Better Proposal Distributions: Object Tracking Using Unscented Particle Filter

Yong Rui and Yunqiang Chen

Collaboration and Multimedia Systems Group, Microsoft Research

One Microsoft Way, Redmond, WA 98052-6399

yongrui@microsoft.com and chenyq@ifp.uiuc.edu

Abstract

Tracking objects involves the modeling of non-linear non-Gaussian systems. On one hand, variants of Kalman filters are limited by their Gaussian assumptions. On the other hand, conventional particle filter, e.g., CONDENSATION, *uses transition prior as the proposal distribution. The transition prior does not take into account current observation data, and many particles can therefore be wasted in low likelihood area. To overcome these difficulties, unscented particle filter (UPF) has recently been proposed in the field of filtering theory. In this paper, we introduce the UPF framework into audio and visual tracking. The UPF uses the unscented Kalman filter to generate sophisticated proposal distributions that seamlessly integrate the current observation, thus greatly improving the tracking performance. To evaluate the efficacy of the UPF framework, we apply it in two real-world tracking applications. One is the audio-based speaker localization, and the other is the vision-based human tracking. The experimental results are compared against those of the widely used* CONDENSATION *approach and have demonstrated superior tracking performance.*

1. Introduction

Reliable object tracking in complex audio-visual environment is an important task. Its applications include human computer interaction [8,9], teleconferencing [19,20], and surveillance [12], among many others. It is also a very challenging task in that objects' state space representation can be highly non-linear and the observation (e.g., audio and/or visual sensory data) is almost always corrupted by background clutters.

Temporal Bayesian filtering (e.g., CONDENSATION [8]) is one of the most successful object-tracking paradigms. Let $x_{0:t}$ and $y_{0:t}$ represent the state trajectory and observation history of a system from time *0* to time *t*, *filtering* is the process of estimating system's current state, based on its past and current observations, i.e., $p(x_t \mid x_{t-1}, y_{0:t})$. For different applications, state x_t and observation y_t can represent different entities. In visual tracking, for example, x_t can be the position and orientation of a human face, and y_t can be the pixel intensities and contours of the captured image. In audio-based tracking, e.g., sound source localization [20], x_t can be the horizontal panning angle, and y_t can be the generalized cross-correlation function between two microphones. Regardless of the applications, the object-tracking problems can be modeled by the same mathematical state space representation:

$$p(x_t \mid x_{t-1}): \quad x_t = f(x_{t-1}, m_{t-1}) \tag{1}$$

$$p(y_t \mid x_t): \quad y_t = h(u_t, x_t, n_t) \tag{2}$$

where Equation (1) is the system dynamics, Equation (2) is the system observation, u_t is the system input, m_t and n_t are the process noise and observation noise, respectively. If *f( )* and *h( )* are linear functions and if Gaussian distribution is assumed for x_t, m_t and n_t , $p(x_t \mid x_{t-1}, y_{0:t})$ has an analytical solution which is the well-known Kalman filter [1]. Unfortunately, tracking objects in real-world environment seldom satisfies Kalman filter's requirements. For example, in human tracking, background clutter may resemble the human face, and in sound source localization, "ghost" sound sources can create multiple peaks in the generalized cross-correlation function. To make the situation worse, the system dynamics and observation can be highly non-linear. In order to deal with the non-linear and/or non-Gaussian reality, two categories of techniques have been developed in the past: *parametric* and *non-parametric*.

The parametric techniques are based on improvements of the Kalman filter. By linearizing non-linear functions around the predicted values, extended Kalman filter (EKF) is proposed to solve non-linear system problems. It is first introduced in control theory [1] and later on applied in visual tracking [3]. Because of its first-order approximation of Taylor series expansion, EKF finds only limited success in tracking visual objects [8]. In recent years, Julier and Uhlmann develop an unscented Kalman filter (UKF) that can accurately compute the mean and covariance of $y = g(x)$, where *g( )* is an arbitrary function, up to the second order (third in Gaussion prior) of the Taylor series expansion of *g( )* [10]. While UKF is significantly better than EKF in density statistics estimation, it still assumes a Gaussian parametric form of the posterior, thus cannot handle multi-modal distributions.

The non-parametric techniques are based on Monte Carlo simulations. They assume no functional form, but instead, use a set of random samples (also called *particles*) to estimate the posteriors. When the particles are properly placed, weighted, propagated, posteriors can be estimated sequentially over time. This technique is more popularly known as the *particle filters* in recent years. The first appearance of particle filters can be traced back to 1950s [7]. While almost dormant in the seventies, there is a renaissance of this technique in the early nineties [6,8,14,17], due to the massive increases in computing power. However, most of them use the state transition prior $p(x_t|x_{t-1})$ as the proposal distribution to draw particles from [8,18]. Because the state transition does not take into account the most recent observation y_t, the particles drawn from transition prior may have very low likelihood, and their contributions to the posterior estimation become negligible. This type of particle filters is prone to be distracted by background clutters [5,11,17]. For clarity, in this paper, we refer this type of filters as the *conventional particle filters*.

Inside the computer vision community, particle filters has also enjoyed considerable attention. Following the pioneering work of CONDENSATION [8], various improvements and extensions have been proposed for visual tracking [2,9,16]. Because the original CONDENSATION algorithm uses the state transition prior as its proposal distribution, it belongs to the conventional particle filters. To design better proposal distributions for CONDENSATION, in general, there are two approaches: the *direct* approach and the *indirect* approach. The indirect approach attacks this problem indirectly by using an auxiliary tracker to generate the proposal distribution for the main tracker. The direct approach, on the other hand, addresses this problem directly in its original space by taking into account the most recent observation. The indirect approach is adopted in the ICONDENSATION algorithm [9], where an auxiliary color tracker is used to generate the proposal distribution for the main contour tracker. While better than the conventional particle filters, this indirect approach has two major limitations. First, in many applications, e.g., audio-based speaker localization, there is simply no easy auxiliary tracker or sensing modality available. Second, and more importantly, the auxiliary tracker itself needs a good proposal distribution if it plans to use particle filters, or it falls back to *ad hoc* approaches.

Merwe *et. al.* have recently developed the unscented particle filter (UPF) in the field of filtering theory [17]. Based on this new development, in this paper, we introduce a *direct* approach to generate better proposal distributions for audio/visual tracking. The UPF is a parametric/non-parametric hybrid of UKF and particle filters. The particle filter part of the UPF provides the general probabilistic framework to handle non-linear non-Gaussian systems, and the UKF part of the UPF generates better proposal distributions by taking into account the most recent observation.

The rest of the paper is organized as follows. In Section 2, we present a new formulation of the particle filter framework that accentuates the importance of the proposal distribution. In Section 3, we present the UKF, which can be used to generate more accurate proposal distributions for particle filters. The resultant filter is the high-performance hybrid filter UPF. In Sections 4 and 5, we apply the UPF framework in two real-world audio/visual tracking applications. One is the audio-based speaker localization, and the other is vision-based human tracking. Experimental results of both applications demonstrate the superior performance of UPF over the conventional particle filters. We give concluding remarks in Section 6.

2. Particle Filtering

In the pioneering work of CONDENSATION [8], extended factored-sampling is used to formulate the particle filter framework. Even though easy to follow, it obscures the role of proposal distributions. In this section, we present a new formulation of particle filtering theory that is centered around proposal distributions. This new formulation illustrates how to improve the particle filter's performance by designing better proposal distributions.

2.1. Bayesian sequential importance sampling

A non-parametric way to represent a distribution is to use particles drawn from the distribution. For example, we can use the following point-mass approximation to represent the posterior distribution of x:

$$\hat{p}(x_{0:t} \mid y_{1:t}) = \frac{1}{N}\sum_{i=1}^{N} \delta_{x_{0:t}^{(i)}}(dx_{0:t}) \tag{3}$$

where δ is the Dirac delta function, and particles $\{x_{0:t}^{(i)}\}$ are drawn from $p(x_{0:t}|y_{1:t}\ )$. The approximation converges in distribution when N is sufficiently large [5,17]. This particle-based distribution estimation is, however, only of theoretical significance. In reality, the posterior distribution is the one that needs to be estimated, thus not known. Fortunately, we can instead sample the particles from a known *proposal distribution* $q(x_{0:t}|y_{1:t})$ and still be able to compute $p(x_{0:t}|y_{1:t})$.

Definition 1 [14]: A set of random samples $\{x_{0:t}^{(i)}, w_t(x_{0:t}^{(i)})\}$ drawn from a distribution q is said to be *properly weighted* with respect to p if for any integrable function $g(\ )$ the following is true

$$E_p(g(x_{0:t})) = \lim_{N\to\infty} \sum_{i=1}^{N} g(x_{0:t}^{(i)}) w_t(x_{0:t}^{(i)}) \tag{4}$$

Furthermore, as N tends to infinity, the posterior distribution p can be approximated by the *properly weighted* particles drawn from q [4,14,17]:

$$\hat{p}(x_{0:t} \mid y_{1:t}) = \sum_{i=1}^{N} w_t(x_{0:t}^{(i)}) \delta_{x_{0:t}^{(i)}}(dx_{0:t}) \tag{5}$$

There are two important points worth emphasizing here. First, the definition says that an *unknown distribution p* can be approximated by a set of *properly weighted* particles drawn from a *known distribution q*. Second, the more difficult problem of distribution estimation is converted to an easier problem of weight estimation. The weights are further given by:

$$\tilde{w}_t(x_{0:t}^{(i)}) = p(y_{1:t} \mid x_{0:t}^{(i)}) p(x_{0:t}^{(i)}) / q(x_{0:t}^{(i)} \mid y_{1:t}) \tag{6}$$

$$w_t\ (x_{0:t}^{(i)}) = \tilde{w}_t\ (x_{0:t}^{(i)}) / \sum_{i=1}^{N} \tilde{w}_t\ (x_{0:t}^{(i)}) \tag{7}$$

where the particles $\{x_{0:t}^{(i)}, w_t(x_{0:t}^{(i)})\}$ are drawn from the known distribution q, $\tilde{w}_t\ (x_{0:t}^{(i)})$ and $w_t\ (x_{0:t}^{(i)})$ are the un-normalized and normalized *importance weights*.

In order to propagate the particles $\{x_{0:t}^{(i)}, w_t(x_{0:t}^{(i)})\}$ through time, it is beneficial to develop a recursive calculation of the weights. This can be obtained straightforwardly by considering the following two facts:

1. Based on the definition of *filtering*, current states do not depend on future observations. That is,

   $$q(x_{0:t} \mid y_{1:t}) = q(x_{0:t-1} \mid y_{1:t-1}) q(x_t \mid x_{0:t-1}, y_{1:t})$$

2. As used in [8] and [17], the state dynamics is a Markov process and the observations are conditionally independent given the states, i.e.:

$$p(x_{0:t}) = p(x_0)\prod_{j=1}^{t} p(x_j \mid x_{j-1}),\ p(y_{1:t} \mid x_{0:t}) = \prod_{j=1}^{t} p(y_j \mid x_j)$$

Substituting the above two equations into Equation (6), we obtain the recursive estimate for the importance weights:

$$\begin{aligned} \tilde{w}_t^{(i)} &= \frac{p(y_{1:t} \mid x_{0:t}^{(i)})\, p(x_{0:t}^{(i)})}{q(x_{0:t-1}^{(i)} \mid y_{1:t-1})\, q(x_t^{(i)} \mid x_{0:t-1}^{(i)}, y_{1:t})} \\ &= \tilde{w}_{t-1}^{(i)} \frac{p(y_{1:t} \mid x_{0:t}^{(i)})\, p(x_{0:t}^{(i)})}{p(y_{1:t-1} \mid x_{0:t-1}^{(i)})\, p(x_{0:t-1}^{(i)})\, q(x_t^{(i)} \mid x_{0:t-1}^{(i)}, y_{1:t})} \\ &= \tilde{w}_{t-1}^{(i)} \frac{p(y_t \mid x_t^{(i)})\, p(x_t^{(i)} \mid x_{t-1}^{(i)})}{q(x_t^{(i)} \mid x_{0:t-1}^{(i)}, y_{1:t})} \end{aligned} \tag{8}$$

To summarize, in the sequential importance sampling step, there are two places involving the proposal distribution. First, particles are drawn from the proposal distribution (Equation (4)). Second, proposal distribution is used to calculate each particle's importance weight (i.e., Equation (8)).

Choosing the right proposal distribution is one of the most important issues in particle filter's design. In reality, there are infinite number of choices of the proposal distribution, as long as its support includes that of the posterior distribution, and it is easy to sample from. As pointed out in [12,14] and [17], the optimal proposal distribution is the one that minimizes the variance of the importance weights conditional on $x_{0:t-1}$ and $y_{1:t}$. In practice, however, finding the optimal proposal is very difficult if not impossible. Instead, the conventional particle filters have chosen to trade the optimality with easy-implementation by using the transition prior $p(x_t|x_{t-1})$ as the proposal distribution [6,8,18]. They sample from the transition prior and calculate the importance weight as follows:

$$\tilde{w}_t^{(i)} = \tilde{w}_{t-1}^{(i)} \frac{p(y_t \mid x_t^{(i)})\, p(x_t^{(i)} \mid x_{t-1}^{(i)})}{q(x_t^{(i)} \mid x_{0:t-1}^{(i)}, y_{1:t})} = \tilde{w}_{t-1}^{(i)} p(y_t \mid x_t^{(i)}) \tag{9}$$

Even though simple to implement, this proposal results in higher Monte Carlo variance and thus worse performance [5,17]. Comparing the transition prior $p(x_t|x_{t-1})$ with the general proposal distribution $q(x_t|x_{0:t-1},y_{1:t})$, we can easily see that the most recent observation y_t is missing in $p(x_t|x_{t-1})$. This may cause serious deficiency in particle filters, especially when the likelihood is peaked and the predicted state is near the likelihood's tail. The particles generated from the transition prior can therefore easily land on low-likelihood areas thus wasted. To overcome this difficulty, we will explore new ways of generating better proposal distribution in Section 3.

2.2. Selective re-sampling

Before we continue on discussing design better proposal distributions, we would like to first present the complete particle-filtering framework in the rest of this section. In addition to choosing better proposal distributions in the sequential importance sampling step, another crucial step in designing particle filters is re-sampling. One of the most important contributions made in the 1990s' particle-filter-renaissance is the introduction of the re-sampling step by Gordon *et. al.* [6]. Its philosophy is to eliminate particles with low importance weights and multiply particles with high importance weights, thus improving the effective particle size.

It can be proven [17] that without re-sampling the variance of the importance weight increases over time. In practice, this means one of the importance weights tends to one, while others become zero. That is, the effective particle size reduces from *N* to almost 1. This degeneracy phenomenon has been observed in several research fields [11,14,17]. In recent years, the re-sampling step has been adopted in almost all of today's particle filtering algorithms. However, cautions must be taken when using the re-sampling step: it should only be done when the effective particle size is small.

The effective particle size *S* can be estimated as follows [5,12,16]:

$$S = \left(\sum_{i=1}^{N} w_t (x_{0:t}^{(i)})^2\right)^{-1} \tag{10}$$

The value of *S* varies between 1 and *N*. When all the particles are of equal weight *1/N*, the effective particle size is *N*. When one particle is of weight 1 and rest are of weight zero, the effective particle size is 1. It is intuitive that when the weights are comparable to each other, re-sampling can only reduce the number of distinctive particles [14]. This suggests that one should not perform the re-sampling step when *S* is large. On the other hand, when the weights are very skewed (e.g., near the degeneracy case), many particles are wasted because of their close-to-zero weight, and the re-sampling step is required to increase the effective particle size. In practice, a pre-defined threshold S_T can be used, e.g., $S_T = N/2$, to determine if the re-sampling step is needed.

2.3. Complete algorithm for a generic particle filter

To generate better particles and to avoid degeneracy, we have discussed proposal distributions in Section 2.1 and the effective sample size in Section 2.2. To summarize, we give the complete algorithm for a generic particle filter in Figure 1.

3. Better proposal – the unscented particle filter

In Section 2.1, we have pointed out the deficiency of using the transition prior $p(x_t|x_{t-1})$ as the proposal distribution. The most obvious way to improve the proposal distribution is to incorporate the current observation data. Various

1. **Sequential importance sampling:**
   - a). Sample *N* particles $x_t^{(i)}$, *i = 1, 2, …, N*, from the proposal distribution *q(x_t|x_{0:t-1},y_{1:t})*. The proposal distribution can be the transition prior as used in traditional particle filters, or more advanced distributions discussed in Section 3.
   - b). Compute the particle weights using Equation (8)
   - c). Normalize the importance weight using Equation (7).
2. **Selective re-sampling:**
   - a). Compute the effective particle size *S* using Equation (9).
   - b). If $S < S_T$, multiple/suppress weighted particles to generate *N* equal-weighted particles.
3. **Output:**
   - a). Use Equation (4) to compute expectations of *g( )*. The conditional mean of x_t can be computed with $g_t(x_t) = x_t$, and conditional covariance of x_t can be computed with $g_t(x_t) = x_t x_t^T$. They can be readily used as the tracking results.

Figure 1. Algorithm for a generic particle filter.

Kalman filters are designed exactly for this purpose, though their performance varies depending on the different approximations they make. So far, the UKF is the best Kalman filter for non-linear systems. By using UKF to generate proposal distributions, we turn a generic particle filter to a high-performance unscented particle filter (UPF) [17]. In the rest of the section, we will first discuss the unscented transformation [10], the basis for UKF. We then give a complete UPF algorithm that uses the UKF to generate its proposal distribution.

3.1. Unscented transformation

In many applications, we need to estimate the low-order statistics, e.g., mean and covariance, of a random variable that undergoes a non-linear transformation *y = g(x).* The unscented transformation (UT) is an elegant way to accurately compute the mean and covariance up to the second order (third for Gaussian prior) of the Taylor series expansion of *g( )* [10,17]. Let n_x be the dimension of *x*, $\bar{x}$ be the mean of *x*, and P_x be the covariance of *x*, the UT computes mean and covariance of *y = g(x)* as follows:

1. Deterministically generate *2n_x+1 sigma points* $S_i=\{X_i, W_i\}$:

$$\begin{aligned}
&X_0 = \bar{x} \\
&X_i = \bar{x} + (\sqrt{(n_x+\lambda)P_x})_i \quad i = 1,\ldots,n_x \\
&X_i = \bar{x} - (\sqrt{(n_x+\lambda)P_x})_i \quad i = n_x+1,\ldots,2n_x \\
&W_0^{(m)} = \lambda/(n_x+\lambda), \quad W_0^{(c)} = W_0^{(m)} + (1-\alpha^2+\beta) \\
&W_i^{(m)} = W_i^{(m)} = 1/(2\cdot(n_x+\lambda)) \quad i = 1,\ldots,2n_x \\
&\lambda = \alpha^2(n_x+\kappa) - n_x
\end{aligned} \tag{11}$$

where κ is a scaling parameter that controls the distance between the sigma points and the mean $\bar{x}$. α is a positive scaling parameter that controls the higher order effects resulted from the non-linear function *g( )*. β is a parameter that controls the weighting of the *0th* sigma point. $\alpha =$, β =0 and κ =2 are the optimal values for the scalar case [17]. $(\sqrt{(n_x+\lambda)P_x})_i$ is the *ith* column of the matrix square root. Note that the *0th* sigma point's weight is different for calculating mean and covariance.

2. Propagate the sigma points through the nonlinear transformation:

$$Y_i = g(X_i) \quad i = 0,\ldots 2n_x \tag{12}$$

3. Compute the mean and covariance of *y* as follows:

$$\bar{y} = \sum_{i=0}^{2n_x} W_i^{(m)} Y_i, \quad P_y = \sum_{i=0}^{2n_x} W_i^{(c)} (Y_i - \bar{y})(Y_i - \bar{y})^T \tag{13}$$

The mean and covariance of *y* is guaranteed to be accurate up to the second order of the Taylor series expansion.

3.2. The unscented Kalman filter

The unscented Kalman filter (UKF) can be implemented using UT by expanding the state space to include the noise component: $x_t^a = [x_t^T m_t^T n_t^T]^T$. Let $N_a=N_x+N_m+N_n$ be the dimension of the expanded state space, where N_m and N_n are the dimensions of noise m_t and n_t, and *Q* and *R* be the covariance for noise m_t and n_t, the UKF can be summarized as follows [10,17]:

1. Initialization:

$$\bar{x}_0^a = [\bar{x}_0^T\, 0\, 0]^T, \quad P_0^a = \begin{bmatrix} P_0 & 0 & 0 \\ 0 & Q & 0 \\ 0 & 0 & R \end{bmatrix} \tag{14}$$

2. Iterate for each time instance t:
   - a). Calculate the sigma points using the procedure in Section 3.1:

$$X_{t-1}^a = [\bar{x}_{t-1}^a \quad \bar{x}_{t-1}^a \pm \sqrt{(n_a+\lambda)P_{t-1}^a}\,] \tag{15}$$

   - b). Time update:

$$X_{t|t-1}^x = f(X_{t-1}^x, X_{t-1}^v), \quad \bar{x}_{t|t-1} = \sum_{i=0}^{2n_a} W_i^{(m)} X_{i,t|t-1}^x \tag{16}$$

$$Y_{t|t-1} = h(X_{t|t-1}^x, X_{t-1}^n), \quad \bar{y}_{t|t-1} = \sum_{i=0}^{2n_a} W_i^{(m)} Y_{i,t|t-1}^x \tag{17}$$

$$P_{t|t-1} = \sum_{i=0}^{2n_a} W_i^{(c)} [X_{i,t|t-1}^x - \bar{x}_{t|t-1}][X_{i,t|t-1}^x - \bar{x}_{t|t-1}]^T \tag{18}$$

   - c). Measurement update:

$$P_{y_t y_t} = \sum_{i=0}^{2n_a} W_i^{(c)} [Y_{i,t|t-1} - \bar{y}_{t|t-1}][Y_{i,t|t-1} - \bar{y}_{t|t-1}]^T \qquad (19)$$

$$P_{x_t y_t} = \sum_{i=0}^{2n_a} W_i^{(c)} [X^x_{i,t|t-1} - \bar{x}_{t|t-1}][Y^x_{i,t|t-1} - \bar{y}_{t|t-1}]^T \qquad (20)$$

$$K_t = P_{x_t y_t} P_{y_t y_t}^{-1} \qquad (21)$$

$$\bar{x}_t = \bar{x}_{t|t-1} + K_t (y_t - \bar{y}_{t|t-1}), \quad P_t = P_{t|t-1} - K_t P_{y_t y_t} K_t^T \qquad (22)$$

Compared with the EKF [1], the UKF does not need to explicitly calculate the Jacobians or Hessians. Therefore, the UKF not only outperforms the EKF in accuracy (second order approximation vs. first order approximation), but also is computationally efficient. Its superior performance has been demonstrated in many applications [10,17].

3.3. Unscented particle filter

Till now, we have discussed both the UKF and the generic particle filters. For UKF, it can easily incorporate the most recent observation into the state estimation (e.g., measure update step in Section 3.2); however, it makes a Gaussian assumption of the state distribution. For the particle filters, on the other hand, they can model arbitrary distributions, but incorporating new observation y_t into the proposal distribution is not an easy task. The conventional particle filters simply ignore y_t, trading for easy implementation. To take advantage of the good features of both UKF and particle filters, and to avoid their limitations, we can use UKF to generate the proposal distribution for the particle filter, resulting the hybrid UPF [17]. Specifically, the proposal distribution for each particle is as follows:

$$q(x_t^{(i)} \mid x_{0:t-1}^{(i)}, y_{1:t}) = N(\bar{x}_t^{(i)}, P_t^{(i)}), \quad i = 1, \ldots, N \qquad (23)$$

where $\bar{x}_t$ and P_t are the mean and covariance of *x*, computed using UKF (Equations (14)-(22)). Note that, even though the Gaussian assumption is not realistic to approximate the posterior distribution *p(x*$_t$ *| x*$_{t-1,}$ *y*$_{0:t}$*)*, it is less a problem to generate each individual particles with distinct $\bar{x}_t$ and P_t. Furthermore, because UKF approximates the mean and covariance of the posterior up to the second order, the non-linearity of system is well preserved. The UPF algorithm is easily obtained by plugging the UKF step and Equation (23) into the generic particle filter algorithm. The complete UPF algorithm is summarized in Figure 2.

1. **Sequential importance sampling:**

a). Update particles $x_t^{(i)}$, *i = 1, …, N*, with the UKF using Equations (15)-(22) to obtain $\bar{x}_t^{(i)}$ and $P_t^{(i)}$.

b). Sample particles $x_t^{(i)}$, *i =1,…, N*, from the proposal distribution $q(x_t^{(i)} \mid x_{0:t-1}^{(i)}, y_{1:t}) = N(\bar{x}_t^{(i)}, P_t^{(i)})$

d). Compute the particle weights using Equation (8)

e). Normalize the importance weight using Equation (7).

……

(rest are the same as the generic particle filters in Figure 1)

Figure 3. The complete algorithm for UPF

So far we have discussed the UKF, and how we use UKF to generate the proposal distribution for UPF. In Sections 4 and 5, we will show how to apply the UPF framework to real-world applications where many practical considerations (e.g., observation ambiguity) need to be taken into account. To evaluate the performance of UPF, we also compare it against the widely used CONDENSATION approach that uses the transition priors as the proposal distribution. We describe an audio-data-based tracking system in Section 4 and a visual-data-based tracking system in Section 5.

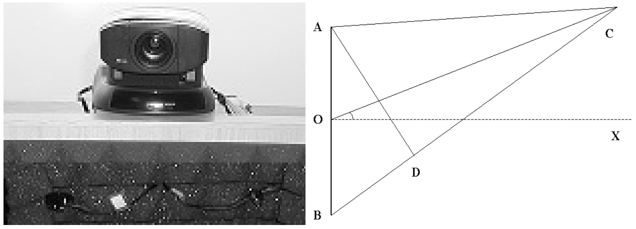
4. UPF tracking using audio sensory data

In many applications, including automated lecture rooms [15] and teleconferencing [19,20], we need to reliably track the location of the person who is talking. This is usually done by using a microphone array and a pan/tilt/zoom camera, as shown in Figure 1(a) [15]. The microphone array can estimate both the horizontal panning angle and the vertical tilting angle of the speaking person. For clarity, we will only focus on panning angle estimation in this section. Estimating the tilting angle follows the same approach.

In theory, two microphones are sufficient to estimate the panning angle. Referring to Figure 3(b), let the two microphones at locations A and B, and the sound source at location C. When the distance of the sound source, i.e., |OC|, is much larger than the length of the microphone pair baseline |AB|, the panning angle $\theta = \angle COX$ can be estimated as follows [15,20]:

$$\theta = \angle COX \approx \angle BAD = \arcsin \frac{|BD|}{|AB|} = \arcsin \frac{D \times v}{|AB|} \qquad (24)$$

where *D* is the time delay between the two microphones, and *v* = 342 m/s is the speed of sound traveling in air. There exists rich literature in time delay estimation in the signal processing community, where *D* is taken as the peak location in the generalized cross-correlation function (GCCF) [19,20]. This approach works well in low-noise non-reverberant environment. In reality, noise and reverberation causes “ghost” peaks in the GCCF causing this approach to break down. UPF provides a powerful framework to handle ghost peaks, and we explore such a



(a) **(b)**

Figure 2. (a) Microphones (lower portion of the figure) and the pan/tilt/zoom camera (upper portion of the figure). (b) Sound source localization.

solution in this section.

In order to utilize the UPF framework in a tracking application, four entities need to be established first: system dynamics $x_t = f(x_{t-1}, m_{t-1})$ to be used in Equation (16), system observation $y_t = h(x_t, n_t)$ to be used in Equation (17), likelihood $p(y_t|x_t)$ to be used in Equation (9), and innovation $y_t - \bar{y}_{t|t-1}$ to be used in Equation (22). Once these four entities are established, tracking proceeds straightforwardly using the UPF algorithm described in Figure 2.

4.1. System dynamics model $x_t = f(x_{t-1}, m_{t-1})$

Let $x = [\theta, \dot{\theta}]^T$ be the state space, where they are the panning angle and velocity of the panning angle, respectively. To model the movement dynamics of a talking person, we use the Langevin process $d^2\theta/dt^2 + \beta_\theta \cdot d\theta/dt = m$, whose discrete form is [19]:

$$\begin{bmatrix}\theta_t \\ \dot{\theta}_t\end{bmatrix} = \begin{bmatrix}1 & \tau \\ 0 & a\end{bmatrix}\begin{bmatrix}\theta_{t-1} \\ \dot{\theta}_{t-1}\end{bmatrix} + \begin{bmatrix}0 \\ b\end{bmatrix} m_t$$
$$a = \exp(-\beta_\theta \tau), \quad b = \bar{v}\sqrt{1-a^2} \tag{25}$$

where β_θ is the rate constant, m is a thermal excitation process drawn from $N(0,Q)$, τ is the discretization time step, and $\bar{v}$ is the steady-state root-mean-square velocity.

4.2. System observation model $y_t = h(x_t, n_t)$

Our system observation y_t is the time delay D_t. Based on Equation (24), the observation relates to the state by

$$y_t = D_t = h(\theta_t, n_t) = |AB| v \sin\theta_t + n_t \tag{26}$$

where n_t is the measurement noise, obeying a Gaussian distribution of $N(0, R)$.

4.3. **Likelihood model** $p(y_t|x_t)$

Because of the noise and reverberation, there is no simple expression for the likelihood model. Let J be the number of peaks in the GCCF. Of the J peak locations, at most one is from the true sound source. Following similar approaches used in [8] and [19], we can therefore define $J+1$ hypotheses:

$$H_0 = \{c_j = C : j = 1, \ldots, J\}$$
$$H_j = \{c_j = T, c_k = C : k = 1, \ldots, J, k \neq j\} \tag{27}$$

where $c_j=T$ means the j^{th} peak is associated with the true sound source, $c_j=C$ otherwise. Hypothesis H_0 therefore means that none of the peaks is associated with the true source. The combined likelihood model is therefore:

$$p(y_t | x_t) = \pi_0 p(y_t | H_0) + \sum_{j=1}^{J} \pi_j p(y_t | H_j)$$
$$= \pi_0 U + N_m \sum_{j=1}^{J} \pi_j N(D_j, \sigma_D) \tag{28}$$
$$s.t. \quad \pi_0 + \sum_{j=1}^{J} \pi_j = 1$$

where π_0 is the prior probability of hypothesis H_0, π_j, $j = 1, \ldots, J$, can be obtained from the relative height of the j^{th} peak, N_m is a normalization factor, D_j is the time delay corresponding the j^{th} peak, U represents the uniform distribution and $N(\,)$ represents the Gaussian distribution.

4.4. Innovation model $y_t - \bar{y}_{t|t-1}$

The same as the likelihood model, the innovation model also needs to take into account the multi-peak fact:

$$y_t - \bar{y}_{t|t-1} = \sum_{j=1}^{J} \pi_j (D_j - \bar{y}_{t|t-1}) \tag{29}$$

where $\bar{y}_{t|t-1}$ is the predicted measurement obtained from UKF (see Equation (22)).

4.5. Experiments

The previous sub-sections have developed the system dynamics, measurement, likelihood, and innovation models. Note how we handle the measurement ambiguity in the likelihood model and innovation model by using multi-hypothesis approach. To evaluate UPF's tracking performance, we compare it with the CONDENSATION approach. The experiment is in a normal office, where various noise and reverberation exist: PC fan noise, hard drive noise, central air conditioner noise, occasional traffic noise, desk's flat-surface reverberation, book shelf surface reverberation, and wall corner reverberation. This is a very challenging environment for audio-based speaker localization. The two microphones are placed 24cm apart from each other. The speaker is about 1.5m away from the microphones. Our software is developed in C++ on Windows 2000 platform. No optimization is attempted and the system runs comfortably in real-time with N=100 and J=10. We report a typical tracking result in Figure 4. The whole test sequence is 33s long. During 0s-4s, 7s-20s and 24s-33s, the speaker is talking. During the entire 33s, the speaker is constantly moving left and right. The solid curve is the ground truth of where the speaker is. The dashed and

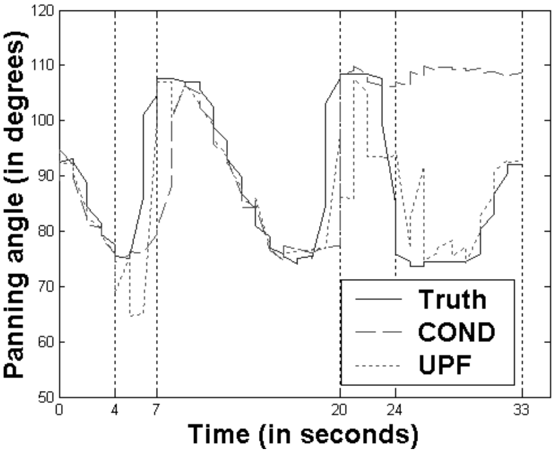


Figure 4. Speaker tracking – a comparison between UPF and standard CONDENSATION with transition prior.

dotted curves are the tracking results from CONDENSATION and UPF, respectively. We have the following observations:

1. When the new observation y_t and transition prior $p(x_t|x_{t-1})$ overlaps, e.g., 0s-4s, both algorithms work well.
2. When the new observation y_t is not too far away from the transition prior $p(x_t|x_{t-1})$, e.g., 7s-20s, both algorithm can still track the speaker, but CONDENSATION is considerably slower, i.e., 7s-9s.
3. When observation y_t is far away from the transition prior $p(x_t|x_{t-1})$, e.g., 24s-33s, UPF is still able to resume tracking after a few seconds, because its proposal distribution generated by UKF takes into account the most recent observation. But CONDENSATION is stuck to a wrong location and never comes back.
4. When the person is not talking, e.g., 4s-7s and 20s-24s, CONDENSATION mostly stays at its old location while UPF searches around. This is because even though the person is not talking, other background noise may still produce small sound sources. The UPF searches based on the new observation -- sometimes accidentally moves in the same direction as the person, e.g., 20s-24s, sometimes totally opposite, e.g., 4s-7s. But as soon as the person starts to talk, UPF resumes tracking.

5. UPF tracking using visual sensory data

Reliable human tracking in cluttered environment has many real-world applications [8,15]. Human head can be modeled by a 1:1.2 ellipse and hence be handled as a parametric contour. One difficulty in contour tracking is the high non-linearity of the likelihood model $p(y_t|x_t)$. Even a small difference in the parametric space could result in large changes in the observation likelihood. Therefore, it is imperative to distribute limited particles in an effective way, which will benefit greatly from a better proposal distribution.

5.1. System dynamics model $x_t = f(x_{t-1}, m_{t-1})$

Let (r, s) represent the image coordinate. In our contour-based tracking, the system states are the position of the ellipse center and its horizontal and vertical velocity, i.e., $x_t = [r_t, s_t, \dot{r}_t, \dot{s}_t]^T$. Similar to Section 4.1, we adopt the Langevin process to model the human movement dynamics:

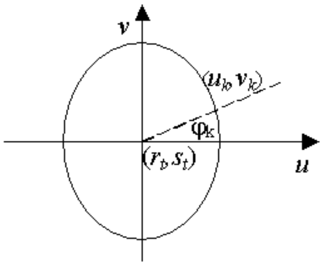


Figure 5. The ellipse and the rays.

$$\begin{bmatrix} r_t \\ s_t \\ \dot{r}_t \\ \dot{s}_t \end{bmatrix} = \begin{bmatrix} 1 & 0 & \tau & 0 \\ 0 & 1 & 0 & \tau \\ 0 & 0 & a_r & 0 \\ 0 & 0 & 0 & a_s \end{bmatrix} \begin{bmatrix} r_{t-1} \\ s_{t-1} \\ \dot{r}_{t-1} \\ \dot{s}_{t-1} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ b_r \\ b_s \end{bmatrix} m_t \qquad (30)$$

5.2. System observation model $y_t = h(x_t, n_t)$

Refer to Figure 5, the ellipse is centered at the current state location (r_t,s_t). We generate K rays from the ellipse center and let them intersect with the ellipse boundary. If we use the ellipse center as the origin of a local coordinate system, the intersections (u_k, v_k), $k = 1, 2, \ldots, K$, can be obtained as

$$\begin{aligned} u_k &= \sqrt{\tan^2 \varphi_k / (1.44 \tan^2 \varphi_k + 1)} \\ v_k &= \sqrt{1/(1.44 \tan^2 \varphi_k + 1)} \end{aligned} \qquad (31)$$

by jointly solving the ellipse equation and the ray equation:

$$\begin{cases} \dfrac{u_k^2}{1} + \dfrac{v_k^2}{1.2^2} = 1 \\ u_k = v_k \tan(\varphi_k) \end{cases} \qquad (32)$$

Transforming the local (u, v) coordinate back to the image coordinate, we obtain the observation:

$$\begin{aligned} y_t &= h(x_t, n_t) \\ &= [(u_k + r_t, v_k + s_t)] + n_t, \quad k = 1, 2, \ldots, K. \end{aligned} \qquad (33)$$

where n_t is the measurement noise, obeying a Gaussian distribution of $N(0, R)$. Note that the observation model is highly non-linear.

5.2. Likelihood model $p(y_t|x_t)$

We use the edge intensity to model the state likelihood. Along each of the K rays, we use Canny edge detector to calculate the edge intensity. The resulting function is a multi-peak function, just like the GCCF in Section 4.3. The multiple peaks signify there are multiple edge candidates along this ray. Let the number of peaks be J, we can use the same likelihood model developed in Section 4.3 to model the edge likelihood along ray k:

$$\begin{aligned} p^{(k)}(y_t | x_t) &= \pi_{k0} p^{(k)}(y_t | H_0) + \sum\nolimits_{j=1}^{J} \pi_{kj} p^{(k)}(y_t | H_j) \\ &= \pi_{k0} U + N_m \sum\nolimits_{j=1}^{J} \pi_{kj} N((u_k, v_k)_j, \sigma_{kj}) \end{aligned}$$

The overall likelihood considering all the K rays is therefore:

$$p(y_t | x_t) = \prod\nolimits_{k=1}^{K} p^{(k)}(y_t | x_t) \qquad (34)$$

5.3. Innovation model $y_t - \bar{y}_{t|t-1}$

The same as the likelihood model, the innovation model also needs to take into account the multi-peak fact:

$$y^{(k)}{}_t - \bar{y}^{(k)}{}_{t|t-1} = \sum\nolimits_{j=1}^{J} \pi_{kj} ((u_k, v_k)_{t,j} - (u_k, v_k)_{t|t-1})$$

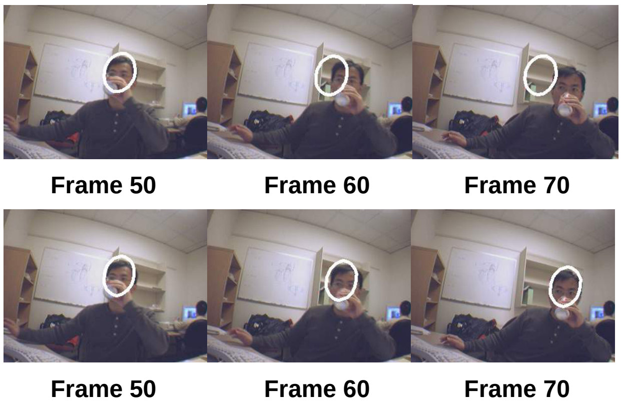


Figure 6. Tracking results. Top row is based on CONDENSATION and bottom row is based on UPF.

where $k = 1, 2, \ldots, K$, π_{kj} is the mixing weight for the j^{th} peak along ray k, and can be obtained from the corresponding edge intensity.

5.4. Experiments

Our tracking system is developed in C++ on Windows 2000 platform. No optimization is attempted and the system runs comfortably at 30 frames/sec with N=30 and J=5. The image resolution is 320x240. The experiments are conducted in normal offices, with bookshelves, PC monitors, and other people in the background. For more tracking sequences, please refer to our supplement material *970.zip* submitted to the conference. We report two typical tracking sequences here. Figures 6 and 7 show the tracking results using CONDENSATION and UPF. In both figures, when the person moves to a location that is not the same as the transition prior predicts, CONDENSATION is easily distracted by background clutter (e.g., the bookshelf in Figure 6 and the PC in Figure 7), because no current observation is taken into account. On the other hand, because UPF's superior proposal distribution places the limited particles more effectively, it tracks both sequences successfully.

6. Concluding remarks

In this paper, we applied a new formulation of the particle filter framework in object tracking, which emphasizes the important role played by the proposal distribution. This new formulation shows us how we can improve particle filter's performance by designing better proposal distributions. We have further shown how to apply the general UPF framework in real-world problems through two tracking applications. Experimental results of both applications demonstrate the superior performance of UPF over the conventional particle filters such as CONDENSATION.

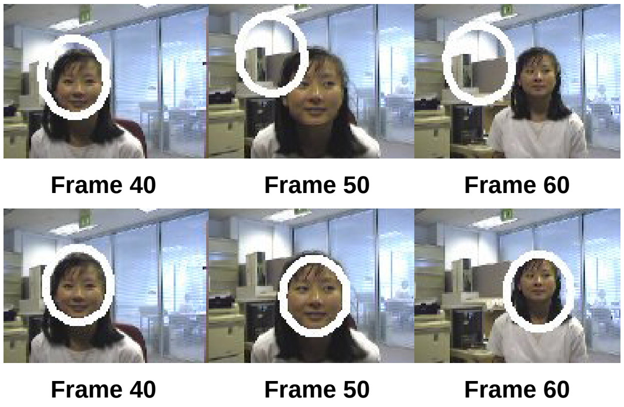


Figure 7. Tracking results. Top row is based on CONDENSATION and bottom row is based on UPF.

7. Acknowledgement

The authors would like to thank S. Birchfield for allowing us to use the test sequence, and thank R. Merwe, A. Doucet, N. Freitas, and J. Vermaak for valuable discussions.

8. References:

1. Anderson, B., and Moore, J., Optimal filtering, Englewood Cliffs, Prentice Hall, New Jersy, 1979.
2. Black, M., and Jepson, A. D., A probabilistic framework for matching temporal trajectories: CONDENSATION-based recognition of gestures and expressions, *Proc. 5th European Conf. Computer Vision*, 1998, pp. 909-924.
3. Blake A., Isard, M., and Reynard, D., Learning to track the visual motion of contours. *Artificial Intelligence*, 78, pp. 101-134.
4. Crisan, D., and Doucet, A., Convergence of sequential Monte Carlo methods, *Technical Report CUED/F-INFENG/TR381*, Dept. of Engineering, University of Cambridge, 2000.
5. Doucet, A., On sequential simulation-based methods for Bayesian filtering. *Technical Report CUED/F-INFENG/TR310*, Dept. of Engineering, University of Cambridge, 1998.
6. Gordon, N., Salmond, D., and Smith, A., Novel approach to non-linear/non-Gaussian Bayesian state estimation, *IEEE Trans. Radar, Signal Processing*, 1993, Vol. 140, pp. 107-113.
7. Hammersley, J. M., and Morton, K. W., Poor man's Monte Carlo, *Journal of the Royal Statistical Society B*, 16, pp. 711-732.
8. Isard, M., and Blake, A., Visual tracking by stochastic propagation of conditional density. *Proc. 4th European Conf. Computer Vision*, pp 343-356, Apr. 1996
9. Isard, M., and Blake, A., ICONDENSATION: Unifying low-level and high-level tracking in a stochastic framework, *Proc. 5th European Conf. Computer Vision*, 1998, pp.893-908.
10. Julier, S. J., and Uhlmann, J. K., A general method for approximating nonlinear transformations of probability distributions, *Technical report*, RRG, Dept. of Engineering Science, University of Oxford.
11. King, O., and Forsyth, D. A., How does CONDENSATION behave with a finite number of samples?, *Proc. of ECCV, 2000, LNCS 1842*.
12. Kong, A., Liu, J. S., and Wong, W. H., Sequential imputations and Bayesian missing data problems, *Jour. of Amer. Stat. Assoc.* Vol. 89, pp. 278-288, 1994.
13. Li, B., and Chellappa, R., Simultaneous tracking and verification via sequential posterior, *Proc. of IEEE CVPR 2000*, pp. 110-117.
14. Liu, J., and Chen, R., Sequential Monte Carlo methods for dynamic systems, *Jour. of Amer. Stat. Assoc.* Vol. 93, pp. 1031-1041, 1998.
15. Liu, Q., Rui, Y., Gupta, A., and Cadiz, J.J., Automating camera management in a lecture room environment, Proc. of ACM Computer-Human Interaction (CHI) 2000.
16. MacCormick, J., and Isard, M., Partitioned sampling, articulated objects, and interface-quality hand tracking, *Proc. ECCV 2000*, LNCS 1831.
17. Merwe, R., Doucet, A., Freitas, N., and Wan, E., The unscented particle filter, *Technical Report CUED/F-INFENG/TR 380*, Cambridge University Engineering Department, August 2000.
18. Vermaak, J., Andrieu, C., Doucet, A., Particle filtering for non-stationary speech modeling and enhancement, *Proc. of IEEE ICSLP 2000*.
19. Vermaak, J., and Blake A., Nonlinear filtering for speaker tracking in noisy and reverberant environments, *Proc. of IEEE ICASSP*, 2000.
20. Wang, H., and Chu, P., Voice source localization for automatic camera pointing system in video conferencing, *ICASSP'97*