Vol.3, No.4, 301-306 (2011)
http://dx.doi.org/10.4236/ns.2011.34039

Natural Science

A new improved filter for target tracking: compressed iterative particle filter

Hongbo Zhu[1]*, Hai Zhao[2], Dan Liu[2], Chunhe Song[2]

[1]Software College Northeastern University, Shenyang, China; *Corresponding Author: polominister@sina.com
[2]Institute of Information and Technology Northeastern University, Shenyang, China; {zhhai,liud,songchh}@mail.neuera.com

Received 27 October 2010; revised 25 November 2010; accepted 20 December 2010.

ABSTRACT

Target tracking in video is a hot topic in computer vision field, which has wide applications in surveillance, robot navigation and human-machine interaction etc. Meanshift is widely used algorithm in video target tracking field. The basic mean shift algorithm only considers the color of targets as the tracking characteristic feature, so if the appearance of the target changes greatly or there exits other objects whose color is similar to the target, the tracking process will fail. To enhance the stability and robustness of the algorithm, we introduce particle filter into the tracking process. Basic particle filter has some disadvantages such as low accuracy, high computational complexity. In this paper, an improved particle filter GA-UPF was proposed, in which a new re-sampling algorithm was used to predict target centroid position. The target tracking system of binocular stereo vision is designed and implemented. Experimental results have shown that our algorithm can tracking object in video with high accuracy and low computational complexity.

Keywords: Filtering Method; Particle Filtering; Unscented Particle Filter; Genetic Algorithm; Target Tracking

1. INTRODUCTION

Target tracking in video is a hot topic in computer vision field, which has wide applications in surveillance, robot navigation and human-machine interaction etc. Many algorithms are proposed, but if the appearance of the target changes greatly or there exits other objects who are affected by the occasion. To enhance the stability and robustness of the algorithm, particle filter is introduced into the tracking process. Basic particle filter (PF) has some disadvantages such as low accuracy, high computational complexity. Due to solve the problem, an improved particle filter GA-UPF was proposed.

GA-UPF is the integration of genetic algorithm (GA) and unscented particle filter (UPF). It successfully offsets the defect in PF. And GA makes sure that particle degeneracy phenomenon and particle sample impoverishment in re-sample process cannot be discovered. Experimental results have shown that our algorithm has better filtering effect.

The remaining of the paper is organized as follows: in the Section 2, a brief description of GPF is presented. In the Section 3, firstly, the GA algorithm is introduced, and a new type GA – one-step predefined GA process is proposed. Then the details of the new PF this paper proposed – GA-UPF is presented. In Section 4 the proposed algorithm is compared to other several different filtering algorithms, and finally, some concluding remarks is given in Section 5.

2. BASIC PARTICLE FILTER

The system probability model set as $p\left(x_t \middle| x_{t-1}\right)$ and $p\left(y_t \middle| x_t\right)$, and the system follows the Markov process. The problem being addressed here is an estimating problem of the state of a system as a set of observations becomes available on-line. The initial value of prior probability density function is $p\left(x_0\right)$. The complete solution of sequential estimation problem is the posteriori probability density function $p\left(x_{0:t} \middle| y_{1:t}\right)$.

Using Monte Carlo sampling points, particle filter executes the filtering process by generating weighted sampling points of state variances recursively. A group of random testing value is $\left\{x_{0:t}^{(i)}, w_t^{(i)}\right\}$, $\left\{x_{0:k}^{(i)},\ i=1,\cdots,N\right\}$ is sample set. As the corresponding weights, $\left\{w_t^{(i)},\ i=1,\cdots,N\right\}$ content with $\sum_i w_t^i = 1$. The posteriori probability density function is similar as:

$$p\left(x_{0:t} \middle| y_{1:t}\right) \approx \sum_{i=1}^{N} w_k^{(i)} \delta\left(x_{0:t} - x_{0t}^{(i)}\right) \quad (1)$$

Through introducing and resampling the key density $q\left(x_{0:t}\middle|y_{1:t}\right)$, the result is as follow:

$$x_{0t}^{(i)} \sim q\left(x_{0:t}\middle|y_{1:t}\right) \tag{2}$$

As Bayes theory and re-sampling principle, the condition of [1]:

$$w_t^{(i)} \propto \frac{p\left(x_{0t}^{(i)}\middle|y_{1:t}\right)}{q\left(x_{0t}^{(i)}\middle|y_{1:t}\right)} \tag{3}$$

Setting decomposition of the key density:

$$q\left(x_{0:t}\middle|y_{1:t}\right) = q\left(x_t\middle|x_{0:t-1},y_{1:t}\right)q\left(x_{0:t-1}\middle|y_{1:t-1}\right) \tag{4}$$

Generic particle filter algorithm can be predicted as follows [2]:

Initialization

For each particle

Draw the states $x_0^{(i)}$ from the prior $p\left(x_0\right)$;

End

For each loop

Importance sampling step

For each particle

Sample $\hat{x}_0^{(i)} \sim q\left(x_t\middle|x_{0:t-1}^{(i)},y_{1:t}\right)$

End

For each particle

Evaluate the importance weights up to a normalizing constant:

$$w_t^{(i)} = w_{t-1}^{(i)}\frac{p(y_t\middle|x_t^{(i)})p\left(\hat{x}_t^{(i)}\middle|x_{t-1}^{(i)}\right)}{q\left(\hat{x}_t^{(i)}\middle|\hat{x}_{0:t-1}^{(i)},y_{1:t}\right)} \tag{5}$$

For each particle, normalize the importance weights:

$$\tilde{w}_t^{(i)} = w_{t-1}^{(i)}\left[\sum_{j=1}^{N} w_t^{(j)}\right]^{-1} \tag{6}$$

//Re-sample

Eliminate the samples with low importance weights and multiply the samples with high importance weights, to obtain N random samples $x_{0:k}^{(i)}$ approximately distributed according to $p\left(x_{0:k}\middle|z_{1:k}\right)$.

Assign each particle an equal weight: $w_t^i = 1/N$. When executing particle filtering, the choice of the proposal distribution is very important. Usually, the transition prior distribution is chosen to be the proposal distribution:

$$q\left(x_t\middle|X_{t-1},Y_t\right) = p\left(x_t\middle|x_{k-1}\right) \tag{7}$$

But as not considering the recent observation, when using the transition prior distribution as the proposal distribution, the filtering result is usually poor, especially when the noise is heavy. In this paper, a kind of semi-iterative unscented transformation was proposed to address this issue.

3. THE GENETIC ALGORITHM

3.1. Genetic Algorithm

GA is an optimization strategy generally employed to find a global best point. In a genetic algorithm, a population of strings (called chromosomes or the genotype of the genome), which encode candidate solutions (called individuals, creatures, or phenotypes) to an optimization problem, evolves toward better solutions. There are three main steps in the whole GA process, which consists of Selection, Crossover and Mutation as **Figure 1**.

GA is a search heuristic that mimics the process of natural evolution. This heuristic is routinely used to generate useful solutions to optimization and search problems as **Figure 1**. Using pseudo code to descript the simple generational genetic algorithm is as follow:

1) The initial population of individual must be chosen.

2) Evaluate the fitness of each individual in that popu-

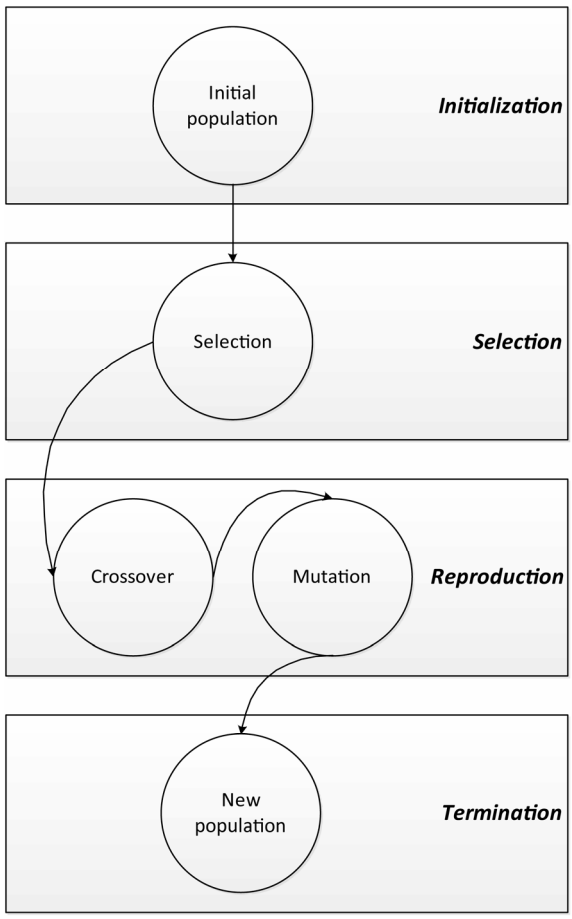

Figure 1. Process of genetic algorithm (GA).

lation.

3) Make selection of the best-fit individuals for reproduction.

4) Breed new individuals through crossover and mutation operations to give birth to offspring.

5) Evaluate the individual fitness of new individuals.

6) Replace least-fit population with new individuals.

3.2. GA in the PF Process

The particles degeneration phenomenon is a widespread problem of particle filter algorithm. In order to eliminate that phenomenon, the re-sample algorithm is introduced as depicted in the Section 2. In the re-sampling process of regular particle filtering, the particles with bigger weights will have more offspring, while the particles with smaller weights will have few or even on offspring. It inspires us to reduce the number of particles with small weights and find more particles with big weights, which will make the proposal distribution more closed to the poster distribution. But re-sampling algorithm causes another problem which loses particle diversity and reduces the performance of the particle filter algorithm, and even makes the distribution function of particle filter algorithm spread. Therefore GA is proposed in PF process.

The choice of fitness function is the most important issue of using genetic algorithm. In the proposed algorithm, each particle in particle algorithm is looked upon a pair of chromosome. We want to find more particles with bigger weights, so the fitness can be chosen as the value of weights directly. As usually the aim of GA algorithms is to minimize the fitness function, so the fitness is the minus value of particle weight in the GA-PF as follows:

$$fitness_t^{(i)} = \frac{p\left(y_t \middle| x_t^{(i)}\right) p\left(\hat{x}_t^{(i)} \middle| x_{t-1}^{(i)}\right)}{q\left(\hat{x}_t^{(i)} \middle| \hat{x}_{0:t-1}^{(i)}, y_{1:t}\right)} \tag{8}$$

Secondly, it's already so large to compute consumption of particle filtering that the GA process computing consumption introduced should be reduced. In the GA-PF algorithm, for every particle, at first, a random number in the range of 0 and 1 will be generated, and only when the number is smaller than a predefined threshold, the GA process can be conducted. A new type of GA process – one step predefined GA is introduced. In this process, only when the new location is better than the original one, the particle will move to the new one, and the location updating process will be conducted only one time in each particle filtering generation. The new particle is then used in the next iteration of the algorithm. Commonly, the algorithm terminates when either a maximum number of generations has been produced, or a satisfactory fitness level has been reached for the particle.

Finally, the location of best particle determines the direction of particle updating in current generation and itself location. And as in the one-step predefined GA process, particle need not remember its individual best location of history; the updating mode uses the social only mode of GA algorithm.

3.3. The GA-UPF Algorithm

In this section, a brief description of UPF is presented as follow:

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1) \tag{9}$$

$$w_t = w_{\max} - \frac{(w_{\max} - w_{\min})t}{g_{\max}} \tag{10}$$

The mentioned one-step predefined GA process is used in the UPF algorithm. The information of UPF algorithm can be found in [3,4]. Where $j \in \{1, 2, \cdots, Dn\}$, $i \in \{1, 2, \cdots, n\}$, n is the size of the population and Dn is the Dimension of the space searched; w is the inertia weight, generally updated as **Eq.3** [4], and $g_{\max}$ is the maximal evolution generations. c_1 and c_2 are two positive constants. And the GA-UPF algorithm can be decried as **Figure 2**.

The detailed process of GA-UPF is as follow:

1) Initial chromosome sample: Draw the new particle group $\left\{x_t^{(i)}, i = 1, 2, \cdots, N\right\}$ from the prior $p\left(x_t \middle| x_{t-1}\right)$;

2) Assign weights: Every kind of GA has to find an effective coding scheme. Real matrix encoding is applied to estimate the parameters for sequential UPF algorithm.

For each loop:

3) Selection: The fitness of each chromosome set as measuring probability in [5], and using Roulette Wheel Selection which the most commonly methods to choose the higher fitness chromosomes, the terminal chromosomes is the model of offspring;

4) Crossover: At this stage, arithmetic crossover is widely used for the chromosome encoded by Real matrix. In arithmetic crossover, a pair of new individual is generated by the linear combination of a pair of parent individual:

$$\begin{aligned} \left(x_t^i\right)' &= \alpha x_t^i + (1-\alpha) x_t^j \\ \left(x_t^j\right)' &= \alpha x_t^j + (1-\alpha) x_t^i \end{aligned} \tag{11}$$

Here $\alpha \in \forall[0,1]$, x_t^i, x_t^j is the pair of parent individual $\left(x_t^i\right)'$, $\left(x_t^j\right)'$ is the pair of new individual

5) Mutation: β is a random real;

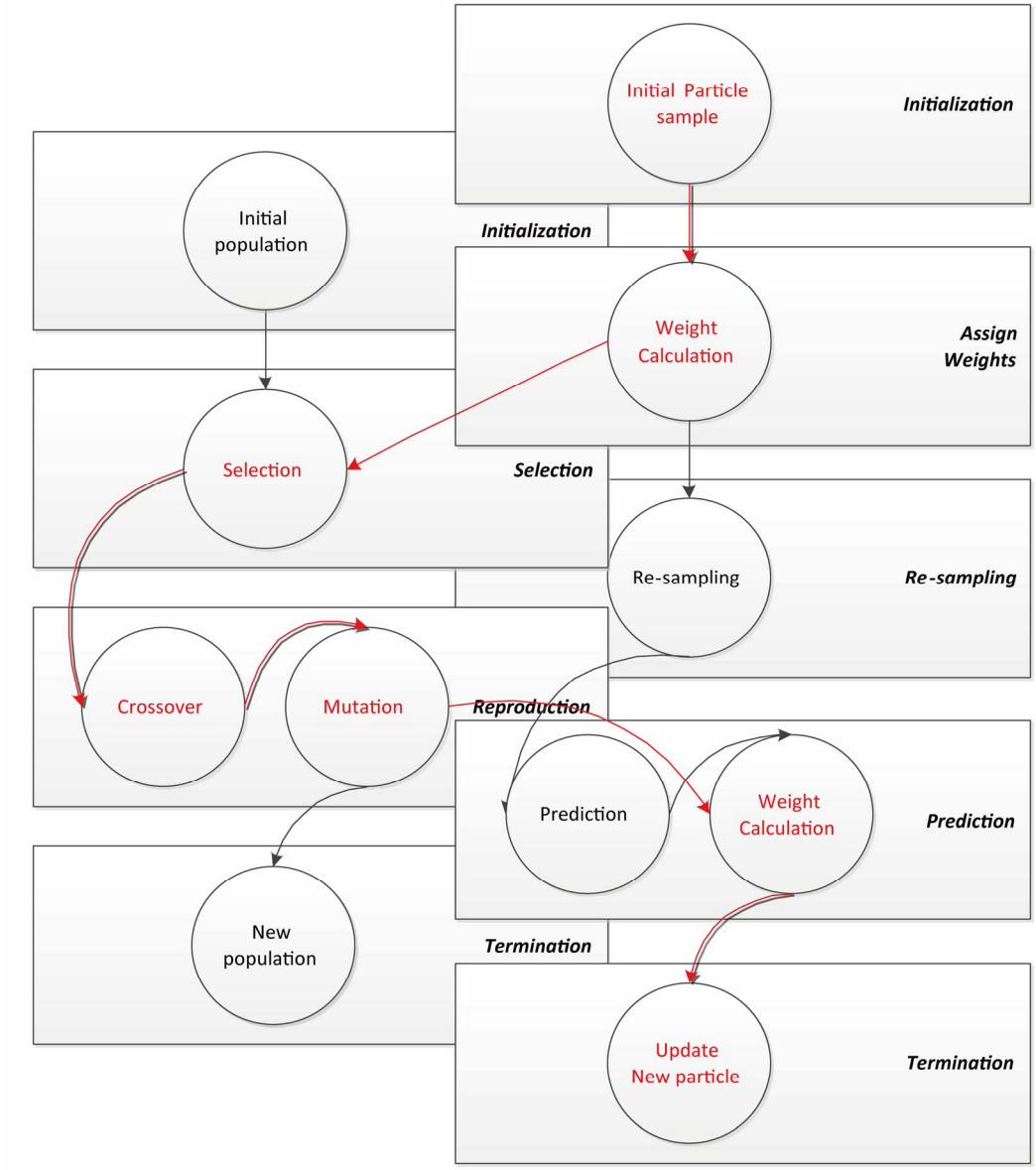

Figure 2. Process of GA-UPF.

If $\alpha < 0.6$

$$\left(x_t^j\right)' = (1+\beta)x_t^j$$

Else

$$\left(x_t^j\right)' = (1-\beta)x_t^j$$

End loop;

Update new chromosome: The final particle is obtained.

4. THE SIMULATION EXPERIMENTS

In this paper, the proposed algorithm was compared to the other seven filtering algorithms – EKF, UKF, GPF, GPFMCMC, EKFPF, EKFPFMCMC, UPF, Experimental settings are shown in the **Table 1**. The settings of other six filtering algorithms are the same as [2,5-8].

4.1. Benchmark Function

In this paper, the following benchmark function [4] was chosen to test the proposal algorithm.

Benchmark 1:

$$x_t = 1 + \cos\left(w\pi(t-1)\right) + 0.5x_{t-1} + w_t$$

$$y_t = \begin{cases} 0.3 * x_t^2 + v_t, & t \le 30 \\ 0.5 * x_t - 2 + v_t, & t > 30 \end{cases}$$

Benchmark 2:

$$x_t = 1 + \cos\left(0.04 * \pi * (t-1)\right) + 0.5 * x_{t-1} + w_{t-1}$$

$$y_k = 0.3 * x_k^2 + \frac{\sin(x_t)}{10} + v_k$$

where $w_t \sim \gamma(3,1)$, $v_k \sim N(0, 1e-4)$, particle number N = 200, sample time T = 100, which the results were the average of 50 times of experiments. $C = 0.5$, R = 1e−4, $Q = 0.75$; $\alpha = 1$, $\beta = 0$, $k = 2$.

4.2. Experimental Results and Remarks

Experimental results are shown in **Figures 3**, **4** and **Tables 1**, **2**. All results are the means of 100 runs, as the results of UPF and GA-UPF are close and far different with the other algorithms.

As shown in the experimental results, it is clear that, EKF has the worst results. While the proposed algorithm has the best results, but has longer running time than EKF. In theory, the running time of GA-UPF is between one and two times of UPF, due to the refining step, and it has been proved by the simulation results.

We extracted a frame from the original video sequences per 5 to 10 frames. This is equivalent to increase the speed and uncertainty of target motion, and

Table 1. Results on benchmark 1.

	MSE	
	mean	variance
EKF	0.6975	0.1248
UKF	0.3234	0.1297
PF	0.18001	0.1650
PF-EKF	0.1426	0.2147
PF-UKF	0.1239	0.1107
GA-UPF	0.0268	0.0359

Table 2. Results on benchmark 2.

	MSE	
	mean	Variance
EKF	0.3375	0.1438
UKF	0.2347	0.1277
PF	0.2301	0.6247
PF-EKF	0.3181	0.1147
PF-UKF	0.2339	0.1347
MPF	0.0968	0.0959

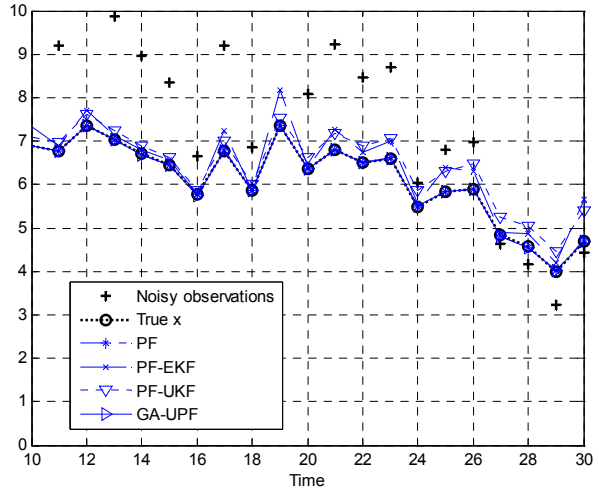

Figure 3. Results on benchmark 1.

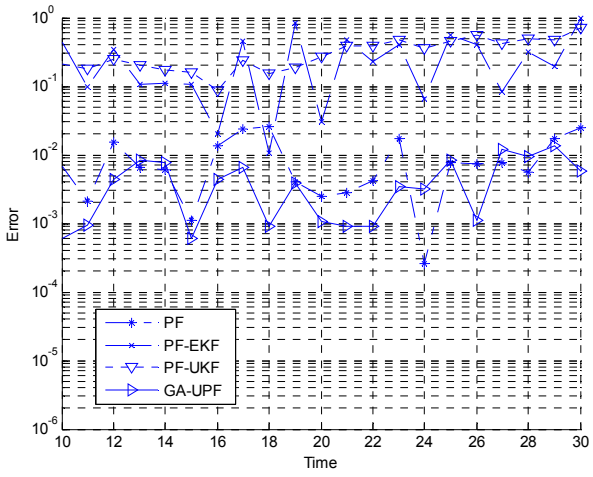

Figure 4. Results on benchmark 2.

then the difficulty level of tracking algorithm is increased. The Experimental result of target tracking is shown as **Figure 5**.

In both of the video cameras (Left and Right), the observed images are compared. We can see that tracking accuracy is in inverse proportion to the distance between targets and cameras. Tracking errors always occur when the target is far away from camera, but the performance shows amazing stability, and the error is within acceptable limits. The improved algorithm and system can achieve the desired result.

5. CONCLUSIONS

Basing on the concept of re-sampling, particles with bigger weights should be re-sampled more time, this paper has proposed a new type of particle filtering algorithm – GA-UPF. In the GA-UPF, after calculating the

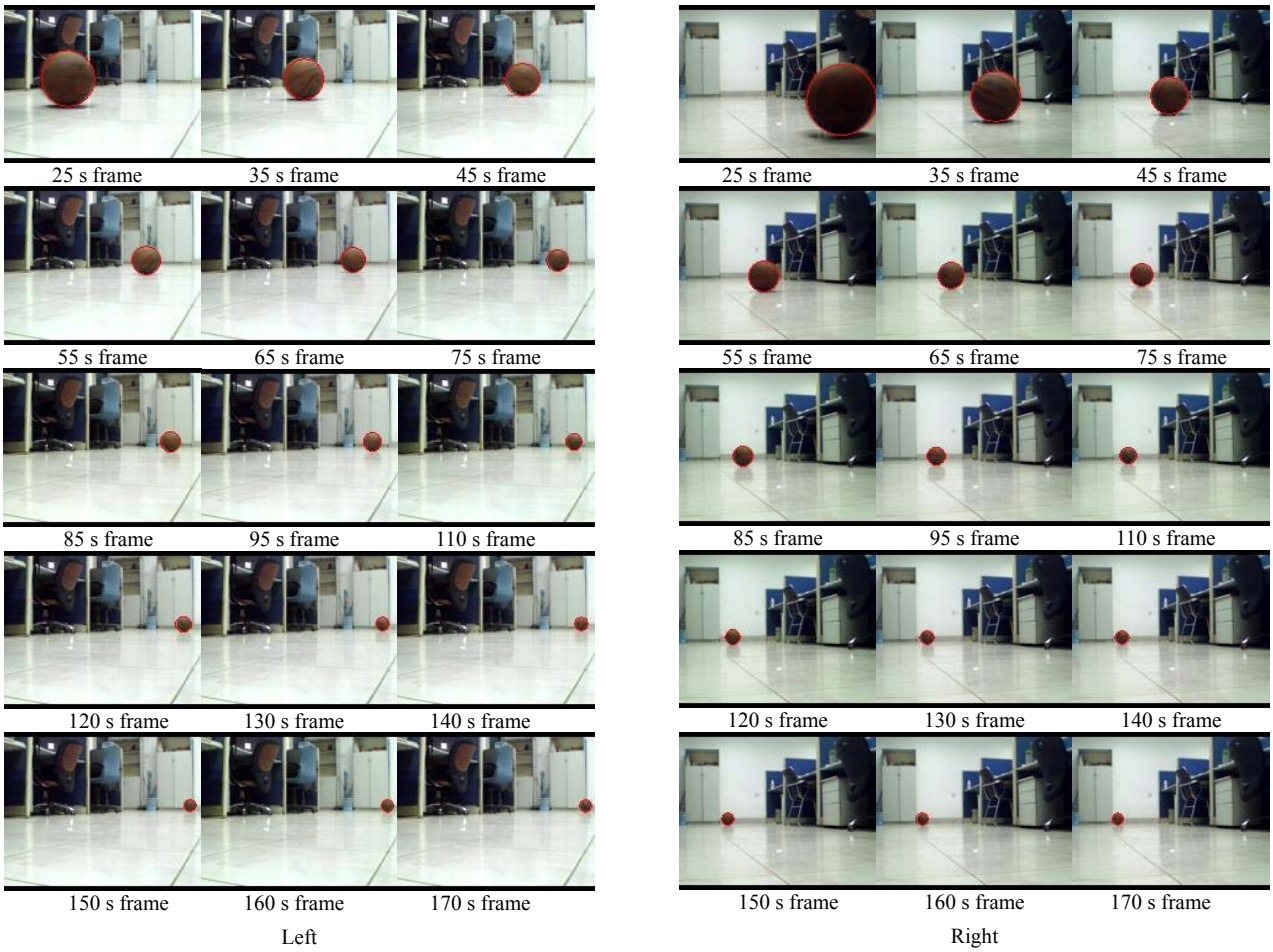

Figure 5. Sample per 5 frames of tracking video image of binocular vision tracking system.

weight of particles, some particles will join in the refining process, which means that these particles will move to the region with higher weights. This process can be regarded as one-step predefined GA process, so the proposed algorithm is named GA-UPF. Although the GA process increases the computing load of GA-UPF, but the refined weights may make the proposed distribution more closed to the poster distribution. In the following experiment, the proposed algorithm has better performances than other several types of filtering methods.

REFERENCES

[1] Hou, Z.Q. and Han, C.Z. (2005) Based on the background pixel classification algorithm of reconstructing. *Journal of Software*, **16**, 1568-1576.

[2] Doucet, A. and Gordon, N. (2001) Sequential Monte Carlo methods in practice. Springer-Verlag, New York.

[3] Mo, Y.W. and Xiao, D.Y. (2003) Hybrid system monitoring and diagnosing based on particle filter algorithm. *Acta Automation Sinica*, **29**, 641-648.

[4] Freitas, J.F.G., Niranjan, M., Gee, A.H., *et al.* (2000) Sequential Monte Carlo methods to train neural network models. *Neural Computation*, **12**, 995-993. doi:10.1162/089976600300015664

[5] Berzuini, C. and Best, N. (1997) Dynamic conditional independence models and Markov chain Monte Carlomethods. *Journal of the American Statistical Association*, **92**, 1403-1412. doi:10.2307/2965410

[6] Belviken, E. and Acklam, P.J. (2001) Monte Carlo filters for non-linear state estimation. *Automatica*, **37**, 177-183. doi:10.1016/S0005-1098(00)00151-5

[7] Lei, L., Li, Y.-J. and Zhang, K. (2007) A fast algorithm for searching object centriods in binary images. *Infrared Technology*, **29**, 548-551.

[8] Higuchi, T. (1997) Monte Carlo filtering using genetics algorithm operator. *Journal of Statistical Computation and Simulation*, **59**, 1-23. doi:10.1080/00949659708811843