Visual Contour Tracking Based on Particle Filters

Peihua Li, Tianwen Zhang

Dept. of Computer Science and Engineering 360, Harbin Institute of Technology, Harbin, Hei Long Jiang Province, P.R. China, 150001 Email: peihualj@hotmail.com

***Abstract*—In the computer vision community, the Condensation algorithm has received considerable attention. Recently, it has been proven that the algorithm is one variant of particle filter (also known as sequential Monte Carlo filter, sequential importance sampling etc.). In sampling stage of Condensation, particles are drawn from the prior probability distribution of the state evolution transition, without making use of the most current observations, therefore, the algorithm demands a large number of particles and is computationally expensive. In this paper, a Kalman particle filter and an Unscented particle filter are presented to try to overcome the problem. These filters adopt sub-optimal proposal distributions, and use the Kalman filter or Unscented Kalman filter to incorporate the newest observation. This kind of sampling strategy can effectively steer the set of particles towards the region with high likelihood, and therefore, can considerably reduce the number of particles needed. Experiments with real image sequence are made to compare the performance of the three algorithms: Condensation, Kalman particle filter, and Unscented particle filter.**

***Index Terms*—Contour tracking, Kalman filter, Particle filter, Unscented Kalman filter, image sequences.**

I. INTRODUCTION

PROBABILISTIC visual contour tracking has been an active research area in the computer vision community in the last ten or more years. It has many potential applications in intelligent robots, in monitoring and surveillance, in biomedical image analysis, in human-computer interfaces, etc. [1]. For these tracking tasks, a common approach is the use of the Kalman filter or extended Kalman filter. While some researchers employ physical snakes as system models in the (extended) Kalman filter [2,3,4], others use constant velocity motion models or learned motion models from training image sequences [5,6].

All the research mentioned above assumed that the probability distributions of the states are Gaussian, and therefore, the means and covariances, computed recursively with a set of (extended) Kalman filter equations, can fully characterise the behaviour of the tracked targets. They thus preclude the possibility of capturing a non-Gaussian distribution of states. In visual contour tracking, many factors exist which lead to the problems of non-Gaussianity and non-linearity, including complex motion models and state representations, and most important of all, a non- linear observation process due to visual clutter [7,9].

Recently, the Condensation algorithm, as an extension of factored sampling [8], has been introduced to solve non-Gaussian, nonlinear contour tracking problems in image sequences [9]. Simultaneously and independently, particle filters, such as the sequential Monte Carlo filter, sequential importance sampling, etc., have been developed to address non-Gaussianity and nonlinearity in science and engineering [17,18,14,24]. It has been proven that they all belong to the same theoretical framework, and that Condensation is also a variant of the theory [10]. Hereafter, we shall adopt the term *particle filter* to include all of these variants.

The basic idea of the particle filter is that the posterior density is approximated by a set of discrete samples (called particles) with associated weights. The algorithm generally involves three steps. The first step is the sampling step, in which particles are drawn from a proposal distribution and are re-weighted by their likelihood according to Bayesian rules. The second step outputs the estimates of the state, such as mean and covariance. In the last step, particles are resampled to ensure a uniform weight distributions.

In the first step of Condensation, the proposal distribution from which particles are drawn is the transition prior, i.e., the probabilistic model of the state evolution. Since no use is made of the newest measurement, a large number of particles may be required to represent the posterior distribution, especially in situations where the new measurements appear in the tail of the prior, or if the likelihood is strongly peaked. Much effort has been made to address the problem. Isard et al [11] proposed an importance sampling method, in which an auxiliary blob tracker is combined into Condensation; but in many cases an auxiliary tracker may not be obtained. MacCormick et al [12] developed partitioned sampling, which, however, requires that the state-space can be sliced. Sullivan et al [13] proposed layered sampling using multi-scale processing of images.

In the field of Neural Network tracking [16] and filtering theory [19] outside computer vision, the Kalman particle and Unscented particle filters have been developed to improve the sampling mechanism. They adopt sub-optimal proposal distributions, using the Kalman filter or unscented Kalman filter to integrate the most current observations. The strategy is able to steer particles towards a region with high likelihood and therefore reduces the number of particles needed in the

algorithm. In this paper, we describe the implementation of the two algorithms for visual contour tracking.

The structure of the paper is as follows. Section II briefly introduces the general particle filter algorithm. In Section III, after describing the motion model and measurement model, we present the Kalman particle and Unscented particle filters in the framework of visual curve tracking. Section IV explains the concept of "critical" size and compares the performance of the three algorithms. Section V contains concluding remarks.

II. Standard Particle Filter

A. The General Target Tracking Problem

Target tracking can be characterised as the problem of estimating the state of a system as a set of observations become available on-line. In the Bayesian filtering framework, we define the densities

$$p(X_k \mid X_{k-1}),\ p(X_0) \quad \text{for} \quad k \geq 1 \tag{1}$$

$$p(Y_k \mid X_k) \quad \text{for} \quad k \geq 1 \tag{2}$$

where the form of the transition prior $p(X_k \mid X_{k-1})$ indicates that the evolution of the state is a Markov process, and $p(Y_k \mid X_k)$ denotes the observation density (likelihood) in the dynamical system, which is conditionally independent given the states.

Our aim is to estimate recursively in time the filtering density $p(X_k \mid Y_{1:k})$, where $Y_{1:k} = \{Y_1, \cdots, Y_k\}$, and the associated expectation of some integral function $f_k(X_k)$. The filtering density (also called posterior density) is estimated in two stages: prediction and update. In the prediction step, the filtering density $p(X_{k-1} \mid Y_{1:k-1})$ is propagated into the future via the transition density $p(X_k \mid X_{k-1})$ as follows:

$$p(X_k \mid Y_{1:k-1}) = \int p(X_k \mid X_{k-1})\, p(X_{k-1} \mid Y_{1:k-1})\, dX_{k-1} \tag{3}$$

The update stage involves the application of Bayes' rule when new data are observed:

$$p(X_k \mid Y_{1:k}) = \frac{p(Y_k \mid X_k)\, p(X_k \mid Y_{1:k-1})}{p(Y_k \mid Y_{1:k-1})} \tag{4}$$

Then the associated expectation can be computed as

$$E_{p(\bullet \mid Y_{1:k})}(f_k(X_k)) = \int f_k(X_k)\, p(X_k \mid Y_{1:k})\, dX_k \tag{5}$$

The prediction and update strategy (3) and (4) provides an optimal solution to the inference, which, unfortunately, involves high-dimensional integration. In most general cases involving non-Gaussianity and nonlinearity, analytical solutions are impossible, so we resort to Monte Carlo methods.

In the particle filter paradigm, drawing samples directly from the posterior density at the current time-step is impossible in most cases. Therefore, a proposal density $\pi(X_k \mid Y_{1:k})$ is used, from which particles can be easily drawn, and which is similar to the posterior density. To facilitate sequential estimation of the system state [14, 18], the proposal density must satisfy the following equation:

$$\pi(X_k \mid Y_{1:k}) = \pi(X_k \mid X_{k-1}, Y_{1:k})\, \pi(X_{k-1} \mid Y_{1:k-1}) \tag{6}$$

It can be seen that the proposal density integrates the most recent observation.

It is convenient to introduce the weight function $\omega(X_k) = \omega_k$ that can be computed sequentially as

$$\omega_k = \omega_{k-1} \frac{p(Y_k \mid X_k)\, p(X_k \mid X_{k-1})}{\pi(X_k \mid X_{k-1}, Y_{1:k})} \tag{7}$$

The posterior distribution can be represented by the weight function up to a proportionality constant, and the estimates can be computed as

$$E_{p(\bullet \mid Y_{1:k})}(f_k(X_k)) = \frac{E_{\pi(\bullet \mid Y_{1:k})}(\omega_k g_k(X_k))}{E_{\pi(\bullet \mid Y_{1:k})}(\omega_k)} \tag{8}$$

How to choose efficient proposal distribution is an active research topic in particle filtering. Doucet et al proposed the optimal proposal distribution (OPD) which minimises the variance of the weights ω_k [20]. However, the design of OPD is a non-trivial task. The rejection method [14] and the method utilising the auxiliary variable[15] can approximate the OPD, but they are computationally inefficient, as discussed in the literature. The method of local linearization of OPD is often used. The Kalman particle filter or Unscented particle filter are also local linearisation methods, where the proposal distribution is Gaussian. More specifically, for each discrete particle $X_k^{(i)}$, a separate Kalman filter or Unscented Kalman filter is used to generate and propagate the Gaussian proposal distribution:

$$\pi(X_k^{(i)} \mid X_{k-1}^{(i)}, Y_{1:k}) = N(\hat{X}_k^{(i)}, \hat{P}_k^{(i)}) \quad i = 1, \cdots, N \tag{9}$$

where $N(\hat{X}_k^{(i)}, \hat{P}_k^{(i)})$ denotes Gaussian with mean $\hat{X}_k^{(i)}$ and covariance $\hat{P}_k^{(i)}$. Detailed information on the proposal distribution is given in a later section.

B. The General Particle Filter

The basic idea of particle filtering is Monte Carlo simulation, in which the posterior density is approximated by a set of particles with associated weights $\{(X_{k-1}^{(i)}, \omega_{k-1}^{(i)}) \quad i = 1, \cdots, N\}$. At every time step we sample from the proposal distribution $\pi(X_k^{(i)} \mid X_{k-1}^{(i)}, Y_{1:k})$ to achieve new particles and compute new weights according to the particle likelihoods. After normalisation of weights, the posterior density can be represented by $\{(X_k^{(i)}, \omega_k^{(i)}) \quad i = 1, \cdots, N\}$.

The standard particle filter is as follows:

The Standard Particle Filter

1. Initialisation

   Draw a set of particles from the prior $p(X_0)$ to obtain $\{(X_0^{(i)}, \omega_0^{(i)}), i=1, \cdots, N\}$, let $k=1$

2. Sampling step

   a) For $i=1, \cdots, N$

   Sample $X_k^{(i)}$ from the proposal distribution $\pi(X_k^{(i)} \mid X_{k-1}^{(i)}, Y_{1:k})$

   b) Evaluate the new weights

   $$\omega_k^{(i)} = \omega_{k-1}^{(i)} \frac{p(Y_k \mid X_k^{(i)}) p(X_k^{(i)} \mid X_{k-1}^{(i)})}{\pi(X_k^{(i)} \mid X_{k-1}^{(i)}, Y_{1:k})}, \qquad i=1, \cdots, N$$

   c) Normalise the weights

   $$\omega_k^{(i)} = \omega_k^{(i)} \Big/ \sum_{j=1}^{N} \omega_k^{(j)}, \quad i=1, \cdots, N$$

3. Output step

   Output a set of particles $\{(X_k^{(i)}, \omega_k^{(i)}), i=1, \cdots, N\}$ that can be used to approximate the posterior distribution as $p(X_k \mid Y_{1:k}) \approx \sum_{i=1}^{N} \omega_k^{(i)} \delta(X_k - X_k^{(i)})$ and the estimate as

   $E_{p(\bullet \mid Y_{1:k})}(f_k(X_k)) \approx \sum_{i=1}^{N} \omega_k^{(i)} f_k(X_k^{(i)})$, where $\delta(\bullet)$ is the Dirac function.

4. Selection (resampling) step

   Resample particles $X_k^{(i)}$ with probability $\omega_k^{(i)}$ to obtain N i.i.d random particles $X_k^{(j)}$, approximately distributed according to $p(X_k \mid Y_{1:k})$.

   Set $\omega_k^{(i)} = 1/N$, $i=1, \cdots, N$.

5. $k = k+1$, go to step 2.

III. Visual Contour Tracking Based on Particle Filter

The dynamical model in this paper follows that described in [6], [9]. The motion model transition prior is a temporal Markov Chain

$$p(X_k \mid X_{k-1}) \propto \exp\{-\frac{1}{2}[(X_k - \bar{X}) - A(X_{k-1} - \bar{X})]^T Q^{-1} \times [(X_k - \bar{X}) - A(X_{k-1} - \bar{X})]\} \quad (10)$$

where A and Q are learned from training image sequences and $\bar{X}$ is a constant vector, indicating the mean shape of the object. Equation (10) can also be equivalently represented as the AR(2) process

$$X_k - \bar{X} = A(X_{k-1} - \bar{X}) + w_k \quad (11)$$

where w_k is Gaussian and independent of the state, and satisfies $E[w_k] = 0, \quad E[w_{k_1} w_{k_2}^T] = \delta(k_1 - k_2) Q$.

The measurement proceeds as follows: for each sampled point $r(s_l)(l=1, \cdots m)$ on the curve, search along the normal to the curve. In general, more than one feature $z_j^{(l)}$ $(j=1, \cdots, n_l)$ will be detected, due to clutter: the feature $\tilde{z}_l$ with maximum contrast is selected as the "true" feature. Provided that the detected features $z_j^{(l)}$ follow a spatial Poisson distribution, the 1-D measurement density along the line normal to $r(s_l)$ can be modeled as

$$p_l(z \mid u = \{u(l)\}) = const \times (q_{01} + \frac{q_{11}}{\lambda} \sum_{j=1}^{n_l} \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{1}{2\sigma^2}(z_j^{(l)} - u(l))^2) \quad (12)$$

where n_l is the number of features detected along the normal, q_{01} is the probability that no feature is detected, $q_{11} = 1 - q_{01}$, λ is the Poisson constant, and $u(l)$ is the search scale on each side of $r(s_l)$. Assuming that feature outputs on distinct normal lines are statistically independent, the overall measurement density becomes

$$p(Y_k \mid X_k) = \prod_{l=1}^{m} p_l(z \mid u = \{u(l)\}) \quad (13)$$

Full details about the measurement model can be found in [9,21]. Each detected feature $\tilde{z}_l$ is a point in the image, with coordinates x_l and y_l. For implementation, we rearrange the measurements into a vector:

$$Y = [x_1 \quad y_1 \quad x_2 \quad y_2 \quad \cdots \quad x_m \quad y_m]^T \quad (14)$$

The measurement noise can be reasonably described as

$$R^n = diag(\sigma^2 \quad \cdots \quad \sigma^2) \quad (15)$$

A. Condensation Algorithm for Contour Tracking

The pioneering work on Condensation has had considerable success for curve tracking in dense clutter[7,9]. The algorithm is essentially a variant of particle filter, in which the proposal distribution is the transition prior, i.e.,

$$\pi(X_k^{(i)} \mid X_{k-1}^{(i)}, Y_{1:k}) = p(X_k^{(i)} \mid X_{k-1}^{(i)}) \quad (16)$$

Hence, the weight becomes

$$\omega_k^{(i)} = \omega_{k-1}^{(i)} p(Y_k \mid X_k^{(i)}) \quad (17)$$

The Condensation algorithm is briefly described as follows, in which differences from the standard particle filter are emphasized.

Condensation Algorithm

1. Initialisation
2. Sampling step
   a) For $i = 1, \cdots, N$

      Sample $X_k^{(i)}$ from the transition prior $p(X_k^{(i)} \mid X_{k-1}^{(i)})$ (10)

   b) Evaluate the new weights according to (13)

      $\omega_k^{(i)} = p(Y_k \mid X_k^{(i)}), \quad i = 1, \cdots, N$

   c) Normalise the weights
3. Output step
4. Selection (resampling) step
5. $k = k + 1$, go to step 2.

B. Kalman Particle Filter for Curve Tracking

In this algorithm, for each discrete $X_{k-1}^{(i)}$, a separate Kalman filter is used to generate the Gaussian proposal distribution (integrating the most current observation) from which the new particle $X_k^{(i)}$ is drawn. The Kalman filter includes linear state-transition and measurement equations. Instead of linearizing the nonlinear measurement equation (13), which is unrealistic, we adopt a linear model as follows:

$$\upsilon_{k,l} = H(\hat{X}_k - X_k) + v_k \quad l = 1, \cdots, m \quad (18)$$

where H is measurement matrix, $\upsilon_{k,l}$ is lth observation innovation in the current time-step k, v_k is scalar measurement noise with $E(v_k) = 0, \quad E(v_{k_1} v_{k_2}^T) = \delta(k_1 - k_{2)} \sigma_1^2$.

In practice, the measurement process is similar to that in the nonlinear measurement model (13). Full details about measurement equation (18) and measurement process can be found in [6].

The Kalman particle algorithm is as follows, where differences from the standard particle filter are emphasised.

The Kalman Particle Algorithm

1. Initialisation

   Draw the particles from the prior $p(X_0)$ to obtain $\{(X_0^{(i)}, \ \omega_0^{(i)}, \ P_0^{(i)}), \ i = 1, \cdots, N\}$, let $k = 1$
2. Sampling step
   a) For $i = 1, \cdots, N$,

      Let $\hat{P}_{k-1|k-1}^{(i)} = P_{k-1}^{(i)}$, $\hat{X}_{k-1|k-1}^{(i)} = X_{k-1}^{(i)}$

      $\hat{X}_{k|k-1}^{(i)} = A\hat{X}_{k-1|k-1}^{(i)} + (I - A)\overline{X}$

      $\hat{P}_{k|k-1}^{(i)} = A\hat{P}_{k-1|k-1}^{(i)} A^T + Q$

      Let $\hat{X}_{k|k}^{(i)} = \hat{X}_{k|k-1}^{(i)}$, $\hat{P}_{k|k}^{(i)} = \hat{P}_{k|k-1}^{(i)}$

      For all the observation innovations $\upsilon_{k,l}^{(i)}, l = 1, \cdots, m$ concerning the predicted mean $\hat{X}_{k|k-1}^{(i)}$, make Kalman updates recursively

      $\hat{X}_{k|k}^{(i)} = \hat{X}_{k|k}^{(i)} + K\upsilon_{j,k}^{(i)}$

      $\hat{K}_k^{(i)} = \hat{P}_{k|k}^{(i)} H^T (H\hat{P}_k^{(i)} H^T + \sigma_1^2)^{-1}$

      $\hat{P}_{k|k}^{(i)} = (I - \hat{K}_k^{(i)})\hat{P}_{k|k}^{(i)}$

   Sample $X_k^{(i)}$ from $\pi(\hat{X}_k^{(i)} \mid \hat{X}_{k-1}^{(i)}, Y_{1:k}) = N(\hat{X}_{k|k}^{(i)}, \hat{P}_{k|k}^{(i)})$, let $P_k^{(i)} = \hat{P}_{k|k}^{(i)}$

Continues on next page

b) Evaluate the new weights

$$\omega_k^{(i)} = \omega_{k-1}^{(i)} \frac{p(Y_k \mid X_k^{(i)}) p(X_k^{(i)} \mid X_{k-1}^{(i)})}{\pi(X_k^{(i)} \mid X_{k-1}^{(i)}, Y_{1:k})}, \qquad i = 1, \quad \cdots, \quad N$$

Normalise the weights

3. Output step

Output a set of particles $\{ (X_k^{(i)}, \quad \omega_k^{(i)}, \quad P_k^{(i)}) \ , i = 1, \quad \cdots, \quad N \}$, and compute the mean $E_{p(\bullet \mid Y_{1:k})}(X_k)$.

4. Selection (resampling) step
5. $k = k+1$, go to step 2.

In stage a) of sampling step 2, for each particle $(X_{k-1}^{(i)}, \quad \omega_{k-1}^{(i)}, \quad P_{k-1}^{(i)})$, predicted mean and covariance are computed, according to motion model (11); then measurements are made to obtain *m* observations about the predicted mean state, and Kalman updates are made recursively for the *m* innovations $\upsilon_{l,k}^{(i)}, l = 1, \quad \cdots, \quad m$. Finally, the updated mean and covariance determine the Gaussian proposal distribution, from which the new particle $X_k^{(i)}$ is drawn. The most current observation is naturally integrated into the sampling step, which can effectively steer the newly drawn particle .

The Unscented Kalman filter was first proposed by Julier et al to address nonlinear state estimation in the context of control theory [22]. The algorithm uses a set of carefully chosen *sigma vectors* to capture mean and covariance of the system. The vectors are propagated through the true nonlinear equations, without linearization. The method can capture the states up to 2nd order, and has the same computational complexity as the Extended Kalman filter. It is superior to EKF both in theory and in many practical situations [22,23].

In the Unscented particle algorithm, the Unscented Kalman filter is used to produce the proposal distribution. The algorithm is as follows, where differences from the standard particle filter are emphasised.

C. Unscented Particle for Curve Tracking

The Unscented Particle Algorithm

1. Initialisation

Draw the particles from the prior $p(X_0)$ to obtain $\{ (X_0^{(i)}, \quad \omega_0^{(i)}, \quad P_0^{(i)}), \ i = 1, \quad \cdots, \quad N \}$

2. Sampling step

a) For $i = 1, \quad \cdots, \quad N$,

Let $\hat{X}(k-1 \mid k-1) = X_{k-1}^{(i)}, \ \hat{P}(k-1 \mid k-1) = P_{k-1}^{(i)}$

- Deterministically compute $2n+1$ sigma vectors and the weights $\{ X_j(k-1 \mid k-1), W_j^{(c)}, \quad W_j^{(m)} \}$, $j = 0, \quad 1, \quad \cdots, \quad 2n$, n is the dimension of the state.

$$X_0(k-1 \mid k-1) = \hat{X}(k-1 \mid k-1)$$

$$X_j(k-1 \mid k-1) = \hat{X}(k-1 \mid k-1) + (\sqrt{(n+\lambda)\hat{P}(k-1 \mid k-1)})_j \ , \qquad j = 1, \quad 2, \quad \cdots, \quad n$$

$$X_j(k-1 \mid k-1) = \hat{X}(k-1 \mid k-1) - (\sqrt{(n+\lambda)\hat{P}(k-1 \mid k-1)})_{j-n}, \qquad j = n+1, \quad n+2, \quad \cdots, \quad 2n$$

$$W_0^{(m)} = \lambda/(n+\lambda) \quad W_0^{(c)} = W_0^{(m)} + (1-\alpha^2+\beta) \qquad W_j^{(m)} = W_j^{(c)} = \frac{1}{2(n+\lambda)}, \quad j = 1, \quad \cdots, \quad 2n$$

where $\lambda = \alpha^2(n+\kappa) - n$ is a scaling parameter. The constant α determines the spread of the sample points around $\hat{X}(k-1 \mid k-1)$. The constant κ is a secondary scaling parameter that is usually set to $3-n$, and β is used to incorporate prior knowledge of the distribution of state and is usually set to 2. $(\sqrt{(n+\lambda)\hat{P}(k-1 \mid k-1)})_j$ is the *j*th column of the matrix square root.

- Prediction

Propagate the vectors in terms of the state equation (11)

$$X_j(k \mid k-1) = AX_j(k-1 \mid k-1) + (I-A)\overline{X} \qquad j = 0, \quad \cdots, \quad 2n$$

Continues on next page

Compute the predicted state mean

$$\hat{X}(k \mid k-1) = \sum_{j=0}^{2n} W_j^{(m)} X_j(k \mid k-1)$$

Make observation to obtain $Y_j(k \mid k-1)$ ($j = 0, \quad \cdots, \quad 2n$) and compute its point density p_j , according to (13), then compute the predicted observation mean

$$\hat{Y}(k \mid k-1) = \sum_{j=0}^{2n} W_j^{(m)} Y_j(k \mid k-1)$$

Compute the predicted covariance

$$\hat{P}(k \mid k-1) = \sum_{j=0}^{2n} W_j^{(c)} [X_j(k \mid k-1) - \hat{X}(k \mid k-1)] \, [X_j(k \mid k-1) - \hat{X}(k \mid k-1)]^T + R^v$$

- Measurement Update

Compute the auto-correlation of measurement and the cross correlation of state and measurement as

$$\hat{P}_{Y_k Y_k} = \sum_{j=0}^{2n} W_j^{(c)} [Y_j(k \mid k-1) - \hat{Y}(k \mid k-1)] \, [Y_j(k \mid k-1) - \hat{Y}(k \mid k-1)]^T + R^n$$

$$\hat{P}_{X_k Y_k} = \sum_{j=0}^{2n} W_j^{(c)} [X_j(k \mid k-1) - \hat{X}(k \mid k-1)] \, [Y_j(k \mid k-1) - \hat{Y}(k \mid k-1)]^T$$

Determine the "true" measurement as $Y(k) = Y_{j^*}(k \mid k-1)$ for which $p_{j^*} = \max_{j=0,\cdots,2n} p_j$, then update the mean and covariance

$$\hat{K}(k) = \hat{P}_{X_k Y_k} \hat{P}_{Y_k Y_k}^{-1}$$

$$\hat{X}(k \mid k) = \hat{X}(k \mid k-1) + \hat{K}(k)(Y(k) - \hat{Y}(k \mid k-1)$$

$$\hat{P}(k \mid k) = \hat{P}(k \mid k-1) + \hat{K}(k) P_{Y_k Y_k} \hat{K}(k)^T$$

Draw $X_k^{(i)}$ from $\pi(X_k^{(i)} \mid X_{k-1}^{(i)}, Y_{1:k}) = N(\hat{X}_{k|k}^{(i)}, \hat{P}_{k|k}^{(i)})$, let $P_k^{(i)} = P_{k|k}^{(i)}$

b) Evaluate the new weights

$$\omega_k^{(i)} = \omega_{k-1}^{(i)} \frac{p(Y_k \mid X_k^{(i)}) p(X_k^{(i)} \mid X_{k-1}^{(i)})}{\pi(X_k^{(i)} \mid X_{k-1}^{(i)}, Y_{1:k})}, \qquad i = 1, \quad \cdots, \quad N$$

c) Normalise the weights

3. Output step

Output a set of particles $\{ (X_k^{(i)}, \quad \omega_k^{(i)}, \quad P_k^{(i)}) \ , i = 1, \quad \cdots, \quad N \}$ and the mean $E_{p(\bullet \mid Y_{1:k})}(X_k)$.

4. Selection (resampling) step
5. $k = k+1$. Go to step 2.

In stage (a) of the sampling step, for each particle in the sample set, a small number of sigma vectors with associated weights are deterministically computed. They capture the distribution of the random particle up to second order and can also deal with a heavy-tailed distribution of the particle [19]. These vectors propagate in time in the same way as the Kalman Filter, but without computation of Jacobians of the state and measurement equations. The Gaussian proposal distribution is determined by the updated mean and covariance, from which the new particle is drawn.

In the Kalman particle and Unscented Kalman particle algorithms, each independent particle evolves as a Gaussian process with its own mean and covariance. The Gaussian proposal density characterises the peak of the random particle and its second-order moment. The newly drawn particle is distributed approximately as its parent particle. It of course inherits the covariance of its parent.

In the two algorithms, each particle together with its configuration and its covariance may be regarded as a component Gaussian of the posterior density, whose distribution evolves according to the Kalman filter or Unscented Kalman filter. Therefore, the two algorithms may be interpreted as a dynamical adaptive mixture of Kalman filters or Unscented Kalman filters.

In contrast to Condensation, these methods require Kalman or Unscented updates for each sample, which introduces more computation. However, the new methods make use of new measurements and consequently, the

sampling is more efficient and can reduce the number of particles considerably. We should note that as the number of particle decreases, the capability of these algorithms to represent multimodal distributions becomes weaker, so that it is generally impossible for the algorithms to use too few particles, whatever the sampling mechanism.

VI. Discussion and Experiments

From a theoretical point of view, we can conclude that, in general, the performance of particle filter algorithms improves as the number of particles increases, which is also justified in real curve tracking experiments [9]. However, as the number of particles grows, the computational load becomes heavy. In real-time visual curve tracking applications it is important to strike an appropriate balance between performance and efficiency.

There is no analytic result concerning the problem of how many particles suffice for a given tracking task. In our experiments, we introduce the concept of *critical size* to compare the three algorithms, below which the tracker fails to follow the object throughout the image sequence. More specifically, the critical is the minimal number of particles required for successful tracking through the whole image sequence, independently of the seeds in the random number generator. Generally speaking, if the critical size is smaller, then the tracker is potentially faster.

The most time-consuming procedure in curve tracking is the measurement, which involves computing normal lines, discretising these lines, extracting grey-level values from images, and most important of all the convolutions. In our algorithms, the measurement procedure was optimized for speed. The programs run under Visual C++ 5.0 on a Pentium II 400 MHz computer. For fairness of comparison, we employ the same parameters of the measurement process: $\sigma_1 = 6$ pixel, $M = 30$ curve normals, in all three algorithms.

In the image sequence used for training,, a hand in a pointing gesture moves slowly in clutter-free background. A fine-tuned default Kalman filter is used to track movement and the results are used to train the motion model. The template of the hand is 2-D affine and is obtained from the first frame.

In the test sequence, a hand in the same pointing gesture moves randomly and swiftly in 3-D space, in front of a computer screen. The contents on the screen constitute considerable clutter.

With a number $N = 500$ of particles, all three algorithms obtain accurate tracking results. For each tracker, we decrease N gradually until we obtain an estimate of the critical size. Tracking is performed 20 times for each tracker, with different seeds in the random number generator.

The critical sizes estimated in this way are: $N = 165$ for Condensation, $N = 50$ for the Kalman particle filter, and $N = 20$ for the Unscented particle filter.

The computation times for each frame (averaged over the 20 estimates) are shown in Figure 1.

We adopt the curves $\tilde{r}_k(s)$ obtained with Condensation using $N = 500$ as the true configurations of the object, then we compute the RMSE as the mean square root of the sum of squared deviations from the true configurations. More specifically,

$$RMSE_k = \frac{1}{M}\sum_{i=1}^{M}\left\|\tilde{r}_k(s) - r_{k,i}(s)\right\| = \frac{1}{M}\sum_{i=1}^{M}\left\|\tilde{\chi}_k - \chi_{k,i}\right\|_2 \qquad i = 1, \cdots K \qquad (19)$$

where $\|\bullet\|_2$ indicates Euclidean 2-norm, $M = 20$ is the number of experiments each with a different random seed, $K = 190$ denotes the total number of frames, $r_{k,i}(s)$ is outline of the tracked object in the *k*th frame in the *i*th experiment, $\tilde{\chi}_k$ and $\chi_{k,i}$ are shape vectors corresponding to curves $\tilde{r}_k(s)$ and $r_{k,i}(s)$ respectively. The contour representation and the shape vector, together with the norm $\|\bullet\|$ will be introduced in the appendix. Figure 2 illustrates the RMSE for each frame.

Some typical tracking results are demonstrated in Figure 3 for Condensation, Figure 4 for the Kalman particle filter, and Figure 5 for the Unscented Kalman particle filter.

The experiments show that the number of particles needed can be decreased considerably by using the Kalman particle or Unscented particle filters. Computation efficiency is increased by the use of the Kalman particle filter, but a much larger computational time is required by the Unscented particle filter. This is not surprising, because most time is spent on the measurement process, so that the total time is approximately equal to the total measurement time, including the times in the sampling stage and in the updating stage.

V. Conclusions

In the unified framework of particle filtering, we presented two visual curve tracking algorithms based on the Kalman particle and Unscented particle filters. Both algorithms employ Gaussian proposal distributions, which integrate the newest measurements. What is striking in this sampling strategy is that the particles are steered towards high likelihood region. However, in the contour tracking framework, fewer particles does not necessarily mean higher computational efficiency, as seen in the experiments on the Unscented particle filter. From the experiments, we can see that Kalman particle filter is superior to the other two in visual curve tracking, and that the Unscented

particle is unsatisfactory.

ACKNOWLEDGEMENTS

We sincerely thank Dr. Arthur Pece for his great help towards improving the clarity, the readability, and the quality of the manuscript. The anonymous reviewers are acknowledged for their valuable comments.

APPENDIX CONTOUR REPRESENTATION AND SHAPE SPACE

The tracked object at time t is modelled as B-spline curve

$$r(s,t)=\begin{bmatrix}x(s,t)\\ y(s,t)\end{bmatrix}=\begin{bmatrix}B(s)^T & 0\\ 0 & B(s)^T\end{bmatrix}\begin{bmatrix}Q^x(t)\\ Q^y(t)\end{bmatrix} \quad \text{for} \quad 0\le s\le L$$

where $B(s)=\begin{bmatrix}b_0(s) & \cdots & b_{q-1}(s)\end{bmatrix}^T$, $b_i(s)\,(0\le i\le q-1)$ is the ith B-spline basis function, Q^x and Q^y are vectors of B-spline control point co-ordinates and L is the number of spans. The configuration of the spline is often restricted to a shape-space of vectors χ defined by(t is omitted)

$$Q=W\chi+Q_0 \qquad \text{or} \qquad \begin{bmatrix}Q^x\\ Q^y\end{bmatrix}=W\chi+\begin{bmatrix}Q_0^x\\ Q_0^y\end{bmatrix}$$

where W is a shape matrix whose rank is less than $2q$, Q_0 is the template curve. In visual contour tracking paradigm, the augmented vector

$$X_k=\begin{bmatrix}\chi_k\\ \chi_{k-1}\end{bmatrix}$$

follows a temporal Markov chain (10), which is fully described in section III. Typically the shape-space may allow affine deformation of the template shape Q_0.

To measure the difference between curves, L_2 norm for the curve $r(s)$ is used, and accordingly, norms $\|\bullet\|$ for control point vector Q and shape vector χ can be defined as

$$\|Q\|=\|\chi\|=\|r(s)\|$$

where

$$\|r(s)\|=\left(\frac{1}{L}\int^L r(s)^T r(s)ds\right)^{\frac{1}{2}},$$

$$\|Q\|=\left(Q^TUQ\right)^{\frac{1}{2}}, \quad U=\frac{1}{L}\int_0^L\begin{bmatrix}B(s)B(s)^T & 0\\ 0 & B(s)B(s)^T\end{bmatrix}ds$$

$$\|\chi\|=(\chi^TW^TUW\chi)^{\frac{1}{2}}$$

Details about the definitions of the norms can be found in [1]. In many situations, the shape space is chosen as planar affine space where shape space can be described as

$$W=\begin{bmatrix}1 & 0 & Q^x & 0 & 0 & Q_0^y\\ 0 & 1 & 0 & Q_0^y & Q_0^x & 0\end{bmatrix}$$

The first two columns of W represent horizontal and vertical translation, the last four columns represent rotation, scaling and shearing. In practice, we choose Q_0 to have its centroid at the origin to make the last four columns free of translation.; and we also orthonormalize W such that $<W_i,\ W_j>=W_i^TUW_j=\begin{cases}0 & i\ne j\\ 1 & i=j\end{cases}$, where W_i is the ith column vector of W, $<W_i,\ W_j>$ indicates the inner product of the two vectors. Then we can easily obtain (19) because the shape matrix satisfies $W^TUW=I$, where I is identity matrix.

REFERENCES

[1] A. Blake, M. Isard, *Active Contours*. Springer-Verlag, 1998.
[2] D. Terzopoulos, R. Szeliski, "Tracking with Kalman Snakes," in *Active Vision*, A. Blake , A. Yuille eds., 1992, pp. 3-20.
[3] D. Metaxas, D. Terzopouilos, "Shape and Nonrigid Motion Estimation through Physics-Based Synthesis," *IEEE Trans. PAMI*, vol. 15, no. 6, pp. 580-591, 1993.
[4] N. Peterfreund, "Robust Tracking of Position and Velocity with Kalman Snakes," *IEEE Trans. PAMI*, vol. 21, no. 6, pp. 564-569, 1999.
[5] A. Blake, R. Curwen, A. Zisserman, "A framework for Spatio-temporal Control in the Tracking of Visual Contours," *Int. J. Computer Vision*, vol. 11, no. 2, pp. 127-145, 1993.
[6] A. Blake, M. Isard, D. Reynard, "Learning to track the Visual Motion of Contours," *J. Artificial Intelligence*, vol. 78, pp. 101-134, 1995.
[7] M. Isard, A. Blake, "Contour Tracking by Stochastic Propagation of Conditional density," in *Proc. 4th European Conf. Computer Vision*, 1996, pp. 343-356.
[8] V. Grenander, Y. Chow, D. Keenan, *HANPS: A Pattern Theoretical Study of Biological Shapes*. New York: Springer-Verlag, 1992.
[9] M. Isard, A. Blake, "Condensation-Conditional Density Propagation for Visual Tracking," *Int. J. Computer Vision*, vol. 29, no. 1, pp. 5-28, 1998.
[10] A. Doucet, J. F. G. de Freitas, N. Gorden, *Sequential Monte Carlo Methods in Practice*. New York: Springer-Verlag, 2001.
[11] M. Isard, A. Blake, "ICondensation: Unifying Low-level and High–level Tracking in a Stochastic Framework," in *Proc. 5th European Conf. Computer Vision*, 1998, pp. 893-908.
[12] J. MacCormick, A. Blake, "Partitioned Sampling, Articulated Objects and Interface-quality Hand Tracking," in *Proc. 7th European Conf. Computer Vision*, 2000.
[13] J. Sullivan, A. Blake, M. Isard, J. MacCormick, "Object Localization by Bayesian Correlation," in *Proc. 7th Int. Conf. On Computer Vision*, vol. 2, 1999, pp. 1068-1075.
[14] A. Doucet, S. J. Godsill, C. Andrieu, "On Sequential Simulation-based methods for Bayesian filtering," *Statistics and Computing*, vol. 10, no. 3, pp.197-208, 2000.
[15] M. K. Pitt, N. Shephard, "Filtering Via Simulation: Auxiliary Particle Filters," *Journal of the American Statistical Association*, vol. 94, no. 446, pp. 590-599, 1999.
[16] J. F. G. de Freitas, M. Niranjan, A. H. Gee, A. Doucet, "Sequential Monte Carlo Methods to Train Neural Networks Models," *Neural Computation*, vol. 12, no. 4, pp. 955-993, 2000.
[17] N. J. Gordon, D. J. Samond, A. F. M. Smich, "Novel Approach to Nonlinear/NonGaussian Bayesian State Estimation," *IEE Proc.-F*, vol. 140, no. 2, pp. 107-113, 1993.
[18] J. S. Liu, R. Chen. "Sequential Monte Carlo Methods for Dynamic System," *Journal of the American Statistical Association*, vol. 93, pp. 1032-1044, 1998.

[19] R. Merwe, A. Doucet, N. Freitas, E. Wan, "The Unscented Particle Filter," Technical Report CUED/F-INFENG/TR380, Cambridge University, Engineering Department, August 2000. Available: http://cslu.cse.ogi.edu/publications/ps/merwe00.pdf

[20] A. Doucet, N. J. Gordon, V. Krishnamurthy, " Particle filters for state estimation of jump Markov linear systems," *IEEE Trans. on Signal Processing*, vol. 49, no. 3, pp. 613-624, 2001.

[21] J. MacCormick, A. Blake, "Spatial Dependence in the Observation of Visual Contours," in *Proc. 5th European Conf. Computer Vision*, 1998, pp. 765-781.

[22] S. J. Julier, J. K. Uhlmann, "Anew Extension of the Kalman Filter to Linear System," in *Proc. of Aerosense: The 11th Int. Symp. On Aerospace/Defense Sensing, Simulation and Control*, Orlando, Florida, 1997, pp. 182-193.

[23] R. Merwe, E. A. Wan, "The Unscented Kalman Filter for Nonlinear Estimation," in *Proceedings of Symposium on Adaptive Systems for Signal Processing, Communication and Control (AS-SPCC), IEEE*, Lake Louise, Alberta, Canada, 2000, pp. 153-158.

[24] G. Kitagawa, "Monte Carlo Filter and Smoother for Non-Guassian Nonlinear State Space Modes," *J. Comput. Graph. Statistics*, vol. 5, pp. 1-25, 1996.

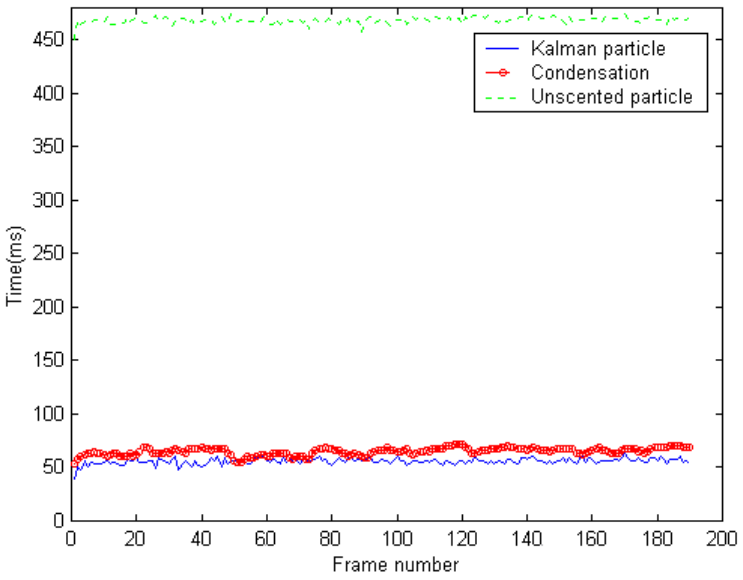

Fig. 1. Tracking time of the three algorithms.

Mean computation time for one frame is approximately 55ms for Kalman particle, 67ms for Condensation, and 460ms for Unscented Kalman particle.

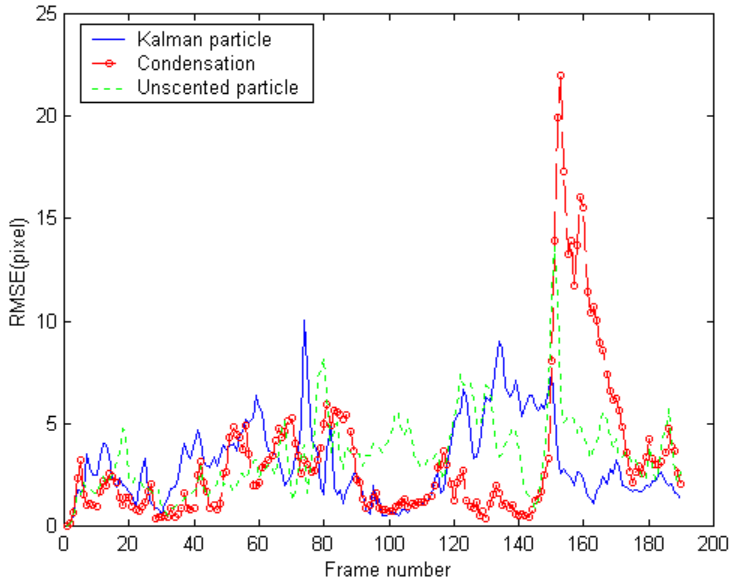

Fig. 2. RMSE of the three algorithms.

Mean RMSE for one frame is approximately 3.66 for Kalman particle, 4.93 for Condensation, and 3.98 for Unscented Kalman particle.

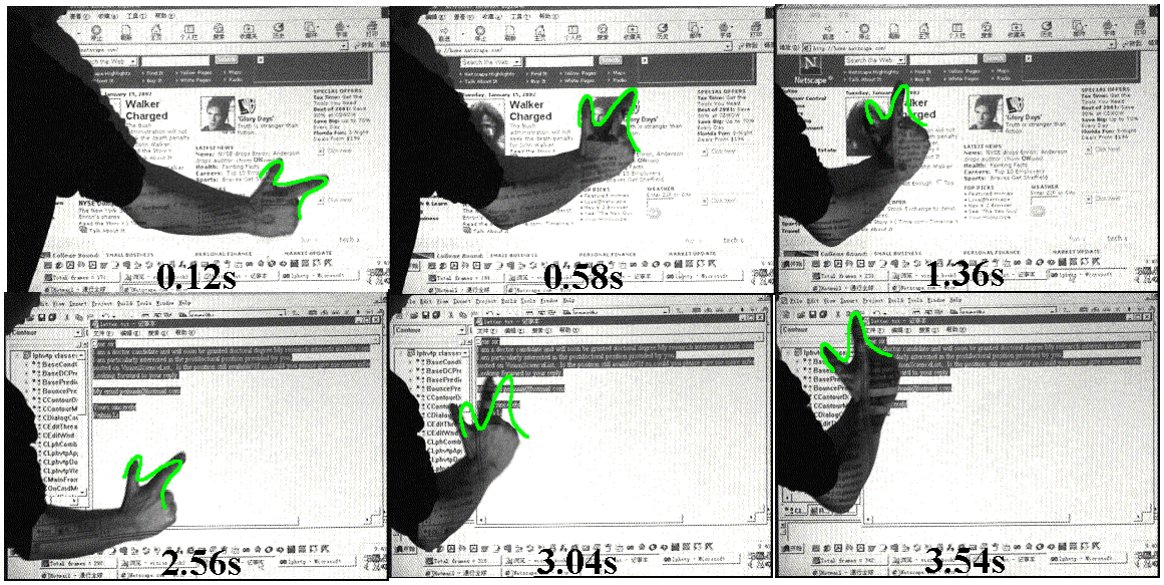

Fig. 3. Tracking results with Condensation

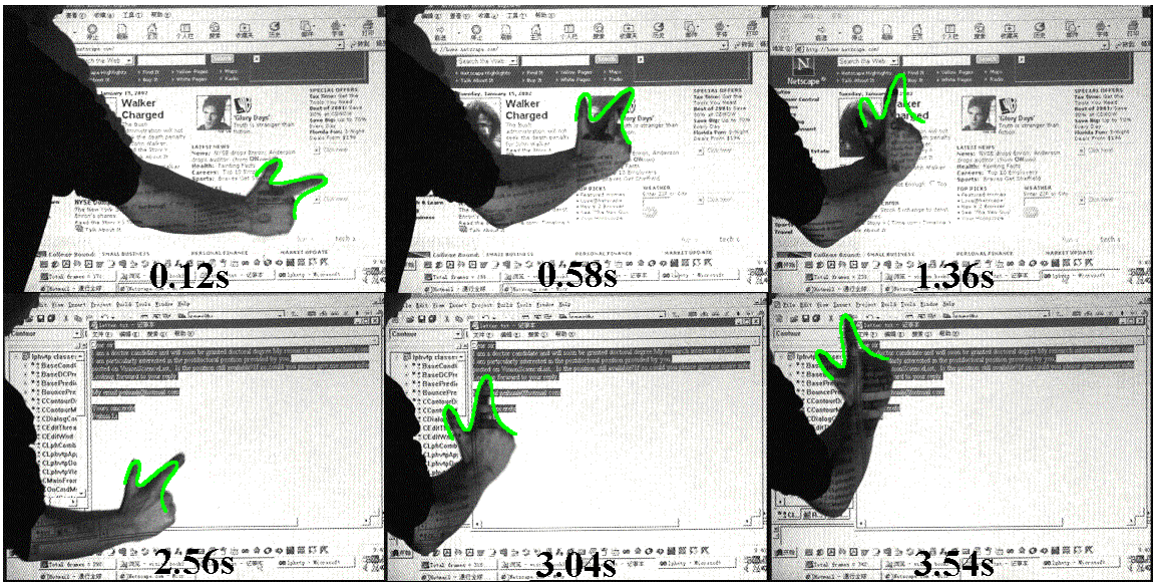

Fig. 4. Tracking results with Kalman particle filter

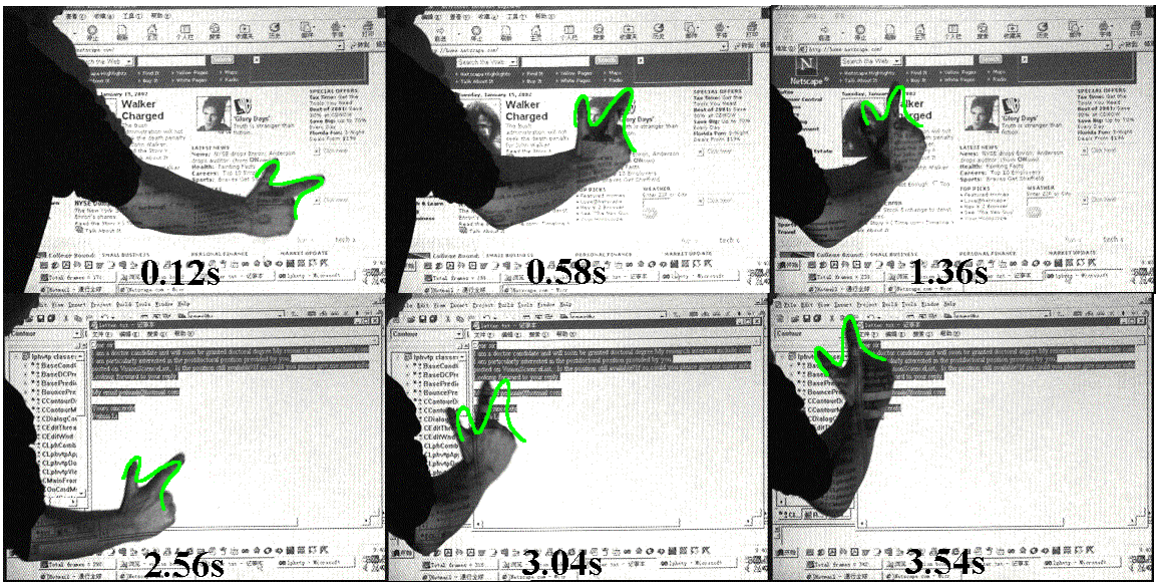

Fig. 5. Tracking results with Unscented Kalman particle filter