

# Particle Filter via Chopthin Resampling for Target Tracking with Glint Noise

C Liu<sup>1,\*</sup>, S C Yang<sup>1</sup> and L D Wang<sup>2</sup>

<sup>1</sup>Army Engineering University, No.97 Heping West Road, Shijiazhuang, Hebei, 050003, China

<sup>2</sup>National Key Laboratory of the Complex Electromagnetic Environment Effect of Electronic Information System, Luoyang, Henan, 471003, China

\*Corresponding e-mail:1561174321@qq.com

**Abstract.** Traditional resampling methods in particle filter suffer from the disadvantage of the low filtering accuracy, the severe fluctuation of effective sample size (ESS) and the great influence of parameter setting. A particle filter algorithm based on chopthin resampling is proposed in this paper. Traditional resampling methods produce a set of equally weighted particles and only occur when ESS below a threshold. In contrast to this, the proposed algorithm enforces an upper bound on the ratio between the weights and can be implemented in every step. Simulation shows that the new method outperforms traditional resampling methods in filtering accuracy, ESS stability and computational efficiency, especially when sample size or observation noise is small.

## 1. Introduction

In target tracking, changes in the target aspect with respect to the radar can cause random wandering of the angle tracking, which is also called angle fluctuations or target glint. The glint is heavy-tailed and non-Gaussian and may severely affect the tracking accuracy, especially that of large targets at short ranges. Kalman filters, such as extended Kalman filter (EKF), unscented Kalman filter (UKF) and cubature Kalman filter (CKF), have been widely applied in target tracking. However, their performance degrades when the observation noises are not Gaussian.

One possible alternative approach to this problem is particle filter (PF). Since the publication of [1], PF has become popular for nonlinear state-space models where the noises of the model are non-Gaussian. During the past two decades, PF has been successfully applied in target tracking. Using the PF method, various distributions of interest are approximated by the generated particles as well as associated weights. Almost all PF methods are based on three operations: particle propagation, weight computation and resampling. Particle generation and weight computation amount to the generation of particles and assignment of weights, whereas resampling replaces one set of particles and their weights with another set.

Resampling, which is one of the key steps of PF, is highly important to PF. Without resampling, PF will suffer from weight degeneracy, i.e., only a few particles are significantly weighted, whereas the weight of most of the particles is almost zero. This means that the obtained estimates will be inaccurate and will have unacceptably large variances. Consequently, various resampling schemes (traditional resampling as they may be called), include multinomial resampling [1], stratified resampling [2], systematic resampling [3] and residual resampling [4], have been proposed.



Generally speaking, resampling is a process in which one samples from the original random measure  $\chi_t = \{x_t^i, w_t^i\}_{i=1}^n$  to create a new random measure  $\tilde{\chi}_t = \{\tilde{x}_t^i, \tilde{w}_t^i\}_{i=1}^N$ . In traditional resampling methods, the weights of the particles after resampling are all equal. However, the benefits of resampling are accompanied with potential side effects such as sample impoverishment and the speedup of implementation of PF. Thus, traditional PF often only perform the resampling step if the effective sample size (ESS) drops below a fixed threshold. This avoids resampling if the weights are relatively even and thus reduces the noise being introduced through the resampling. On the other hand, this causes the ESS to fluctuate over time.

Recently, a new resampling algorithm called chopthin was proposed [5]. In contrast to standard resampling methods, the algorithm does not produce a set of equally weighted particles; instead it merely enforces an upper bound  $\eta$  on the ratio between the weights. Chopthin will not alter the weights much if they are relatively even, thus it can be executed at every step of a particle filter. Simulation studies show that chopthin consistently outperforms the other resampling methods and implicitly controls the ESS.

This paper is organized as follows. After Section 1 which gives the introduction, a brief review of PF and resampling is presented in Section 2 and the chopthin resampling is discussed in Section 3. Then, Section 4 provides the numerical simulation result via comparison of chopthin resampling with numerous typical resampling methods and finally Section 5 summarizes the results and proposes some recommendations for further research.

## 2. State estimation by the particle filter

Consider a discrete time nonlinear stochastic system given by the state equation (1) and the measurement equation (2):

$$x_t = f(x_{t-1}, u_t), t = 1, 2, \dots \quad (1)$$

$$y_t = h(x_t, v_t), t = 1, 2, \dots \quad (2)$$

Where the vectors  $x_t \in \mathbb{R}^n$  and  $y_t \in \mathbb{R}^m$  represent the state of the system and the measurement at time  $t$ , respectively;  $u_t \in \mathbb{R}^n$  and  $v_t \in \mathbb{R}^m$  are independent state and measurement noises, with known probability density function (pdf)  $p(u_t)$  and  $p(v_t)$ ;  $f: \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}^n$  and  $h: \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^m$  are known functions; and the pdf  $p(x_0)$  of the initial state  $x_0$  is known. The system can alternatively be described by the transition pdf  $p(x_t | x_{t-1})$  and the measurement pdf  $p(y_t | x_t)$ , which can be obtained from equation (1) and equation (2), respectively.

The objective of PF here is the sequential estimation of filtering distribution  $p(x_t | y_{1:t})$ , which can be expressed in terms of the filtering distribution  $p(x_{t-1} | y_{1:t-1})$ , i.e., in a recursive form by

$$p(x_t | y_{1:t}) \propto \int p(y_t | x_t) p(x_t | x_{t-1}) p(x_{t-1} | y_{1:t-1}) dx_{t-1} \quad (3)$$

Where the symbol  $\propto$  signifies “proportional to”. This update can be implemented analytically in a very few cases, therefor, one resorts to approximations. The basic idea of PF is to represent the belief  $p(x_{t-1} | y_{1:t-1})$  by a set of  $N$  weighted samples  $\chi_{t-1} = \{x_{t-1}^i, w_{t-1}^i\}_{i=1, \dots, N}$ , i.e.,

$$p(x_{t-1} | y_{1:t-1}) \approx \sum_{i=1}^N w_{t-1}^i \delta(x_{t-1} - x_{t-1}^i) \quad (4)$$

Where  $\delta(\cdot)$  is the Dirac delta impulse, and  $w_{t-1}^i$  is the weight associated with the system state  $x_{t-1}^i$ . As  $N \rightarrow \infty$ , the discrete weighted approximation in equation (4) approaches to the true posterior density  $p(x_{t-1} | y_{1:t-1})$ .

First basic step of PF is to draw particles  $x_t^i$  from  $p(x_t | y_{1:t})$ , which is referred to as particle propagation or importance sampling because a particle  $x_{t-1}^i$  is moved forward in time and is a parent

of  $x_t^i$  [6]. However,  $p(x_t|y_{1:t})$  is unknown in most cases, therefore, one has to use an importance density function  $\pi(x_t)$ . A simplest and widely used choice is

$$\pi(x_t) = p(x_t|x_{t-1}) \quad (5)$$

The second basic step is computing weights of the particles. As particles generated from  $\pi(x_t)$  is different from  $p(x_t|x_{t-1})$ , the particle weights have to be modified according to

$$w_t^i \propto w_{t-1}^i p(y_t|x_t^i) p(x_t^i|x_{t-1}^i) / \pi(x_t^i) \quad (6)$$

Subsequently, the particle weights are normalized so that they sum up to one, i.e.

$$w_t^i = w_{t-1}^i / \sum_{i=1}^N w_{t-1}^i \quad (7)$$

PF using only the aforementioned two steps has sample degradation problem, i.e. a few particles have most of the weight and the rest of the particle weights are negligible. For this reason, PF needs a third step, referred to as resampling. Formally, resampling is a process in which one modify the random measure  $\chi_t = \{x_t^i, w_t^i\}_{i=1}^n$  to a new random measure  $\tilde{\chi}_t = \{\tilde{x}_t^i, \tilde{w}_t^i\}_{i=1}^N$ . Then, the random measure  $\chi_t$  is replaced with  $\tilde{\chi}_t$ . In the process, particles with low importance weights are most likely removed and those of large weights are replicated. Although the number of resampled particles  $N$  is not necessarily equal to the number of propagated numbers  $n$ , traditional resampling methods keep  $n = N$  and the weights after resampling are equal. However, resampling can also delete good samples from the sample set, causing sample impoverishment. Thus, resampling should only be applied when necessary and therefore it is important to find a criterion when to perform a resampling step. Effective sample size (ESS) as a mean to measure the variation of the weights is defined as [7]

$$N_{eff} = 1 / \sum_{i=1}^M (w_i^j)^2 \quad (8)$$

Where  $N_{eff}$  varies in the  $[1, M]$  range. Resampling is required when  $N_{eff}$  falls below a selected threshold  $N_{thr}$ . This avoids resampling if the weights are relatively even and thus reduces the noise being introduced through the resampling. This results in ESS to fluctuate over time.

The general algorithm of the PF can be summarized in table 1 as follows:

**Table 1.** SIR particle filter.

Input: target number of particles $N$ ; ESS threshold $N_{th}$ ; resampling scheme $r$ ; observations $y_{1:T}$
Output: weighted particles $(w_t, x_t^i)_{1:n_t}$ , for $t=1, \dots, T$
Sample $x_0^i \sim p(x_0)$ , $i=1, \dots, N$
$w_i = 1/N$ , $i=1, \dots, N$
Let $n_0 = N$
for $t=1, \dots, T$ do
Sample $x_t^i \sim p(x_t x_{t-1}^i)$ , $i=1, \dots, n_{t-1}$ $w_i = w_{t-1}^i p(y_t x_t^i)$ , $i=1, \dots, n_{t-1}$
Normalize $w_i = w_{t-1}^i / \sum_{i=1}^{n_{t-1}} w_{t-1}^i$
if $ESS(w) \leq N_{th}$ , then
Run $r$ to get a set of particles $(w_i, x_t^i)_{1:n_t}$
end if
end for

### 3. Chopthin resampling

Recently, a new resampling algorithm called chopthin was proposed in [5]. In contrast to standard resampling methods, the algorithm does not produce a set of equally weighted particles; instead it merely enforces an upper bound  $\eta$  on the ratio between the weights. As mentioned in Section II, traditional resampling method only occurs when ESS drops below a fixed threshold. In contrast to this, chopthin can be executed at every step of a particle filter. This is because chopthin will not alter the weights much if they are relatively even.

Denote the particle set before resampling as  $\chi_t = \{x_i, w_i\}_{i=1}^n$  and after resampling as  $\tilde{\chi}_t = \{\tilde{x}_j, \tilde{w}_j\}_{j=1}^N$ . Let  $G$  be the  $\sigma$ -field generated by  $w_1, \dots, w_n$  and let  $C^i$  be the number of times that the  $i$ th particle is resampled. The constraints that chopthin should satisfy are as follows:

$$(1) E(C^i \tilde{w}_i | G) = w_i \quad \forall i \text{ (Unbiasedness)}$$

$$(2) \sum_{i=1}^n C^i = N \text{ (Target count)}$$

$$(3) \sum_{i=1}^n w_i = \sum_{j=1}^N \tilde{w}_j \text{ (Conserve weight)}$$

$$(4) \tilde{w}_i / \tilde{w}_j \leq \eta \quad \forall i, j \text{ (Bounded ratio)}$$

Property (1) ensures that the total weight of a particle is equal before and after resampling. Property (2) ensures that  $N$  particles are returned after resampling. Property (3) ensures that the sum of all the weights before and after resampling are equal. Property (4) bounds the ratio of weights returned from chopthin.

The general algorithm of the chopthin can be summarized in table 2 as follows:

**Table 2.** Generic chopthin.

Input: particle weights $(w_i)_{1:n}$ ; maximal weight ratio $\eta$ ; target number of particles $N$ ; function $h_a^\eta : [0, \infty) \rightarrow [0, \infty)$	
Output: ancestors $I \in \{1, \dots, n\}^N$ , weights $\tilde{w} \in [0, \infty)^N$	
Let $a$ be a solution to $\sum_{i=1}^n h_a^\eta(w_i) = N$	//Find $a$
Let $L = \{j: w_j < a\}$ and $U = \{j: w_j \geq a\}$	
Let $I = ()$ and $\tilde{w} = ()$	
Draw $u \sim U(0, 1)$	
for $i \in L$ do	//Thin
$u = u + h_a^\eta(w_i)$	
if $u \geq 1$ then	
Append $(i)$ to $I$ and $(a)$ to $\tilde{w}$	
$u = u - 1$	
end if	
end for	
Let $N_L = \text{length}(I)$ , $N_U = N - N_L - \sum_{i \in U} \lfloor h_a^\eta(w_i) \rfloor$ and $\varsigma := \sum_{i \in L} w_i - aN_L / \sum_{i \in U} \{h_a^\eta(w_i)\}$	
Systematic $\{ \{h_a^\eta(w_j)\}; j \in U \}; N_U \}$ ; returns $m^j$ for $j \in U$	
for $i \in U$ do	//Chop
$c = \lfloor h_a^\eta(w_i) \rfloor + m^i$	
append $(i, \dots, i) \in N^c$ to $I$	
$\hat{w}_i = w_i + \varsigma \{h_a^\eta(w_i)\}$	

---

```

    append ( $\hat{w}_i/c, \dots, \hat{w}_i/c \in \mathbb{R}^c$  to  $\tilde{w}$ 
    return  $I, \tilde{w}$ 
end for

```

---

The key steps of generic chopthin include find  $a$ , thin and chop. The parameter  $a$  is a threshold of weights, i.e., particles with weights below  $a$  will be thinned and particles with weights above  $a$  will be chopped. For a particle with weight  $w$ , the expected number of offspring from chopthin will be  $h_a^\eta(w)$ , which is a given function that depend on  $a$  and  $\eta$ . Hence, first key step of chopthin is choose  $h_a^\eta(w)$  and find  $a$  such that

$$\sum_{i=1}^n h_a^\eta(w_i) = N \quad (9)$$

One such choice given by [5] is

$$h_a^\eta(w) = \begin{cases} w/a & \text{if } w < a \\ 1 & \text{if } a \leq w < \eta a/2 \\ 2w/\eta a & \text{if } w \geq \eta a/2 \end{cases} \quad (10)$$

We use table 3 to solve equation (9) using equation (10) for  $a$ .

**Table 3.** Fast determination of  $a$ .

---



---

Input: particle weights $w_i$ ; maximal weight ratio $\eta$ ; target number of particles $N$
Output: $a > 0$ such that $\sum_{i=1}^n h_a^\eta(w_i) = N$
$w^u = w^l = w$ , $s^l = 0$ , $c^m = 0$ , $s^u = 0$ , $c^u = 0$
while $w^u \neq \emptyset$ or $w^l \neq \emptyset$ do
if $ w^l  \geq  w^u $ , then sample $a$ uniformly from $w^l$ and let $b = \eta a/2$
else, sample $b$ uniformly from $w^u$ and let $a = 2b/\eta$
end if
$h = s^l/a + \sum_{v \in w^l} \min(v/a, 1) + c^m + \sum_{v \in w^u} \max(v/b - 1, 0) + s^u/b - c^u$
if $h = N$ , then return $a$
if $h > N$ , then $w^l = \{v \in w^l; v > a\}$ , $s^l = s^l + \sum_{v \in w^l} v \mathbf{I}(v \leq a)$ , $w^u = \{v \in w^u; v > b\}$
else $c^m = c^m + \sum_{v \in w^l} \mathbf{I}(v \geq a)$ , $s^u = s^u + \sum_{v \in w^u} v \mathbf{I}(v \geq b)$ , $c^u = c^u + \sum_{v \in w^u} \mathbf{I}(v \geq b)$ , $w^u = \{v \in w^u; v < b\}$
end if
return $a = (s^l + 2s^u/\eta)/(N - c^m + c^u)$

---



---

It is proved by [5] that the expected effort of algorithm in table 3 is linear in  $n$ , and overall the expected computational effort is  $O(\max(n, N))$ . Algorithm 3 works by determining which weights are above or below  $a$  and which weights are above or below  $\eta a$ , without fully knowing  $a$  yet.

Second key step of chopthin is Thinning, which determines the new weight and the number of offspring for particles with weights below  $a$ . With this step, particles with small weights will have 0 or 1 offspring, which is determined by systematic resampling on  $\{h_a^\eta(w_i) : w_i < a\}$ . This step returns  $N_L$  particles with weight  $a$  such that  $E(N_L | G) = \sum_{i: w_i < a} w_i / a$ . The total weight of the surviving particles is  $aN_L$ . Thus, the total sum of the weights may have changed. We compensate for this using  $\zeta$  in the chopping step, thus ensuring property (3).

Last key step, the chopping step, determines how the particles with weights above  $a$  are subdivided. Before chopping, the original weight  $w_i$  is adjusted to  $\hat{w}_i$  using  $\zeta$ , where

$\hat{w}_i = w_i + \zeta\{h_a^\eta(w_i)\}$  and  $\{x\} := x - \lfloor x \rfloor$ . This adjustment ensures that the totals sum of the weights is conserved, property (3) and that the chopped weights are unbiased. Each large weight will receive  $c^i = \lfloor h_a^\eta(w_i) \rfloor + m^i$  offspring. The  $m^i$  are determined by a second systematic resampling step on the fractional parts  $\{h_a^\eta(w_i)\}$ . Further, the entire algorithm will return exactly  $N$  particles. Thus property (2) is satisfied.

#### 4. Simulation

In this section, the performance of the proposed approach is examined by simulating a bearings-only target tracking problem. The kinematics of the target can be modeled by the following nonlinear equation:

$$x_t = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix} x_{t-1} + \begin{bmatrix} T^2/2 & 0 \\ T & 0 \\ 0 & T^2/2 \\ 0 & T \end{bmatrix} \begin{bmatrix} v_{1,t} \\ v_{2,t} \end{bmatrix} \quad (11)$$

Where the state of the target is  $x_t = [x_{1,t}, x_{2,t}, x_{3,t}, x_{4,t}]$ ,  $x_{1,t}$  and  $x_{3,t}$  denote positions,  $x_{2,t}$  and  $x_{4,t}$  denote velocities in the x and y directions at the Cartesian coordinates, respectively;  $T$  is the sampling period; the process noise  $v_{1,t} \sim N(0, \sigma_{v1}^2)$  and  $v_{2,t} \sim N(0, \sigma_{v2}^2)$  are independent Gaussian white noise.

A radar located at the origin is used to measure the angle between the target and the X-axis. Hence, the measure equation is described as

$$\theta_t = \arctan(x_{3,t}/x_{1,t}) + w_t \quad (12)$$

Where  $w_t$  denote the glint noise, which is often modelled as a mixture of a Gaussian distribution with high probability and a heavy-tailed Laplacian distributed noise with low probability, i.e.,

$$p(w) = (1 - \varepsilon) p_G(w) + \varepsilon p_L(w) \quad (13)$$

Where  $\varepsilon$  is the glint probability and the subscripts G and L stand for Gaussian and Laplacian, respectively, i.e.,

$$p_L(w) \triangleq L(w; 0, \sigma_L) \triangleq \frac{1}{2\eta} e^{-|w|/\sigma_L} \quad (14)$$

$$p_G(w) \triangleq N(w; 0, \sigma^2) \triangleq \frac{1}{\sqrt{2\pi}\sigma} e^{-w^2/2\sigma^2} \quad (15)$$

The numerical values of the parameters of the dynamic system are as follows:

Sampling period  $T = 1$  s. Process noise standard deviation  $\sigma_{v1} = \sigma_{v2} = 0.001$  m/s<sup>2</sup>. Measurement noise parameters  $\varepsilon = 0.05$ ,  $\sigma^2 = 0.4$  mrad<sup>2</sup>,  $\sigma_L = 3$  mrad. Initial state of the target  $x_0 = [-0.05 \text{ m}, 0.001 \text{ m/s}, 0.7 \text{ m}, -0.055 \text{ m/s}]^T$ , associated covariance  $P_{00} = \text{diag}[0.25 \text{ m}^2, 2.5 \times 10^{-5} \text{ m}^2/\text{s}^2, 0.09 \text{ m}^2, 10^{-4} \text{ m}^2/\text{s}^2]^T$ . The initial state estimate  $\hat{x}_{00} = [0, 0, 0.4 \text{ m}, -0.05 \text{ m/s}]^T$ .

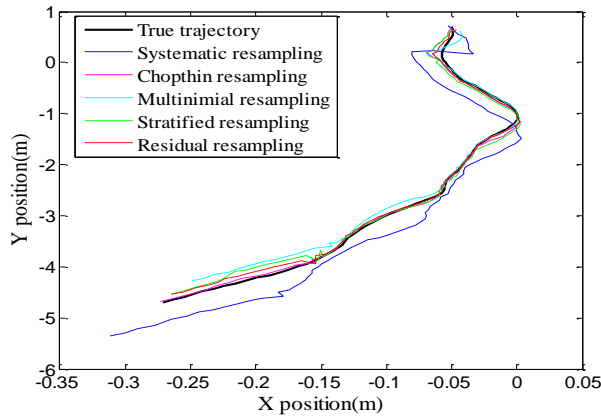
Numerous typical resampling methods are compared with chopthin resampling via simulations in terms of ESS and estimation accuracy. The typical resampling methods used for simulation include systematic, multinomial, stratified, residual. The detailed pseudocodes for selected algorithms are presented in [8].

The simulation adopts different resampling methods but the same observation at all time-steps and the same starting number of particles 2000 are adopted for each filter. All resampling algorithms maintain a constant number of particles that is equal to 2000.

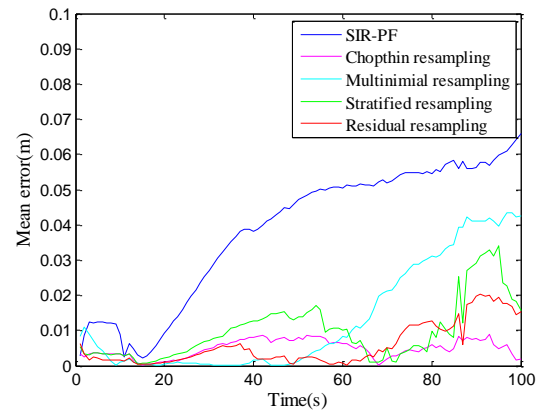
The tracking scene in one trial is given in figure 1. The mean tracking position error of a total 100 trials are given in figure 2. The tracking position error is the Euler distance between the estimate and the true position of the target which is defined as

$$\text{error} = \sqrt{(x_{1,t} - \hat{x}_{1,t})^2 + (x_{3,t} - \hat{x}_{3,t})^2} \quad (16)$$

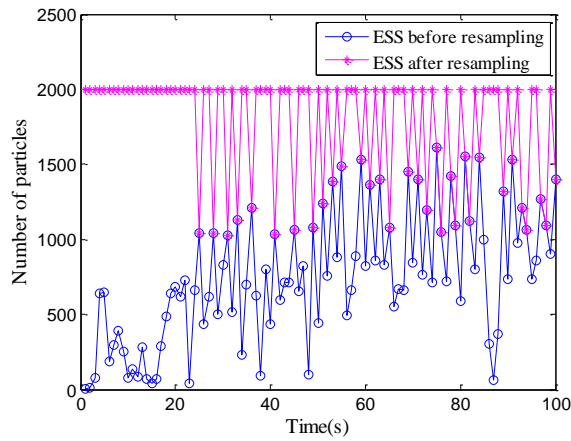
Where  $[\hat{x}_{1,t}, \hat{x}_{3,t}]^T$  is the x-y position estimate of the target at time t. Figure 3 shows the effect of different resampling methods on ESS in one trial during 100s.  $N_{eff}$  used in the traditional resampling methods is set to 1000,  $\eta$  used in the chopthin is  $3+\sqrt{8}$ .



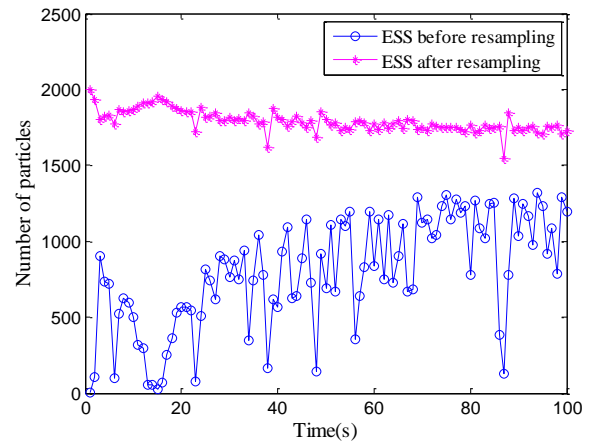
**Figure 1.** Target tracking scene in one trial



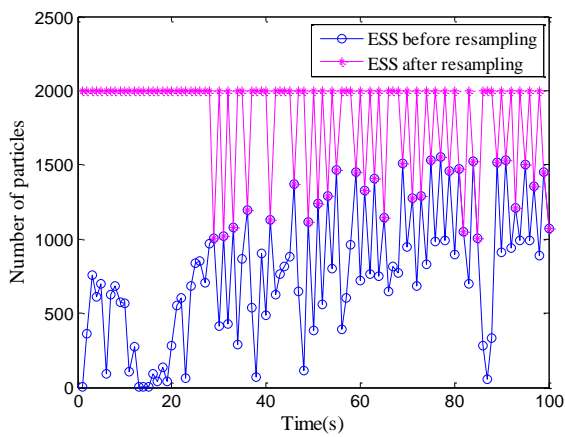
**Figure 2.** Mean tracking error of 1000 trials.



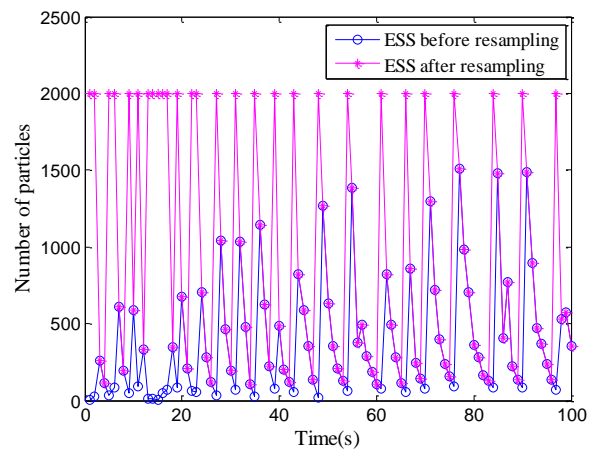
(a)



(b)

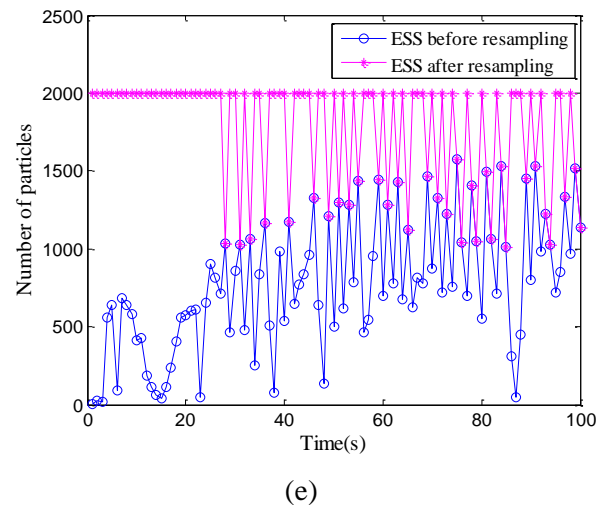


(c)



(d)





**Figure 3.** ESS before and after resampling for different resampling methods.

Overall, the simulation results demonstrate that, first of all, the traditional resampling methods can rarely produce very different results except systematic resampling, which is worse than other methods, second, the chopthin methods consistently outperforms traditional resampling methods, and third, chopthin resampling implicitly controls the ESS than traditional resampling methods, as resampling for traditional resampling methods only occurs if the ESS has dropped below  $N_{eff}$ .

## 5. Conclusion

Resampling is an essential procedure for particle filtering, which is of both theoretical and practical importance. In this paper, a new target tracking algorithm via particle filter with chopthin resampling is proposed. Simulations demonstrated that the chopthin methods consistently outperforms traditional resampling methods used in particle filters. The simulations also demonstrated that chopthin can be used at every iteration with no detrimental effects. We have also shown that imposing a bound on the ratio between weights implicitly controls the ESS.

## References

- [1] N. Gordon, D. Salmond, and A. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," IEE Proc., F Radar Signal Process., vol.140, no.2, pp. 107-193, 1993
- [2] G. Kitagawa, "Monte Carlo filter and smoother and non-Gaussian nonlinear state space models," J.Comput.Graph.Stat., vol. 5,no. 1,pp. 1-25,1996
- [3] J. Carpenter, P. Clifford, and P. Fearnhead, "An improved particle filter for nonlinear problems," IEE Proc., Radar Sonar Navigat., vol. 146, no. 1, pp. 2–7, 1999.
- [4] E. R. Beadle and P. M. Djuric', "A fast-weighted Bayesian bootstrap filter for nonlinear model state estimation," IEEE Trans. Aerosp. Electron. Syst., vol. 33, no. 1, pp. 338–343, 1997.
- [5] Gandy, D.H. Lau,"The chopthin algorithm for resampling", IEEE Transactions on Signal Processing, 2016,64(16):4273-4281
- [6] D. B. Rubin, "A noniterative sampling/importance resampling alternative to the data augmentation algorithm for creating a few imputations when fractions of missing information are modest: The SIR algorithm," J. Amer. Statist. Assoc., vol. 82, no. 398, pp. 543–546, 1987.
- [7] Kong, J. S. Liu, and W. H. Wong, "Sequential imputations and Bayesian missing data problems," J. Amer. Statist. Assoc., vol. 9, no. 425, pp. 278–288, 1994.
- [8] Li, T., Bolic,M., and Djuric, P.M., 2015. Resampling methods for particle filtering: classification, implementation, and strategies. IEEE Signal Processing, 2015, 32(3):70-86