International Journal of Electrical, Electronic and Communication Sciences
ISSN: 2517-9438
Vol:1, No:9, 2007

A New Particle Filter Inspired by Biological Evolution: Genetic Filter

S. Park, J. Hwang, K. Rou, and E. Kim

Abstract**—**In this paper, we consider a new particle filter inspired by biological evolution. In the standard particle filter, a resampling scheme is used to decrease the degeneracy phenomenon and improve estimation performance. Unfortunately, however, it could cause the undesired the particle deprivation problem, as well. In order to overcome this problem of the particle filter, we propose a novel filtering method called the genetic filter. In the proposed filter, we embed the genetic algorithm into the particle filter and overcome the problems of the standard particle filter. The validity of the proposed method is demonstrated by computer simulation**.**

Keywords**—**Particle filter, genetic algorithm, evolutionary algorithm.

I. Introduction

TO estimate the latent variables of the dynamics, several filtering methods have been reported, for example, Kalman filter (KF) [1] and grid based filter [2]. In these filters, the posterior density probability was assumed to be Gaussian. Unfortunaly, however in many real problems the posterior density is not Gaussian but is multimodal and its performance is not as good as expected.

To overcome this problem, many researches have been reported on the nonlinear filtering methods such as extended Kalman filter (EKF) and the approximation grid based filter.[2,3] One of the famous methods is the particle filter [2-4]. In the particle filter, any assumption on the functional form of the posterior is not made. Instead, the posterior probability density is approximated as a set of particles. When the particles are properly placed, weighted and propagated, posteriors can be estimated sequentially over time. The density of particles represents the probability of posterior function. By using a finite number of particles, we can estimate almost any kind of system dynamics; even nonlinear system with non-Gaussian, or multimodal distributions.

In spite of these advantages, the particle filter has a serious drawback: Even with a large number of particles, it may happen that there are no particles in the vicinity of the correct state. This drawback is called the particle deprivation problem. To overcome these drawbacks of particle filter, we embed the genetic algorithm into the standard particle filter. This philosophy is not completely new. In the past relative research, K. Uoraki *at el.* are combined the evolutionary algorithm, especially evolutionary programing with the standard particle filter[5]. But K. Uoraki ‘s work was not fully exploited advantage of evolutionary algorithm.. In this paper, by using genetic algorithm which is one of the evolutionary algorithm, we can solve several drawbacks of particle filter which would be occurred in some special tracking case, like maneuvering target of state jump tracking. And as we reform the genetic operations, such as crossover and mutation, we can use genetic algorithm more properly than previous work.

The organization of this paper is as follows; In Section 2, the particle filter is briefly reviewed. In Section 3, the genetic filter is proposed and the detailed explanations are given. In Section 4, simulation is conducted and the result is presented to show the effectiveness of the genetic filter. Finally, conclusion remarks are made in Section 5..

II. Particle Filter

The particle filter is a special version of the Bayes filter based on Monte Carlo sampling. In the particle filter, we represent the posterior probability $p(x_k \mid z_{1:k})$ as a set S_k of N weighted samples

$$S_k = \{(x_k^m, w_k^m) \mid m = 1, \cdots, N\}. \quad (1)$$

Here, x_k^m denotes the m-th particle of S_k and the w_k^m is the associated importance weights. The standard particle filter (SIR) is summarized as in Table I [2].

Manuscript received November 30, 2006. This work was supported by grant No. R01-2006-000-11016-0 from the Basic Research Program of the Korea Science & Engineering Foundation.

S. Park is with the Department of Electrical and Electronic Engineering, Yonsei University , 134 Shinchon-dong, Sudaemoon-ku, Seoul 120-749, Korea (phone: 82-2-2123-2863; e-mail:keiny@yonsei.ac.kr).

J. Hwang is with the Department of Electrical and Electronic Engineering, Yonsei University , 134 Shinchon-dong, Sudaemoon-ku, Seoul 120-749, Korea (phone: 82-2-2123-2863; e-mail:purnnara@yonsei.ac.kr).

S. Park is with the Department of Electrical and Electronic Engineering, Yonsei University , 134 Shinchon-dong, Sudaemoon-ku, Seoul 120-749, Korea (phone: 82-2-2123-2863; e-mail: ycs025@yonsei.ac.kr).

E. Kim is with the Department of Electrical and Electronic Engineering, Yonsei University , 134 Shinchon-dong, Sudaemoon-ku, Seoul 120-749, Korea (phone: 82-2-2123-2863; e-mail: etkim@yonsei.ac.kr).

TABLE I
THE ALGORITHM OF SIR

1: Algorithm SIR $[\{x_k^m, w_k^m\}|_{m=1}^M] = SIR[\{x_{k-1}^m, w_{k-1}^m\}|_{m=1}^M, z_k]$
2: Initialize particles (weight)
3: For $m = 1:N$
4: Draw $x_k^m \sim p(x_k | x_{k-1}^m)$
5: Assign a weight w_k^m to the particle according to $p(z_k | x_k^m)$
6: End for
7: For $m = 1:N$
6: Resampling x_k^m according to w_k^m
7: End for
8: Replace $k \leftarrow k+1$ and return Line 3

The particle filter algorithm consists of three steps: sampling, calculation of the importance weight and resampling. In the sampling step, samples are generated according to $p(x_k | x_{k-1}^m)$. In the step of importance weighting, the importance weight is computed for each particle as $p(z_k | x_k^m)$. In the resampling step, the particles with different weights are sampled again with replacement according to their weights and the particles with different weights are replaced by the new particles with equal weights $(1/N)$. The particles with larger weights are more likely to be selected than the particles with smaller weights.

III. THE PARTICLE FILTER WITH VARYING NUMBER OF PARTICLES (PFVANP)

The Bayes filter is the most generic estimation method but it is not appropriate for direct implementation because it requires the convolution of the distribution functions. Instead, the Kalman filter is the concrete and simplest implementation of the Bayes filter and is widely employed in many engineering fields. But its use is limited to the case of the linear estimation with the Gaussian noise.

These days, the particle filter has received the high attention in the signal processing community as an alternative method for the state estimation. It deals with not only linear models with Gaussian noise but also the non-linear models with non-Gaussian noise. But, as mentioned in Section 1, the particle filter suffers from *the particle deprivation problem*: Even with a large number of particles, it may happen that there are no particles in the vicinity of the correct state. In order to overcome the drawbacks of the particle filter, we propose a new filtering method called the genetic filter. The basic idea of the genetic filter is to view the particles of the particle filter as the chromosomes of the genetic algorithm and embed the genetic algorithm [6] into the particle filter. Shown in Fig. 1 is the procedure of the Genetic filter, which is very similar to the standard genetic algorithm.

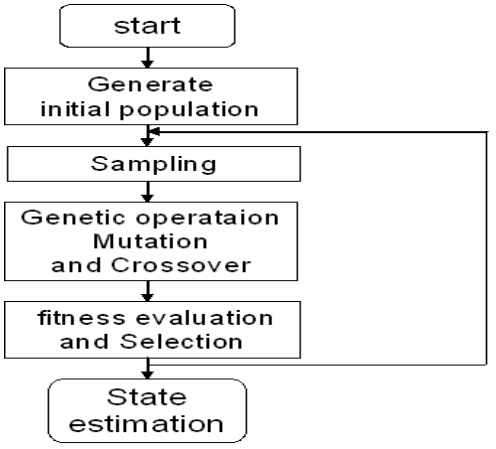

Fig. 1 Procedure of the Genetic Filter

The procedure of the genetic filter consists of three steps: sampling step, the genetic operation step and the resampling step. In the samling step, a new population of chromosomes is generated from the one time earlier population X_{t-1} according to the state transition probability $p(x_k | u_k, x_k^m)$ (sampling).

In the second step of proposed algorithm, the genetic operations such as crossover and mutation are applied to the temporary set of the chromosomes. Crossover and Mutation play the important roles in solving the inherent particle deprivation problem while selection generate degeneracy problem.

The third step is that the fitness values of the selected chromosomes are evaluated (fitness evaluation). The fitness value of each chromosome is the measurement probability $p(z_k | x_k^m)$. Then, the chromosomes are resampled with replacement according to their fitness values (resampling). The new population X_t represents the posterior density $p(x_k | z_k, u_k, x_{k-1})$.The algorithm of the genetic filter is given below in Table II.

TABLE II
ALGORITHM OF THE GENETIC FILTER

1: Algorithm Genetic Filter (x_{k-1}^m, u_k, z_k)
2: $[\{x_k^m, w_k^m\}_{m=1}^N] = \mathrm{GF}[\{x_{k-1}^m, w_{k-1}^m\}_{m=1}^N, z_k]$
3: For $m = 1:N$
4: Draw $x_t^m \sim p(x_k | u_k, x_{k-1}^m)$
5: Do genetic operation as table (3) and (4)
6: Assign the particle a weight w_k^m according to $p(z_k | x_k^m)$
7: Select x_k^m according to w_k^m by roulette wheel selection method
8: End

A. Sampling

The sampling step is same as standard particle filter, which describe in chapter 2. In the sampling step, samples are generated according to $p(x_k \mid u_k, x_{k-1}^m)$.

B. Genetic Operation

The Genetic operations consist of two operators, crossover and mutation. Crossover is to choose a pair of chromosomes promoted in the selection and mate them to produce a new pair of offsprings. The mutation operator is to change a chromosome randomly. In this paper, we use the arithmetic crossover [6], and residual mutation.

The arithmetic crossover produces the offspring chromosomes from parent chromosomes and the produced offsprings are located in convex set region of parent chromosomes, so it can search the convex set region where is not existed by its parent chromosomes. The arithmetic crossover is represented in Table III.

TABLE III
ARITHMETIC CROSSOVER

$\left(x_k^m\right)' = a \cdot x_k^m + (1-a) \cdot x_k^n$

$\left(x_k^n\right)' = a \cdot x_k^n + (1-a) \cdot x_k^m$

where a is a random number between 0 and 1, x_k^m, x_k^n are parent chromosomes at k-th time population, and $\left(x_k^m\right)', \left(x_k^n\right)'$ are produced chromosomes at k-th time population.

Residual mutation, which is proposed in this paper, is offered to overcome particle deprivation problems. The detail of residual mutation is that the mutated chromosome is made by sum of original chromosome value and normal random variable with mean and variance, $N(z_k - x_k^m, q)$. Since the measurement residual value is used in mutation step, even if particle deprivation problem occurs while track out target, we can move the location of some chromosomes near the real state by residual mutation. The detail of residual mutation is shown in Table IV.

TABLE IV
RESIDUAL MUTATION

$\left(x_k^m\right)' = x_k^m + c$

where c is random variable with $N(z_k - x_k^m, q)$, x_k^m is a parent chromosome at k-th time population, q is a predefined constant and $\left(x_k^m\right)'$ is a produced chromosomes at k-th time population.

However, since the particle deprivation is not always happened, if the particle deprivation problem doesn't be happened, the crossover and mutation are not improve the tracking performance, even thought they can cause another computational burden. So the crossover and mutation probability are governed whether the particle deprivation is happened or not. So, in this paper, we decide the crossover and mutation probability according to tracking performance. If the recently tracking performance is small, the most chromosomes are not located around the real state, so we should assign crossover and mutation probability larger values. And if it is not, we can assign crossover and mutation probability smaller values. By adapting the mutation and crossover probability values, we can get both the better tracking performance and efficiency of computational complexity. The detail of determination of crossover and mutation probability is represented as equation (4).

$$\rho_C = Min\left(10, \frac{1}{\text{averaging position tracking performance from (k-6)th to (k-1)-th time}}\right)\% \qquad (2)$$

$$\rho_M = Min\left(5, \frac{1}{\sqrt{\text{averaging position tracking performance from (k-6)th to (k-1)-th time}}}\right)\%$$

where ρ_C is crossover probability, ρ_M is mutation probability and $Min(a,b) = \begin{cases} a & \text{if } a \le b \\ b & \text{otherwise} \end{cases}$.

Equation (4) means that the crossover and mutation probability are depend on their tracking performance, however, it is bounded some constant value.

By genetic operation, we can enlarge the search space, so although the chromosomes are not located near the real state, the chromosomes can be shifted around real state by mutation and crossover.

C. Selection

After genetic operation, we select the chromosomes which have large fitness values. In the selection step, the sampled and produced chromosomes by genetic operations are resampled with replacement according to their fitness values, $p(z_t \mid x_t^m)$. In the standard genetic algorithm, there are several selection methods such as ranking method or elitism method.[6] In this genetic filter, we use the well known roulette wheel method in the resampling step. The basic idea of the roulette wheel method is that the higher the fitness value is, the more likely the chromosome is selected (or resampled) in the genetic algorithm (or filter). The fitness value of each chromosome is the measurement probability

$$fitness(x_k^m) = p(z_k \mid x_k^m) \qquad (3)$$

Shown in Table V is the algorithm of the roulette wheel selection.

TABLE V
ALGORITHM OF THE ROULETTE WHEEL SELECTION.

1: Calculate $Total\ \ fitness(f_{sum}) = \sum fitness$

2: Choose a random number, R_s, between 0 and f_{sum}

3: Add together the fitness of the population members (one at a time) stopping immediately when the sum is greater than R_s.

The last individual added is the selected individual and a copy is passed to the next generation.

As in line 6 of Table II, the resampled chromosome is added to the population set X_t of the chromosomes. The resulting population, the set of chromosomes x_k^m, represents posterior density probability, $p(x_k \mid x_{k-1}, u_k, z_k)$, which is

$$p(x_k \mid x_{k-1}, u_k, z_k) \approx \frac{1}{M}\sum_{m=1}^{M}\delta(x - x_k^m) \cdot \quad (4)$$

where M is tge number of chromosomes and $\delta(x) = \begin{cases} 1, & \text{if } x = 0 \\ 0, & \text{otherwise} \end{cases}$.

IV. Simulation Results

In this section, we present an example to show the effectiveness of the suggested filter. In this simulation, the target is assumed to move along x-direction and trajectories is governed by

$$\begin{bmatrix} x_k \\ \dot{x}_k \\ \ddot{x}_k \end{bmatrix} = \begin{bmatrix} 1 & T & \frac{T^2}{2} \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{k-1} \\ \dot{x}_{k-1} \\ \ddot{x}_{k-1} \end{bmatrix} + \begin{bmatrix} \frac{T^2}{2} \\ T \\ 1 \end{bmatrix} w_k \quad (5)$$

$$z_k = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_k \\ \dot{x}_k \\ \ddot{x}_k \end{bmatrix} + v_k \cdot \quad (6)$$

where x_k, $\dot{x}_k$, and $\ddot{x}_k$ are distance, velocity, acceleration of the target respectively. T is the radar scan time. Noise w_k and v_k are assumed to be mutually independent and be a white Gaussian noise with zero mean and variances $Q = 0.1^2$ and $R = 2^2$. Shown in Fig. 2 are the estimation errors of the two filters, PF and GF. The dashed line denotes the GF, the dotted line denotes the PF. We use three hundred particles for each filter.

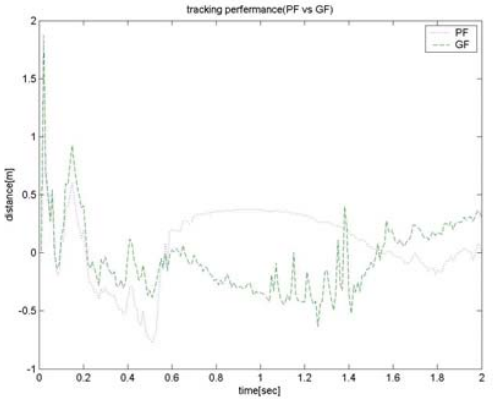

Fig. 2 The tracking error of PF (red line) and GF(blue line)

To show in Fig. 2, the performance of GF is not different from it of PF. In GF case, the chromosomes spread in all directions, however, highly possible chromosomes are selected by selection stage. The root mean square error of each state is shown below Table VI.

TABLE VI
Root Mean Square Error of Each Filtering Method (m)

	Particle Filter	Genetic Filter
Distance RMSE	0.3259	0.3222
Velocity RMSE	1.2297	0.9861
Acceleration RMSE	1.0015	0.9245

Fig. 3 represents the maneuvering target case. As you see in Fig. 3, if the target is maneuvering fast, the PF cannot track out the real state, because, in PF case, the particles cannot be moved fast through all stage of particle filter: sampling, calculation of importance weight and resampling. But, GF is able to search the chromosomes wider than particle filter by using genetic operators. The dotted line represents the PF, the dashed line represents the GF, and the real state is represented by the solid line.

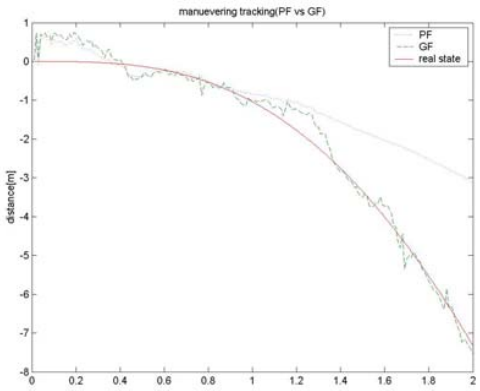

Fig. 3 Maneuvering Target Tracking – PF (dotted line), GF (dashed line) and state (solid line)

The effect mutation is represented in Fig. 4. In PF case, if state jump somewhere (like kidnap problem of mobile robot [3]), the particles are not located around the real state, the particles cannot find real state, in PF case. But, in GF case, although the chromosomes are not located around the real state, the chromosomes can be moved by residual mutation.

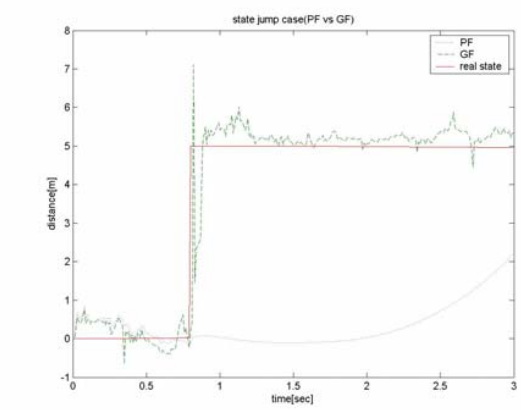

Fig. 4 Particle deprivation problem case – PF (dotted line), GF(dashed line) and state(solid line)

V. Conclusion

In this paper, we have proposed a new particle filter with varying number of particles. By adjusting the number of particles, we can reduce the computational load while maintaining the tracking performance. Therefore, the proposed method is very suitable for practical applications. Further, although the resampling step can resolve the particle degeneracy problem, it cannot be implemented in a parallel hardware. Since there is no resampling (selection) step in the PFVaNP, the proposed method is very suitable for the hardware implementation.

References

[1] Kalman, R.E, "A New Approach to Linear Filtering and Prediction Problems," *Trans, ASME*, J. Basic Engineering, vol.82, pp34-45, Mar. 1960.
[2] B. Ristic, S. Arulampalam, and N. Gordon, Beyond the Kalman Filter: Particle Filters for Tracking Applications, Boston, London: Artech House., 2004.
[3] S. Thrun, W. Burgard, and D. Fox, Probabilistic Robotics, Cambridge, Massachisetts, London, England: The MIT Press, 2005.
[4] J.E. Handschin and D.Q. Mayne, "Monte Carlo techniques to estimate the conditional expectation in multi-stage non-linear filtering," Intern. Journal of Control, vol.9, no. 5, pp.547-559, 1969.
[5] K. Uosaki, Y. Kimura and T. Hatanaka, "Nonlinear State Estimation by Evolution Strategies based Particle Filters," Proc. Congress on Evolutionary Computation Vol. 1, pp884 – 890, 19-23-th June 2004.
[6] Z. Michalewicz. Genetic Algorithm + Data Structure = Evolution Programs. Berling Heidelberg, New York: Springer-Verlag, third ed., 1996.

Seomgkeun Park was born in Korea, in 1981. He received the B.S. degrees and M.S. in electrical and electronic engineering from Yonsei University, Seoul, Korea, in 2004, 2006, respectively. He is currently a Ph.D. candidate of Dept. of Electrical and Electronic engineering in Yonsei University. His research interests include artificial intelligence and signal processing.

JaePil Hwang was born in Korea, in 1977. He received the B.S. degrees and M.S. in electrical and electronic engineering from Yonsei University, Seoul, Korea, in 2003, 2005, respectively. He is currently a Ph.D. candidate of Dept. of Electrical and Electronic engineering in Yonsei University.

KyungJin Rou was born in Korea, in 1980. He received the B.S. degrees in electrical and electronic engineering from Yonsei University, Seoul, Korea, in 2006. He is currently a MS. candidate of Dept. of Electrical and Electronic engineering in Yonsei University.

Euntai Kim was born in Seoul, Korea, in 1970. He received the B.S. (summa cum laude) and the M.S. and the Ph.D. degrees in electronic engineering, all from Yonsei University, Seoul, Korea, in 1992, 1994 and 1999, respectively.From 1999 to 2002, he was a full-time lecturer in the Department of Control and Instrumentation Engineering at Hankyong National University, Kyonggi-do, Korea. Since 2002, he has joined the faculties of the School of Electrical and Electronic Engineering at Yonsei University, where he is currently an associate professor. His current research interests include computational intelligence and its application to intelligent service robot and intelligent home network. Dr. Kim is an associate editor of International Journal of Control and Systems.