelihood of the samples. Many observation models have been built for particle filtering tracking. One of them is a contour based appearance template (Isard & Black, 1998). The tracker based on a contour template gives an accurate description of the targets but performs poorly in clutter and is generally time-consuming. The initialization of the system is relatively difficult and tedious. In contrast, color-based trackers are faster and more robust, where the color histogram is typically used to model the targets to combat the partial occlusion, and non-rigidity (Nummiaro et al., 2002; Perez et al., 2002).

In this article, the observation model is made based on color information of the target obtained by building the color histogram in the RGB color space. This section describes how the color features is modeled in a rectangular 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, rotation and scale invariant. They are also flexible in the types of object that they can be used to track, including rigid and non-rigid object.

The color distribution is expressed by an *m*-bins 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 $\mathbf{X} = \{x_i, i = 1, 2, \ldots, n\} \in R$, the *m*-bins 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:

$$h_j = \frac{1}{n} \sum_{(x_i) \in \mathrm{X}} \delta_j [b(x_i)] \tag{11}$$

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

To increase the reliability of the target model, smaller weight are assigned to the pixels that are further away from region center by employing a weighting function

$$g(r) = \begin{cases} 1 - r^2 & r < 1 \\ 0 & otherwise \end{cases} \tag{12}$$

Here, *r* is the distance from the center of the region.

Using this weight, the color histogram $p_y = \left\{ p_y^{(u)} \right\}$ $u = 1, \ldots, m$ at location $\boldsymbol{y}$ is calculated as

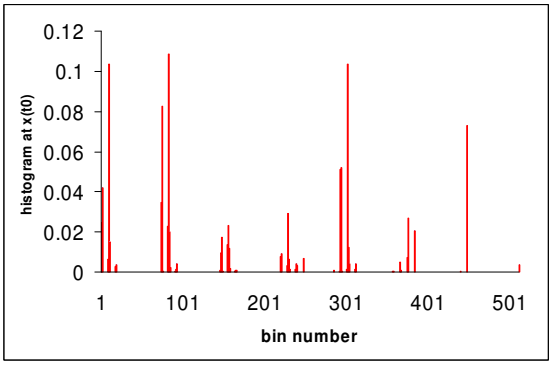
$$p_y^{(u)} = f \sum_{j=1}^{I} g\left(\frac{\left\| y - x_j \right\|}{a} \right) \delta \left[h\left(x_j \right) - u \right] \tag{13}$$

where, m is number of bins, I is the number of pixels in the region R, x_j is the position of pixels in the region R, δ is the Kronecker delta function, a is the normalization factor, f is the scaling factor to ensures that $\sum_{u=1}^{m} p_y^{(u)} = 1$, and $g(.)$ is weighting function, respectively.
Fig. 7 shows an example of target histogram at time step k. In subsequent frames, at every time k, 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 model histogram q and the histogram $p(x_t)$ of object in the region described by the i^{th} particle x_k^i are.
The similarity between two color histograms can be calculated using Bhattacharya distance $d = \sqrt{1-\rho[p,q]}$, where $\rho[p,q] = \sum_{u=1}^{m} \sqrt{p^{(u)} q^{(u)}}$. Similar histogram will have a small Bhattacharya distance which corresponds to high sample weight. Based on the Bhattacharya distance, the weight $\pi^{(i)}$ of the sample state $x^{(i)}$ is calculated as,

$$\begin{aligned} \pi^{(i)} &= \frac{1}{\sqrt{2\pi\sigma}} \exp\left(\frac{d^2}{2\sigma^2}\right) \\ &= \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{\left(1-\rho\left[p\left(\mathrm{x}^{(i)}\right), q\right]\right)}{2\sigma^2}\right) \end{aligned} \tag{14}$$

Where $p(x^{(i)})$ and q are the color histogram of the samples and target model, respectively. During the resample step, samples with a high weight may be chosen several times leading to identical copies, while others with relatively low weights may be ignored.

(a) target object at time k



(b) histogram of a target

Fig. 7. An example of color histogram distribution of a target model at time k

3.5 State estimation

Each of the target state is estimated according their mean estimation as following equation,

$$\mathrm{x}_k = (1-\alpha_{\mathrm{x}})(\mathrm{x}_{k-1} + \dot{\mathrm{x}}_{k-1}\Delta t) + \alpha_{\mathrm{x}}\left(\sum_{i=1}^{N} \bar{\pi}_k^i \hat{\mathrm{x}}_k^i\right)$$

$$\mathrm{w}_k = (1-\alpha_{\mathrm{w}})\mathrm{w}_{k-1} + \alpha_{\mathrm{w}}\left(\sum_{i=1}^{N} \bar{\pi}_k^i \hat{w}_k^i\right)$$

$$\dot{\mathrm{x}}_k = (1-\alpha_{\dot{\mathrm{x}}})\dot{\mathrm{x}}_{k-1} + \alpha_{\dot{\mathrm{x}}}\left(\frac{\mathrm{x}_k - \mathrm{x}_{k-1}}{\Delta t}\right) \tag{15}$$

$$\dot{\mathrm{w}}_k = (1-\alpha_{\dot{\mathrm{w}}})\dot{\mathrm{w}}_{k-1} + \alpha_{\dot{\mathrm{w}}}\left(\frac{\mathrm{w}_k - w_{k-1}}{\Delta t}\right)$$

Here, $\alpha_{\mathrm{x}}, \alpha_{\dot{\mathrm{x}}}, \alpha_{\mathrm{w}}, \alpha_{\dot{\mathrm{w}}} \in [0,1]$ denote the adaptation rates.

3.6 Resample step

The resample step is performed for the next sample iteration to generate a new samples set. During the resample step, samples with a high weight may be chosen several times leading to identical copies, while others with relatively low weights may be ignored and deleted. The resample step can be done in several different ways. One straightforward way is as the following steps [Ristic et al., 2004].

1. Generate a random number *r* that is uniformly distributed on [0, 1].
2. Calculate the normalized cumulative probability

$$c_k^{(0)} = 0, \quad c_k^{(i)} = c_k^{(i-1)} + w_k^{(i)}, \quad c_k'^{(i)} = \frac{c_k^{(i)}}{c_k^{(N)}} \tag{16}$$

3. By binary search, find the smallest *j* for which $c_k' \geq r$ and set the new particle $x_k^i = x_k^j$

#12 #16 #20

(a) Using non adaptive likelihood

#12 #16 #20

(b) Using adaptive likelihood

Fig. 8. Object tracking using adaptive and non adaptive likelihood

3.7 Adaptive likelihood

The observation likelihood function is very important for the tracking performance of particle filter. The reason is that the observation likelihood function determines the weights of the particles and determines how the particles are resampled. Resample step is used to decrease the number of low-weighted particles and increase the number of particles with high weights. The observation likelihood function is expressed in Eq. (14). In this case, the observation likelihood function is the only function that contributes to particle's weight. We can rewrite the equation as,

$$\pi^{(i)} \propto \exp\left(-\frac{d^2}{2\sigma^2}\right) \tag{17}$$

here, the σ parameter determines the steepness of the function, i.e. how fast the function will decrease when the distance is large (bad particles).

It is reported that the value of the parameter σ has a major influence on the properties of the likelihood (Freeman & Adelson, 1992). Typically, the choice of this value is left as a design parameter to be determined experimentally. However, in this article, an adaptive scheme is proposed which aims to maximize the information available in the likelihood using the Bhattacharya distance d. For this purpose, we define the minimum squared distance $d^2_{\min}$ as the minimum distance d^2 of the set of distance calculated for all particles. Rearranging the likelihood function yields,

$$\log(\pi) \propto -\frac{d^2}{2\sigma^2} \tag{18}$$

from which we can obtain σ that gives maximum likelihood as,

$$\sigma \propto \frac{\sqrt{2}}{2} d^2_{\min} \tag{19}$$

Fig. 8 shows a comparison the object tracking method using the adaptive and non-adaptive likelihood model. Using non adaptive likelihood, the particles with different Bhattacharya distance will be assigned with equal weights. As a result, the likelihood function will not differentiate the difference between particle with small distance and particle with big distance. This condition may assign the bad particles with high weight (Fig. 8 (a)). On the other hand, the adaptive likelihood ensures the best particles are assigned with the highest weights and the worst ones are assigned with small weight (Fig. 8 (b)).

3.8 Target model update

The problem for object tracking with color occurs when the region around the object is cluttered, illumination is change and the object appearance is change. In this way, a color feature based tracking does not provide reliable performance because it fails to fully model the target. To overcome this problem, the adaptive target model is applied based on the best match of the color model. However, this is a sensitive task. The target models are only updated when two conditions hold: (i) the target is not occluded and (ii) the likelihood of

the estimated target's state suggests that the estimate is sufficiently reliable. In this case, they are updated using an adaptive filter

$$q_k = (1 - \alpha_q) q_{k-1} + \alpha_q p_{est} \tag{20}$$

where $a_q \in [0, 1]$ is the learning rate that contribute to the updated histogram and p_{est} is histogram of the estimated state, respectively. In order to determine when the estimate is reliable, the likelihood of the current estimate is computed, π_{est}. The appearance is then updated when this value is higher than an indicator of the expected likelihood value and is calculated following an adaptive rule

$$\lambda_k = (1 - \alpha_u) \lambda_{k-1} + \alpha_u \pi_{est} \tag{21}$$

here λ_k is expected likelihood, $a_u \in [0, 1]$ is the learning rate and π_{est} is estimated likelihood, respectively. This value indicates that the object has to be well matched to the model histogram before the update is applied.

Fig. 9 and Fig. 10 show the comparison of object tracking with and without target model update. As shown in Fig. 9, non-adaptive model fail to detect the object properly. However, the adaptive model can track the object successfully. It is because, using the adaptive model, the target model is always updated based on the best match of the color model. Moreover, as shown in Fig. 10, using adaptive model the scale of object appearance is getting adapted correctly compare to without using adaptive model.

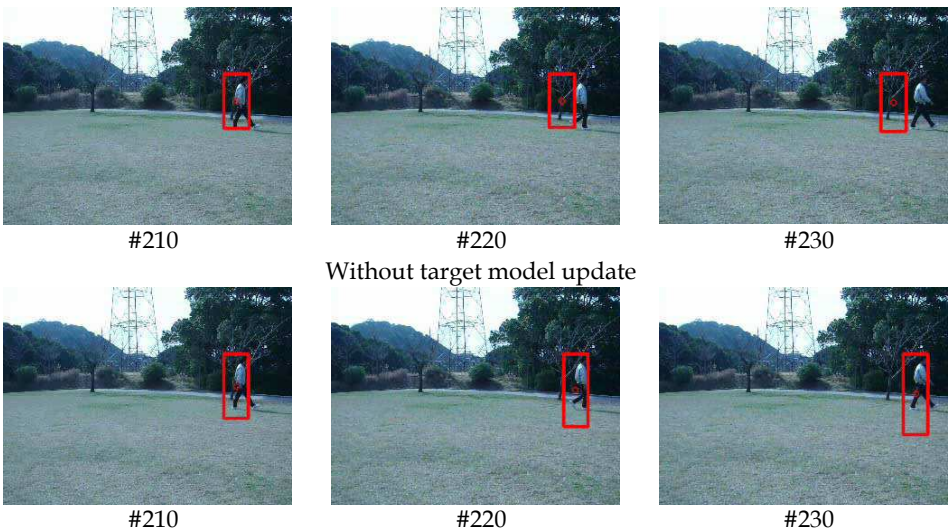


Fig. 9. Object tracking with and without model update

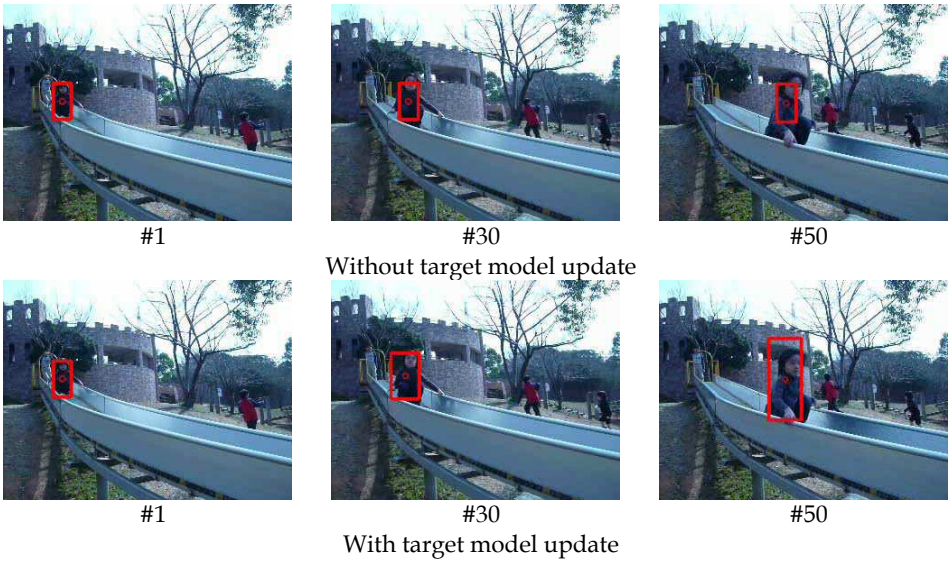


Fig. 10. Object tracking with model update shows the scale adaptation

4. Experimental results

In order to evaluate our proposed method, we have done the experiments using a video camera to track the objects in outdoor environment. The experiments are implemented on Pentium 4 with 2.54 [GHz] CPU and 512 [MB] RAM. The resolution of each frame is 320x240 [pixels] image. The color histogram is calculated in RGB space with $8\times8\times8$ [bins]. The experimental results are shown in Fig. 11 – Fig. 18. In each experimental result, the white dots represent the samples' distribution, the red dot represents the mean state of the samples' position and the red rectangle represents the bounding box of the tracked object used in calculation of the histogram distribution.

We did the experiments in three conditions. In the first condition, we tracked the moving object when the initial position of the object is unknown as shown in Fig. 12. In the second condition, we tracked the moving object when the initial position of the object is known as shown in Fig. 13 and Fig. 15. In the third condition, we tracked the moving object based on the appearance condition including the handling of object occlusion by another object as shown in Fig. 16 and Fig. 17.

4.1 Object tracking with unknown initial position

In this experiment, we track the moving object when the initial position of the object is unknown. We set the initial samples to be uniform distribution in $x_0 \sim U(1,320)$ and $y_0 \sim U(1,240)$. The experimental result is shown in Fig. 12. As presented in the figure, initially, the samples are distributed uniformly around the scene (frame #1). The object moves from left-hand to right-hand side and begin to appear on frame #5, however, the object starts to be tracked on frame #10. In this experiment, the variance of the samples' position distribution is utilized to judge whether the object has been tracked. We consider the object

has been tracked by the system when the variance is below 10 and it occurs on the frame #10. Fig. 11 shows the variance positions of the samples. We can understand from that figure that at the beginning the variance is very high but after some while it decreases below the threshold. At that time, we determine that object is being tracked.

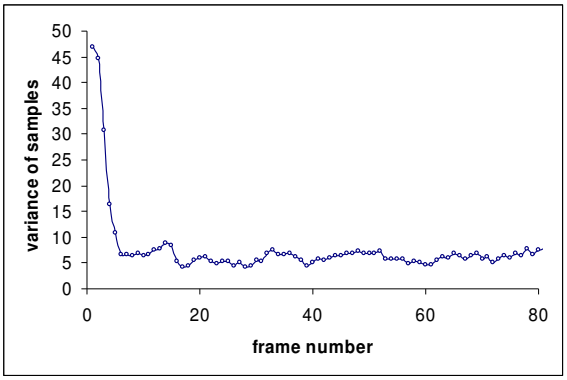


Fig. 11. Variance of samples' position

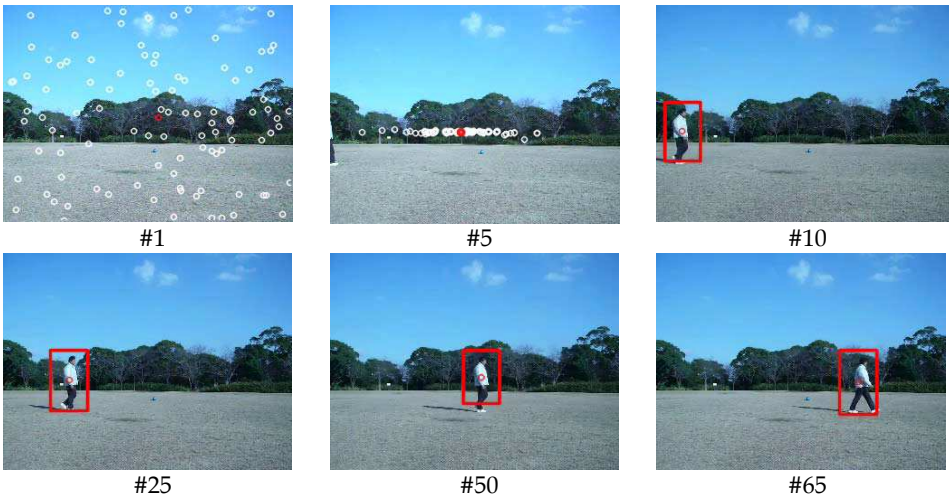


Fig. 12. Object tracking with unknown initial position

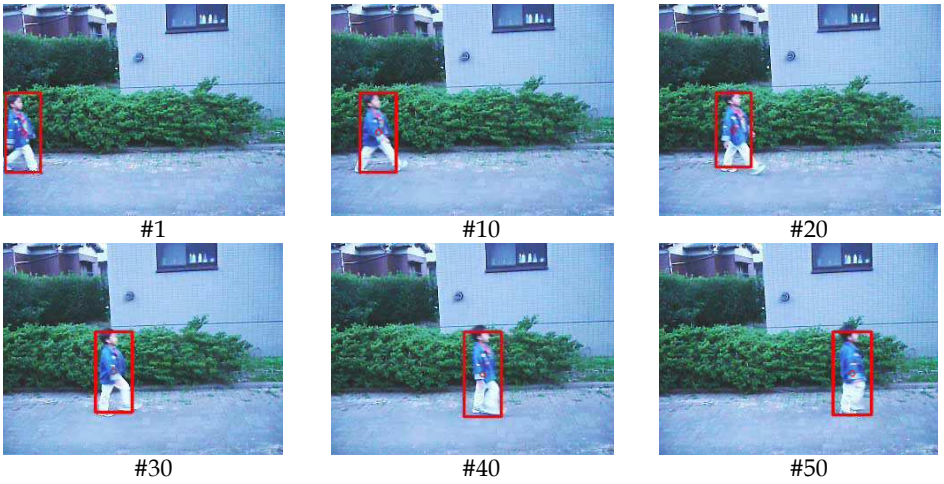


Fig. 13. Object tracking with known initial position

4.2 Object tracking with known initial position

In this experiment, the initial position of the moving object is known. We set the initial samples around the initial position of the object i.e. $x_0 \sim N(22,3)$ and $y_0 \sim N(145,3)$. The tracking result is presented in Fig. 13. Initially, the samples are distributed around initial position of the object. The object moves from left-hand to right-hand side and is tracked from the initial position (frame #1). We successfully tracked the object from the initial position until the end of the scene. Fig. 14 shows the comparison between true and estimated position of the tracked object. From that figure, we obtained the RMSE (root mean square error) of the estimated position is about 2.06 for X position and 2.18 for Y position.

Another result is shown in Fig. 15. Here, we set the initial samples around the initial position of the object i.e. $x_0 \sim N(65,3)$ and $y_0 \sim N(106,3)$. The obejct is tracked from the initial position. We can observe that the scale of object bounding box is adaptively updated. It is because, in this experiment, the target model is always updated using the adaptive model based on the best match of the color model.

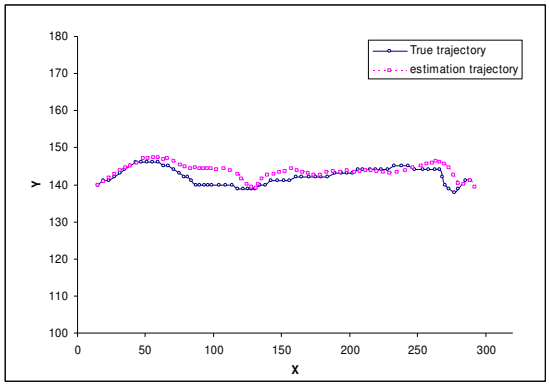


Fig. 14. True vs. estimate position

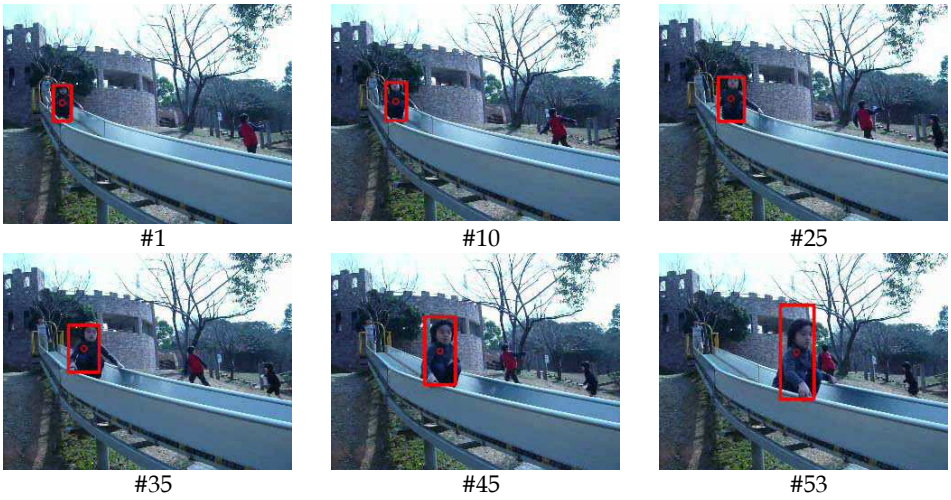


Fig. 15. Another result of object tracking with known initial position

Fig. 16. Object tracking based on appearance condition

4.3 Object tracking with appearance condition and in the presence of occlusion

In this experiment, the samples initialize around the expected region where the target is most likely to appear, such as image borders and edge of the occluded object (tree). The experimental results are shown in Fig.16.

As presented in Fig. 16, the initial samples are placed at the position where the object is most likely to appear, such as image borders and edge of the occluded object (tree) as shown in frame #1. The object is started to appear on frame #6, but as the appearance condition is not fulfilled, the object is still judged not to be tracked. However, when the appearance condition is fulfilled (frame #9), the object starts to be tracked.

Fig. 17 shows the object occlusion handling based on appearance condition. As shown in that figure, we successfully track the object although the part and full occlusion occur. When occlusion occurs, the final mean state estimate is still calculated and the object is still tracked (frame #45 – frame #50). On that condition, the update and resample step are not performed due to no object color information can be represented the object. The samples are just propagated based on the system model and the target model is not updated until no more occlusion is detected. The object appears behind the tree (frame #55), due to the fulfilling of appearance condition, then the sample update and resample is performed again and the object continues to be tracked until the end of the frame.

The number of samples that fulfill the appearance condition is shown in Fig. 18. As presented in that figure, the object is detected and started to be tracked when the number of appearance more than 10. That means the object is detected and tracked when the number of samples fulfilling the appearance condition is more than 10 samples. We can see from the figure, the number of appearance is started by zero at the initial position. After some while, the number of appearance increases due to appearance of the object. Using the threshold 10 samples appearance, the object starts to be tracked on frame #9 (Fig. 16). The object continues to be tracked until it is occluded by tree (Fig. 17 frame #45 – frame #50). At this time, the number of samples at appearance condition is dropped to zero. At frame #55, the number of samples at appearance condition is higher again. It shows that the object is no more occluded by tree.

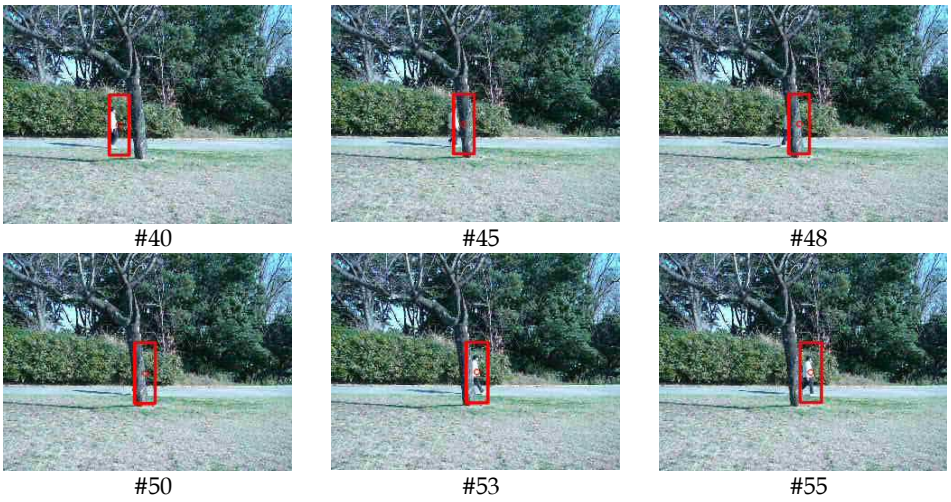


Fig. 17. Object tracking in the presence of oclusion

5. Conclusions

This paper presented a method to track the moving object employing particle filter in conjunction with color information in both known and unknown initial position of the object and in the presence of occlusion with static object. The robust color likelihood is used to evaluate samples properly associated to the target which present high appearance

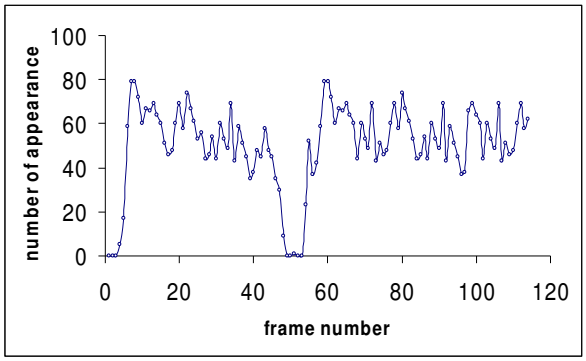


Fig. 18. Number of samples fulfilling the appearance condition

variability. We rely on Bhattacharya coefficient between target and sample histogram to perform this task. Model updating is carried to update the object in the presence of appearance change. The problem occlusion by static object can be tackled by prediction based on the dynamical model and appearance condition. The performance of the tracking algorithm was evaluated by the experiments. The experimental results show the algorithm can successfully track the single moving object in the presence of occlusion with static object and both known and unknown initial position of the object

Furthermore, the performance of the objects tracking can be improved in several ways such as adding the background modeling information in the calculation of likelihood, detection of appearing objects and extend the method to track multiple object in the presence of object occlusion and so on. Taking them into consideration could lead to some improvement. These are remaining for our future works.

6. References

Stauffer, C. & Grimson, W. (1999). Adaptive background mixture models for real-time tracking, *Proceeding of IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 246-252.

LIU, Y.; Haizho, A. & Xu Guangyou.(2001). Moving object detection and tracking based on background subtraction, *Proceeding of Society of Photo-Optical Instrument Engineers (SPIE)*, Vol. 4554, pp. 62-66.

McIvor, A. M. (2000). Background subtraction techniques, *Proceeding of Image and Vision Computing*, 6 pages.

Heikkila, J. & Silven, O. (1999). A real-time system for monitoring of cyclists and pedestrians, *Proceeding of Second IEEE Workshop on Visual Surveillance*, pp 74–81.

Lipton, A; Fujiyoshi, H. & Patil, R.(1998) .Moving target classification and tracking from real-time video, *Proceeding of IEEE Workshop Applications of Computer Vision*, pp. 8-14.

Collins, R. ; Lipton, A.; Kanade, T.; Fujiyoshi, H.; Duggins, D.; Tsin, Y.; Tolliver, D.; Enomoto, N. & Hasegawa. (2000). System for video surveillance and monitoring, *Technical Report CMU-RI-TR-00-12*, Robotics Institute, Carnegie Mellon University.

Meyer, D.; Denzler, J. & Niemann, H. (1998). Model based extraction of articulated objects in image sequences for gait analysis, *Proceeding of IEEE Int. Conf. Image Proccessing*, pp. 78-81.

Phung, S.; Chai, D. & Bouzerdoum, A. (2003). Adaptive skin segmentation in color images, *Proceeding of IEEE International Conference on Acoustics, Speech and Signal Processing*, Vol. 3, pp. 353-356.

Cho, K. M.; Jang, J. H. & Hong, K. S. (2001). Adaptive skin-color filter, *Pattern Recognition*, pp. 1067-1073.

Broida, T. & Chellappa, R. (1986). Estimation of object motion parameters from noisy images, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 8, No. 1, pp. 90–99.

Ristic, B.; Arulampalam, S. & Gordon, N. (2004). Beyond the Kalman filter: Particle filters for tracking applications, Artech House.

Isard, M. & Blake, A. (1998). CONDENSATION –Conditional Density Propagation for Visual Tracking, *Intl. Journal of Computer Vision*, Vol. 29, No. 1, pp. 5-28.

Kitagawa G. (1996). Monte Carlo filter and smoother for non-Gaussian nonlinear state space models, *Journal of Comput. and Graph. Stat.*, Vol. 5, No. 1, pp. 1–25.

Gordon, N.; Salmon, D. & Smith, A. (1993). Novel approach to nonlinear/non-Gaussian Bayesian state estimation, *IEE Proc. on Radar and Signal Processing*, Vol. 140, No. 2, pp. 107-113.

Tanizaki, H. (1987). Non-Gaussian state-space modeling of non-stationary time series, *Journal of American Statistic Association*, Vol. 82, pp. 1032–1063.

King, O. & Forsyth, D. (2000). How does Condensation behave with a finite number of samples, *Proc. of 6th European Conference on Computer Vision*, pp. 695-709.

Nummiaro, K.; Koller, E. & Gool, L. (2002). An adaptive color-based particle filter, *Image and Vision Computing*, Vol. 25, No. 1, pp. 99–110.

Perez, P.; Hue, C. & Vermaak, J. (2002). Color-based probabilistic tracking, *Proc. of the 7th European Conference on Computer Vision*, 661–675.

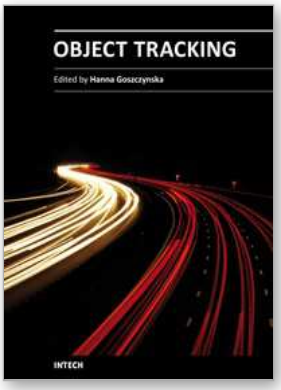
Comaniciu, D.; Ramesh, V. & Meer, P. (2000). Real-time tracking of non-rigid objects using mean shift, *Proc. of the IEEE Conf. on Comp. Vision and Pattern Recog.*, pp. 142–149.

Comaniciu, D.; Ramesh, V. & Meer, P. (2003), Kernel-based object tracking, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 25, No.5, pp. 564-577.

Deutscher, J. & Reid, I. (2005). Articulated body motion capture by stochastic search, *Intl. Journal of Computer Vision*, Vol. 61, No. 2, pp. 185-205.

Simon, D. (2006). *Optimal State Estimation*, Jhon Wiey & Sons, Inc, Publication.

Freeman, W. & Adelson, E. (1991). The Design and Use of Steerable Filters, *IEEE Transactions on Pattern* Analysis *and Machine Intelligence*, Vol. 13, No. 9, pp. 891-906.



Object Tracking

Edited by Dr. Hanna Goszczynska

ISBN 978-953-307-360-6
Hard cover, 284 pages
Publisher InTech
Published online 28, February, 2011
Published in print edition February, 2011

Object tracking consists in estimation of trajectory of moving objects in the sequence of images. Automation of the computer object tracking is a difficult task. Dynamics of multiple parameters changes representing features and motion of the objects, and temporary partial or full occlusion of the tracked objects have to be considered. This monograph presents the development of object tracking algorithms, methods and systems. Both, state of the art of object tracking methods and also the new trends in research are described in this book. Fourteen chapters are split into two sections. Section 1 presents new theoretical ideas whereas Section 2 presents real-life applications. Despite the variety of topics contained in this monograph it constitutes a consisted knowledge in the field of computer object tracking. The intention of editor was to follow up the very quick progress in the developing of methods as well as extension of the application.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Budi Sugandi, Hyoungseop Kim, Joo Kooi Tan and Seiji Ishikawa (2011). Object Tracking Based on Color Information Employing Particle Filter Algorithm, Object Tracking, Dr. Hanna Goszczynska (Ed.), ISBN: 978-953-307-360-6, InTech, Available from: http://www.intechopen.com/books/object-tracking/object-tracking-based-on-color-information-employing-particle-filter-algorithm

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821