Received May 17, 2021, accepted June 12, 2021, date of publication June 22, 2021, date of current version July 1, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3091595

An Adaptive Particle Filter for Target Tracking Based on Double Space-Resampling

ZHENG GONG[1], GANG GAO[2], AND MINGANG WANG[1]
[1]School of Astronautics, Northwestern Polytechnical University, Xi'an 710072, China
[2]Luoyang Optoelectro Technology Development Center, Luoyang 471000, China

Corresponding author: Zheng Gong (gong8487424@163.com)

ABSTRACT Particle filter has been widely applied in nonlinear target tracking due to the ability to carry multiple hypothesis and relaxation of linearity/Gaussian assumption. In this paper, an adaptive double space-resampling particle filter is proposed to increase the efficiency and robustness of filtering by adjusting the sample size. The first resampling operation, adopted before the prediction of samples, generates a larger number of equal-weighted samples and some auxiliary samples to enhance the robustness of filtering. The second resampling, adopted between the prediction and updating step, decreases the sample size for weight updating which is the most time consumption part of particle filter. The particle space sampling technique is used in both space-resampling, which adjusts the sample size according to not only the weights of samples but also their spatial distribution. The efficiency of filtering is improved and the robustness of algorithm is enhanced, simultaneously. The degeneracy and sample impoverishment problems can be counteracted. Simulation and experiment contrast results demonstrate that the proposed method is robust and efficient.

INDEX TERMS Monte Carlo, particle filter, space sampling method, target tracking.

I. INTRODUCTION

Over the past decades, particle filters have been applied with great success to a variety of state estimation problems [1]–[3], especially in nonlinear and non-Gaussian systems for which there is no analytical optimal solution [4], [5]. In simple words, particle filter (PF) is based on Sequential Monte Carlo approach [6]–[9], which utilizes a large number of samples (particles) to represent the posterior probability distributions. The samples are propagated over time using a combination of sequential importance sampling and resampling steps [10], [11]. These methods are very flexible and can be easily applied to nonlinear and non-Gaussian dynamic models.

As key performance assessment criteria for algorithms, computational complexity and the robustness of algorithm are two challenging problems for the application of particle filters. On the one hand, it is due to the sample-based approximation that particle filters combine efficiency with the ability to represent probability densities. When particle filter is running with a small sample set, the key problem is how to approximate the probability distribution function (PDF) properly so that the effectiveness and diversity of samples can be maintained to avoid the emergence of sample impoverishment [12]. On the other hand, although Monte Carlo method has a high inherent robustness, the convergence results of the state-of-the-art is hard to control and PF faces many critical problems, such as: degeneracy and sample impoverishment problem. Multiple studies have tried to address these challenges.

Firstly, efforts have been made to make more effective use of the available samples, thereby allocating the number of particles efficiently is required, with the purpose of reducing the computational burden on embedded processors by limiting the number of cycles. One of the most elegant methods for adjusting the number of particles is KLD-sampling approach originally developed by Fox, et al [13] and gracefully extended for the resampling method design [14], [15]. The key idea of KLD-sampling/resampling method is to determine the number of samples based on statistical bounds on the sample-based approximation quality [13]. Furthermore, Soto [16] presents a revised bound for KLD-sampling based on the variance of importance sampling. Fitzgerald [17] proposed the independent partitions method as an

The associate editor coordinating the review of this manuscript and approving it for publication was Sun-Yuan Hsieh.

advanced proposal scheme by reducing the number of particles needed for multiple target tracking while Pan [18] used rate-distortion theory to determine the optimal particle number.

Secondly, based on importance sampling and resampling, PF suffers from two critical non-robust problems: weight degeneracy and sample impoverishment [10]–[12]. Sample degeneracy is an unavoidable phenomenon in particle filter. Resampling can be adopted to solve this problem [10], [11], but it will lead to the loss of effectiveness and diversity of the samples, namely the sample impoverishment [12]. These works aim to make more effective use (increase robustness and accuracy) of the available samples (available computing resource). However, to our knowledge, most approaches are based only on the weights of samples regardless of samples' spatial distribution. This can result in that most computational resource is allocated to a small region of the most possible estimate [10]–[12], causing weight degeneracy or sample impoverishment. The only difference between degeneracy and sample impoverishment is whether the computational resource concentrating happens by samples (sample impoverishment) or by weights of samples (degeneracy) [11], [12]. To avoid this, we propose a novel methodology of particles' allocation referred as space sampling method, which samples and resamples not only based on the weights of samples but also their space distribution. What is more, following the idea of the KLD-resampling method [14], the sample size can also be adapted according to their space distribution to improve the real-time performance of PF. These two parts are implemented in double space-resampling. The contributions of the proposed method are summarized as follows.

1) The first resampling named Auxiliary Resampling is executed before the prediction step of the PF to improve the robustness of the filter. The samples size in the stage can be increased without increasing execution time heavily, since the time consumption of the prediction step is only a very small proportion in the PF.

2) The second space resampling named Merging Resampling is executed between the prediction and the updating step of the PF to decrease the weight updating computation and improve the efficiency of filtering, which is the most time consumption part of the PF.

3) A particle space sampling (PSS) technology, simple and fast for practical implementation, is used in double space-resampling.

The reminder of this paper is organized as follows. We begin in Section II an investigation of the PF, in which we excavate the inherent drawbacks of the PF. We give details of our solutions, the double space-resampling methodology and its theory study in Section III. This is the core contribution of this paper. Simulation and experiment results are presented in Section IV before we conclude in Section V.

II. BACKGROUND

The recursive Bayesian filter attempts to construct the posterior probability density function of the state based on all available information. In order to develop the details of the algorithm, let x_t denotes the state at time instant t. z_t is the perceptual data (observation) at time t, u_t is the odometry data (control measurement) between time t-1 and t, and $p(x_0)$ is the initial state. There are two basic stages of prediction and updating in typical Bayesian estimation framework. Assuming the environment is Markov [19], that is, past and future data are (conditionally) independent if one knows the current state, the prediction model can be described by

$$p(x_t\,|z_{t-1}\,, u_{t-1}) = p(x_t\,|x_{t-1}, u_{t-1}\,) \times p(x_{t-1}\,|z_{t-1}, u_{t-2}\,), \tag{1}$$

and the updating model is

$$p(x_t\,|z_t\,, u_{t-1}) = \frac{p(z_t\,|x_t\,)p(x_t\,|z_{t-1}\,, u_{t-1})}{p(z_t\,|z_{t-1}, u_{t-1}\,)}. \tag{2}$$

The PF represents the posterior by a set of random particles with associated weights and computes estimate based on these particles and weights. Let $S_t = \{x_t^i, w_t^i\}_{i=1,2,\ldots,Nt}$ denote a random measure that characterizes the posterior PDF $p(x_t|z_{1:t})$, where $\{x_t^i\}_{i=1,2,\ldots,Nt}$ is a set of particles with associated weights $\{w_t^i\}_{i=1,2,\ldots,Nt}$, and N_t is the total number of particles. The weights are normalized such that $\sum_i w_t^i = 1$. Then, the weighted posterior density at t can be written as

$$p(x_t\,|z_{1:t}\,) \approx \sum_{i=1}^{N_t} w_t^i \delta(x_t - x_t^i) \tag{3}$$

where the δ is Dirac delta measure. It can be proved that for $N \to \infty$ the approximation tends to the true posterior $p(x_t|z_{1:t})$. The samples are drawn from a known *importance density* $q(x_t^i\,|x_{0:t-1}^i, z_{0:t}\,)$ and the weights w_t^i are chosen according to sequential importance sampling (SIS), which relies on

$$w_t^i \propto w_{t-1}^i \frac{p(z_t\,|x_t^i)p(x_t^i\,|x_{t-1}^i\,)}{q(x_t^i\,|x_{0:t-1}^i, z_{0:t}\,)} \tag{4}$$

However, after a few iterations, most particles have negligible weight and the weight is concentrated on a few particles only. Resampling, eliminating particles that have small weights and concentrating on particles with large weights, is a very intuitive idea to counteract this problem [10]–[12]. But some side effects may arise at the meantime, such as sample impoverishment. That is very few different particles shave significant weight while some edged and isolated particles with small weight are discarded at the resampling process, which has much the same effect as sample degeneracy, and is even more severe in the case of small process noise. One of our objects in this paper is just to overcome the side effect of resampling. Intuitively, the sampling importance resampling (SIR) algorithm is also referred to sequential importance sampling and resampling (SISR), and can be depicted as Fig. 1.

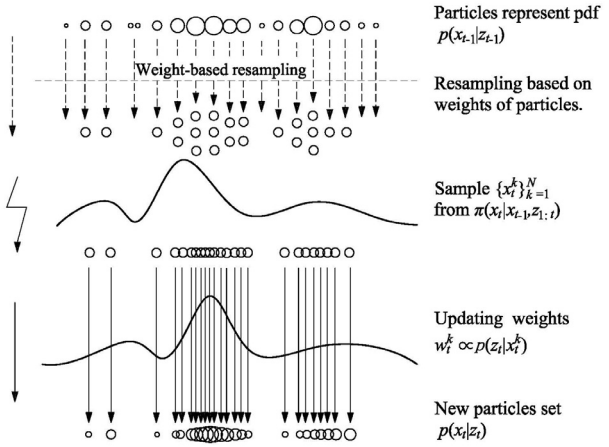

FIGURE 1. SIR based particle filter. The number of particles is fixed in the whole iteration. The arrows in the left of the figure indicate different particle propagation.

III. DOUBLE SPACE-RESAMPLING

Based on the investigation of the PF in the previous section, we try to improve the performance of the PF though twice space sampling operation with regard to the particles' spatial distribution. That is, on one hand, to find a compromise of the contradiction between the computing complexity and the robustness need: we use different number of particles at the prediction and updating step respectively. On the other hand, to develop a novel resampling method that allocates the computing resource without loosing the diversity of particles. This method should be easy to implement, and take into account the spatial similarity of particles.

In the following, we firstly decompose the state space into grid cells to study the spatial similarity and the diversity of particles. Then, the scheme of double space-resampling based on the discrete partition of the state space is described. Further, a theoretical analysis is given to illustrate the rationality of our approach.

A. DISCRETE STATE SPACE DIVISION

To facilitate the description, we make the following definition

Definition 1: The number of particles distributed in a grid is defined as particle density. If there is no particle in a grid, this grid is called an empty grid; otherwise it's a non-empty grid.

In this paper, we propose a double space-resampling based particle filter. To construct the required variable-precision grids, a "self-fission" method [20] is adopted, i.e., Algorithm 1 Particles division. In Algorithm 1, $x_t^p = (x, y, \theta)$ denotes the state of robot described as three-dimensional state space, where (x, y) represents the location in coordinate system and θ denotes the direction angle of the robot. Particle Division $\{g_{i,j,h}, L/2\}$ means that the particle density is superior to a threshold α, the grid will be divided again to half size. One chooses a starting grid size L_{star} to begin the particles division, and then the particle density of each nonempty grid is detected: if its particle density is superior to a threshold α, the grid will be divided again to half size. The detection and division are repeated until the particle density of all grids is under the threshold α or the grid size is lower to the lower bound L_{min}, which is used to avoid that grids are divided too small. In fact, $L_{\min}$ denotes a scalar, which represents the size of grids. In the following algorithms, L_{min} and α are global constant variables. The $(i, j, h)_{\text{th}}$ grid cell $g_{i,j,h}$ with $N_{i,j,h}$ particles in it is described as

$$g_{i,j,h} = \{(x_{i,j,h}^k, w_{i,j,h}^k \,|k = 1, 2, \ldots, N_{i,j,h})\}. \tag{5}$$

B. FIRST RESAMPLING: AUXILIARY RESAMPLING

The immediate reason for loss of diversity in the traditional only-weight-based resampling procedures (refer to Fig. 1) is that some low-weight but important in spatial distribution particles may be discarded. To avoid this situation, we reintroduce the estimates of the nonempty grids from which no particle is sampled, by keeping alternatively the sample with the greatest weight as

$$g_{i,j,h} \Rightarrow (\hat{x}_t^n, \hat{w}_t^n) : \begin{cases} \hat{w}_t^n = \max\{w_{i,j,h}^k\}_{k=1}^{N_{i,j,h}} \\ (\hat{x}_t^n, \hat{w}_t^n) \in g_{i,j,h} \end{cases} \tag{6}$$

or their mean as

$$g_{i,j,h} \Rightarrow (\hat{x}_t^n, \hat{w}_t^n) : \begin{cases} \hat{x}_t^n = \sum_{k=1}^{N_{i,j,h}} x_{i,j,h}^k w_{i,j,h}^k / \hat{w}_t^n \\ \hat{w}_t^n = \sum_{k=1}^{N_{i,j,h}} w_{i,j,h}^k \end{cases} \tag{7}$$

where the new particle $(\hat{x}_t^n, \hat{w}_t^n)$ is called auxiliary particle, Symbol "$\Rightarrow$"means generated by sampling. In our approach, we prefer using (7), which is named as particle space sampling (PSS) since it sums up all the particles with their weights according to the particles' spatial distribution.

Appended with the auxiliary operation using equations (6) or (7) to guarantee that there is at least one particle survive for each nonempty grid, the improved resampling scheme, named auxiliary resampling in this paper, is detailed in Algorithm 2. In Algorithm 2, $G_t = G_t - g_{i,j,h} \leftarrow x_t^n \in g_{i,j,h}$ denotes that the i-th particle with lower weight is removed in multinomial resampling. It should be pointed out that the multinomial resampling [10] is just what we used in the auxiliary resampling scheme, but not a necessity to be restricted to. In fact, one may use any other resampling method as given in [10], [11].

Actually, particle filters have the problem of "dimension disaster", that is, the number of particles required by particle filter increases exponentially with the increase of state dimension, which makes the calculation of filtering increase rapidly. Therefore, it is difficult for particle filter to deal with the high dimension systems. In fact, a N-dimension space can be described as $S = A_1 \times A_2 \times \cdots \times A_N$. If the i-th dimension of S is divided into m_i intervals of equal length. The whole space S can be divided into $m_1 \times m_2 \times \cdots \times m_N$ disjoint spaces grids.

Algorithm 1 Particle Division

Input:
$S_t = \{(x_t^i, w_t^i)\}_{i=1}^{N_t}$, particle set
L, grid size
Output:
$G_t = \{g_{i,j,h} \,|i, j, h \in \mathbb{N}\}$, particles' division in grids
Procedure:
$G_t = \{\}$
$\forall i, j, h$: **set** $g_{i,j,h} = \{\}$, $d_{i,j,h} = 0$
$\forall x_t^p = (x, y, \theta)$: **for** $p = 1: N_t$ **do**
$(x, y, \theta) \xrightarrow{L} (i, j, h)//x_t^p$ fall into the grid $g_{i,j,h}$
$d_{i,j,h} = d_{i,j,h} + 1$
$g_{i,j,h} = g_{i,j,h} \cup (x_t^p, w_t^p)$
end for
$\forall$ *i, j, h*: **if** $d_{i,j,h} \geq \alpha$ and $L \geq L_{min}$ **do**
$G_t = G_t \cup$ *Particle Division* $\{g_{i,j,h}, L/2\}$
else
$G_t = G_t \cup g_{i,j,h}$
end if

C. SECOND RESAMPLING: MERGING RESAMPLING

It should be kept in mind that it's unwise to pursue only a little more accuracy at the price of unbearable computational cost, and therefore it is not simply 'the more particles, the better'. To some degrees, we can assume adjacent particles represent the same state if the distance between them is much less than the sensor reading error and the filtering accuracy requirement. They can use the same likelihood $p(z_t|x_t)$, which will reduce the computation of weight updating greatly. Thus, the second resampling with the aim to reduce the number of particles is implemented in a grid with its size below a threshold b by merging particles via (7). The interpretation of the threshold b is that there is enough reliability for the assumption hold. The second resampling is named merging resampling, which can be illustrated as Algorithm 3.

D. DOUBLE SPACE-RESAMPLING PARTICLE FILTER

Double space-resampling operations have been described in the last few sections, leaving an important issue to decide when should these two resampling be excused. According to the previous knowledge of the computational complexity and robustness of the PF, it's easy to understand why we execute the auxiliary resampling at the beginning of the iteration and the merging resampling between the prediction and the updating step (refer to Fig. 2). As another distinguishing feature of the auxiliary resampling, we set a large sample size (N in the Algorithm 2) for better robustness of the prediction step without increasing the total time-cost of the PF as the second resampling can reduce the sample size adaptively.

Essentially, the double space-resampling approach is an improvement of SIR algorithm and is correspondingly named SIRR (sampling importance double space-resampling) in this paper, like the structure of [21]. The difference is,

Algorithm 2 Auxiliary Resampling

Input:
$S_t = \{(x_t^i, w_t^i)\}_{i=1}^{N_t}$, raw particle set
N, Basic sample size
L, grid size
Output:
$\hat{S}_t = \{(\hat{x}_t^k, \hat{w}_t^k)\}_{k=1}^{\hat{N}_t}$, resampling particle set
Procedure:
$\hat{S}_t = \{\}$
1) State space division
$G_t = \{g_{i,j,h} \,|i, j, h \in \mathbb{N}\} =$ *Particle Division* $\{S_t, L\}$
2) Basic multinomial resampling
for $m = 1: N$ **do**
$u =$ rand //uniform random number $\in$ (0,1)
$q = 0$
for $n = 1: N_t$ **do**
$q = q + w_t^n$
if $q >= u$ **do**
$\hat{x}_t^m = x_t^n$
$\hat{S}_t = \hat{S}_t \cup (\hat{x}_t^m, \frac{1}{N})$
$G_t = G_t$ - $g_{i,j,h} \leftarrow x_t^n \in g_{i,j,h}$
break
end if
end for
end for
3) Sampling auxiliary particles
$p = 0$
$\forall g_{i,j,h} \in G_t$: **while** $G_t \neq \{\}$ **do**
$p = p + 1$
$g_{i,j,h} \Rightarrow (\hat{x}_t^p, \hat{w}_t^p)$ via (6) or (7)
$\hat{S}_t = \hat{S}_t \cup (\hat{x}_t^p, \hat{w}_t^p)$
$G_t = G_t$ - $g_{i,j,h}$
end while
4) Normalize the weights
$\hat{N}_t = N + p$
for $k = 1: \hat{N}_t$
$\hat{w}_t^k = \hat{w}_t^k [\sum \hat{w}_t^k]^{-1}$
end for

the "importance" in SIRR is not limited to weights but also particles' spatial distribution. SIRR has distinguished feathers over SIR at: edged and isolated particles are retained as auxiliary particles at the auxiliary resampling process, and particles are merged according to their spatial distribution at the merging resampling process.

To summarize the SIRR algorithm: we choose a large sample size for the prediction step and supplement auxiliary particles to improve the robustness and the accuracy of the estimation in the first auxiliary resampling operation. Then the number of particles is reduced in the second merging resampling operation which is adopted between the prediction and the updating steps. On the one hand, the double space-resampling method tries to allocate the limitedly computational resources to appropriate requirements. On the

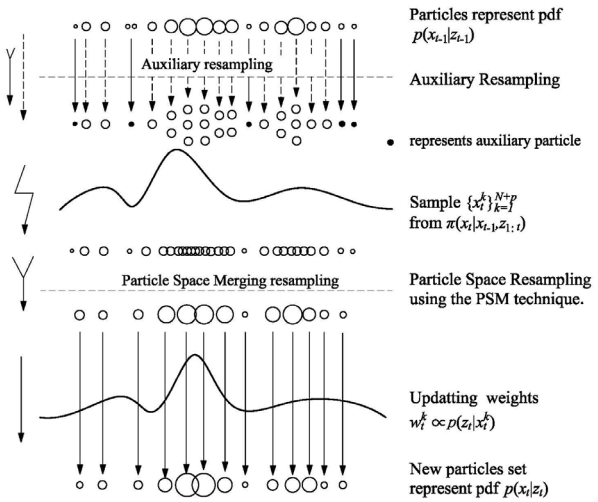

FIGURE 2. SIRR based particle filter.

other hand, the particle space sampling technique used in it is simple and fast to implement with a computational complexity $O(N)$, where N is the number of particles and there is no complicated computing and analysis.

Algorithm 3 Merging Resampling

Input:

$S_t = \{(x_t^i, w_t^i)\}_{i=1}^{N_t}$, raw particles

L, grid size

Output:

$\tilde{S}_t = \{(\tilde{x}_t^i, \tilde{w}_t^i)\}_{i=1}^{\tilde{N}_t}$, resampling particles

Procedure:

$\tilde{S}_t = \{\}$

$G_t = \{g_{i,j,h} \,|i, j, h \in \mathbb{N}\} = Particle\ Division\ \{S_t, L\}$

$\forall g_{i,j,h} \in G_t$: **while** $G_t \neq \{\}$ **do**

$g_{i,j,h} \Rightarrow (\tilde{x}_t^k, \tilde{w}_t^k)$ via (10)

$\tilde{S}_t = \tilde{S}_t \cup (\tilde{x}_t^k, \tilde{w}_t^k)$

$G_t = G_t$ - $g_{i,j,h}$

end for

E. BIAS ANALYSIS

Proposition 1: PSS using merging formula (7) does not change the mean of the particles' distribution but reduce the variance.

Proof: Supposing all the particles are divided into K nonempty grid cells using (6), the mean of the particles' distribution can be easily obtained by

$$e = E(g_{i,j}) = \sum_{i,j}^{K} \sum_{k=1}^{N_{i,j}} x_{i,j}^k w_{i,j}^k \tag{8}$$

and the variance is

$$\delta = \sum_{i,j}^{K} \sum_{k=1}^{N_{i,j}} (x_{i,j}^k - e)^2 w_{i,j}^k \tag{9}$$

If particles are merged by (7), the mean of particles will be not changed as

$$e' = \sum_{k=1}^{K} x_t^k w_t^k = \sum_{i,j}^{K} \sum_{k=1}^{N_{i,j}} x_{i,j}^k w_{i,j}^k = e \tag{10}$$

but the variance is reduced

$$\begin{aligned}
\delta' &= \sum_{k=1}^{K} (x_t^k - e')^2 w_t^k \\
&= \sum_{i,j}^{K} (\sum_{k=1}^{N_{i,j}} x_{i,j}^k w_{i,j}^k / \hat{w}_t^n - e)^2 \hat{w}_t^n \\
&= \sum_{i,j}^{K} [(\sum_{k=1}^{N_{i,j}} x_{i,j}^k w_{i,j}^k / \hat{w}_t^n)^2 - 2e(\sum_{k=1}^{N_{i,j}} x_{i,j}^k w_{i,j}^k / \hat{w}_t^n) + e^2] \hat{w}_t^n \\
&\leq \sum_{i,j}^{K} \sum_{k=1}^{N_{i,j}} (x_{i,j}^k - e)^2 w_{i,j}^k = \delta
\end{aligned} \tag{11}$$

The inequality in (11) comes from Cauchy-Schwarz inequality. The equality of the inequality is satisfied only when grids are divided so small that there is no more than one particle in each grid. From (10) and (11), one can see that the sample mean value of original particle distribution and merging particle distribution is the same, but the sample variance of the resampled particle is smaller. Therefore, the merging particles are unbiased consistent estimation with the original particle distribution.

IV. SIMULATIONS AND EXPERIMENTS

A. COMPARISON OF PARTICLE FILTERS

For the sake of evaluating and comparing the effectiveness of particle filters, we study the following nonlinear time-varying framework with a prediction function on a standard PC with a Core (TM) 2 CPU, a 2.93 GHz processor and a 2.0 G RAM.

$$x_t = \frac{x_{t-1}}{2} + \frac{25x_{t-1}}{1 + x_{t-1}^2} + 8\cos(1.2t) + e_t \tag{12}$$

and an updating function

$$o_t = \frac{x_t^2}{20} + v_t \tag{13}$$

where e_t and v_t are zero mean Gaussian random variables with variance 10 and 1, respectively. The system is highly nonlinear, and its likelihood function is bimodal, which requires high performance of the filtering method. Therefore, the nonlinear system is widely used to verify the effectiveness of particle filter method. We use root mean square error (RMSE) to evaluate the estimate accuracy as usual, which is calculated by

$$\text{RMSE} = \sqrt{\frac{\sum_{t=1}^{T} (x_t - \widehat{x}_t)^2}{T}}. \tag{14}$$

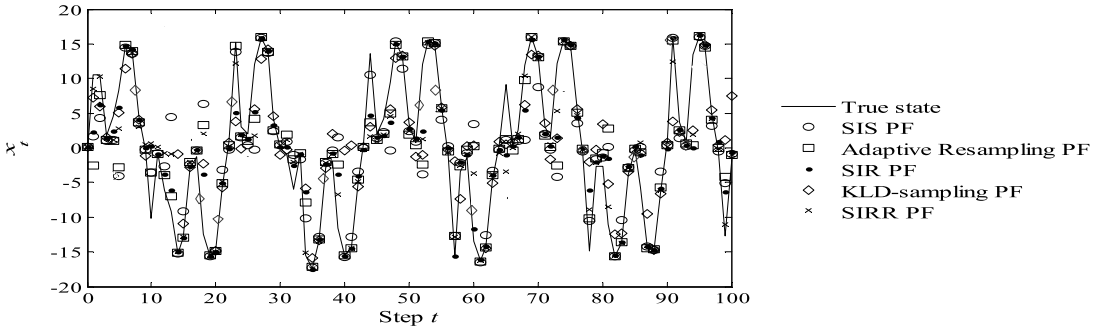

FIGURE 3. Simulation results of particle filters. All particle filters work in the models.

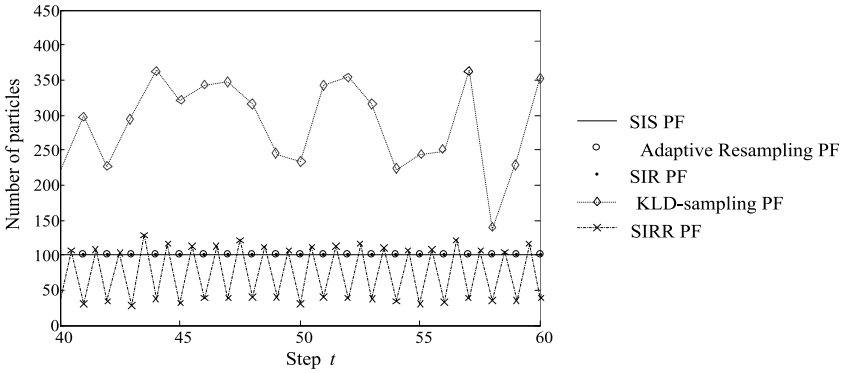

FIGURE 4. Number of particles in filtering. The number of particles is fixed on the desired level in SIS, adaptive resampling and SIR particle filtering, and changes to the need level calculated by KLD in the KLD-sampling approach, which happens twice at each iteration in our approach.

where T is the sum of iterations, in time steps, and we choose $T = 1000$ in our instance.

In our instance, four other particle filters are compared with our SIRR-based PF. Sorted by the times of resampling at each iteration, they are basic SIS PF (none, no resampling), adaptive-resampling PF (resample or not), SIR PF (once) and SIRR PF (twice). In the following, we will firstly give a brief introduction of the adaptive resampling and KLD-sampling approaches with their parameter settings.

Adaptive-resampling particle filter means to resample only when the variance of the non-normalised weights is superior to a pre-specified threshold, which is often assessed by the variability of the weights using the so-called Effective Sample Size (ESS) criterion [22], which is given by

$$ESS = (\sum_{i=1}^{N_t} (w_t^i)^2)^{-1}. \tag{15}$$

The ESS takes values between 1 and N_t and the resampling is implemented only when it is below a threshold N_T, typically $N_T = N_t/2$ in our instance.

For KLD-sampling/resampling particle filter, the number of particles needed is calculated briefly by

$$N_t = \frac{k-1}{2\varepsilon}(1 - \frac{2}{9(k-1)} + \sqrt{\frac{2}{9(k-1)}} z_{1-\delta})^3 \tag{16}$$

where k is the number of bins with support (for details see [13], [14]). $z_{1-\delta}$ is the upper quantile of the standard normal distribution. In our approach, (1-δ) is fixed to 95%, $\varepsilon = 0.05$, and the bin size is 1.

The true state x_t is one-dimension, so our double space-resampling will be implemented in one-dimension grids with variable-precision sizes. We choose the starting grids size $L_{star} = 1$, the smallest threshold $L_{min} = 0.02$, and the upper value of particle density of grids $\alpha = 8$.

Fig. 3 and Table 1 present the simulation results and Fig. 4 shows the volatility of sample size in filtering. From Fig. 4, it can be seen that the number of particles is 100 and is fixed in basic SIS, adaptive-resampling and SIR particle filter, but adaptively changes in our approach and the KLD-sampling PF. Furthermore, the sample size in our approach is consistent with the size of grid cells and it can reduce the number of particles more efficiently than the KLD-sampling approach. From Fig. 3, one can observe that, the RMSE of SIS-PF, Adaptive resampling PF, SIR-PF, KLD-sampling PF, and our method are 5.8613, 4.9769, 5.2739, 5.0265, and 4.43, respectively, which means that the estimate accuracy of our approach is improved compared to the other PF methods. In fact, the advantage of the proposed method is that it can adjust the sample size according to not only the weights of samples but also their spatial distribution dynamically. Therefore, the filtering accuracy can be improved. Moreover, the estimation accuracy is related to meshing parameters

TABLE 1. Particle filters performance reference table.

Particle filters	Average number of particles for prediction	Average number of particles for updating	RMSE	Running time/s (1000 iterations)
SIS-PF (N=100)	100	100	5.8613	0.2006
Adaptive resampling PF (N =100)	100	100	4.9769	2.6913
SIR-PF (N=100)	100	100	5.2739	3.7051
KLD-sampling PF (1-δ= 95% ε=0.05 the bin size is 1)	286.43	286.43	5.0265	7.4806
Double space-resampling PF a=8	111.71	34.56	4.43	7.089

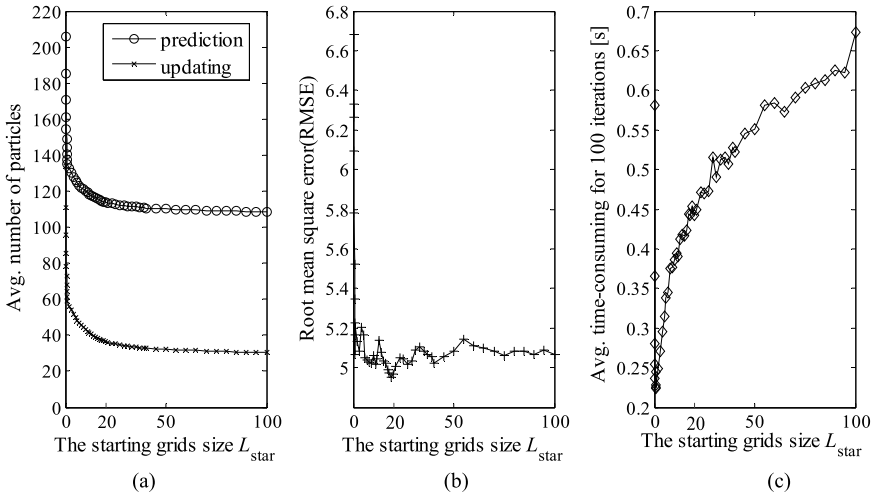

FIGURE 5. The influence of starting grid size L_{star} on (a) the number of particles, (b) root mean square error and (c) time consumption for 100 iterations with fixed $L_{min} = 0.05$ and $a = 8$.

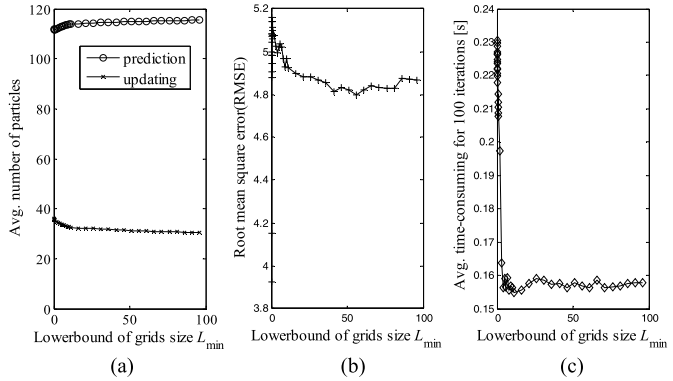

FIGURE 6. The influence of lower bound of grid size L_{min} on (a) the number of particles, (b) root mean square error and (c) time consumption for 100 iterations with fixed $L_{star} = 1$ and $a = 8$.

L_{star}, $L_{\min}$, and the upper value of particles density α. However, the running time of our approach is 7.089s, which means the computing efficiency of our approach is the worst. This result is not surprising, since the updating step is less time consumption than the prediction step. Thus, our approach, trying to improve the computing efficiency by reducing the times of updating computation, only works when the resampling time is much less than the weight updating computation of particles, and such is usually the case of PF. As we have analysed in Section II. B, the filtering updating step is much more time consumption than the simple movement prediction step and then our method will ''work''. This will be illustrated in our following experiments in Section IV. C.

B. PARAMETER SETTINGS

For a further view of the feature of our double space-resampling approach, the influences of different parameter settings, i.e., L_{star}, L_{min}, and α, on the performance of our double space-resampling approach are discussed in this section. We keep two parameters fixed and one variable to test the change of average number of particles at the prediction step and the updating step respectively, the estimation RMSE and the time consumption for 100 iterations. The results are given in Fig. 5 for the starting grid size L_{star}, Fig.6 for the lower bound of grid size L_{min} and Fig. 7 for the upper value of particles density α.

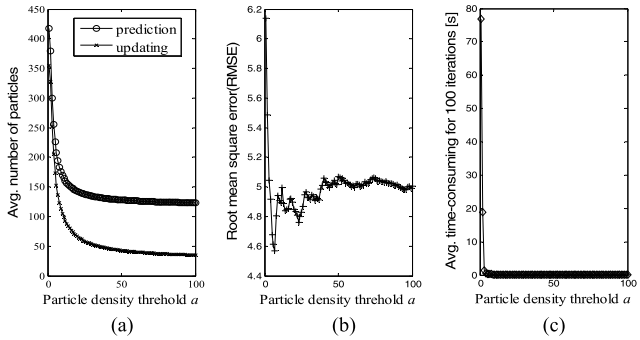

FIGURE 7. The influence of upper value of particle density a on (a) the number of particles, (b) root mean square error and (c) time consumption for 100 iterations with fixed $L_{star} = 1$ and $L_{min} = 0.05$.

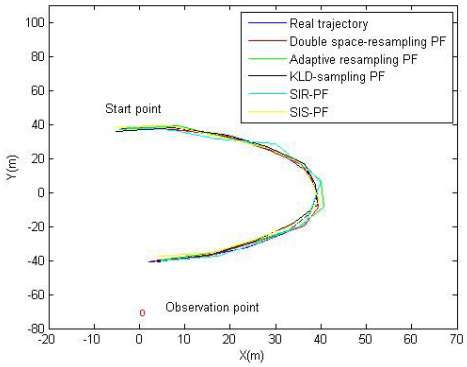

FIGURE 8. The target tracking comparison result.

1) From Fig.5, one can see that the number of particles both at the prediction step and the updating step and the estimation RMSE decrease, but the time consumption increases with the growth of the starting grid size L_{star}, significantly when L_{star} is smaller than 1. The number of particles and the estimation RMSE change negligibly when L_{star} is greater than 20 or about while the time consumption continuously increases under a level of 0.7 second for 100 iterations.

2) The number of particles increases at the prediction step but decreases at the updating step insignificantly with the growth of the lower bound of grid size L_{min}, as shown in Fig. 6. The estimation RMSE increases very significantly when L_{min} is smaller than 0.1 (there are some values between 0.01 and 0.1) and then reduces with the growth of L_{min} until it's over 40. By contrast, the time consumption reduces greatly when L_{min} is smaller than 3 and to a certain level with the growth of L_{min}.

3) From Fig. 7, one can observe that the number of particles, the estimation RMSE and the average time consumption reduce with the growth of the particle density threshold α, especially significantly when α is smaller than 7 or about. And they tend to be stable when α is greater than 10 or about. In particular, the average time consumption for 100 iterations is unbearable large when α is 1 and 2, which indicates the particle division will cost a lot of time when the division is too fine.

Thus far, we can see there are some contradictions between the RMSE and the time consumption when the parameters are varied. It's difficult to find the best balance (if there is) of parameter settings. Fortunately, the parameters can be chosen considering the sensor noise and the filter accuracy requirement via (7). Note that L_{star} is greater than L_{min} and α should be larger than 3 at least in practice.

TABLE 2. Particle filters performance reference table.

Particle filters	RMSE	Running time/s
SIS-PF	5.4986	0.8172
Adaptive resampling PF	4.2963	3.8423
SIR-PF	4.9852	4.4615
KLD-sampling PF	4.3694	4.9312
Double space-resampling PF	3.7566	4.5234

C. TARGET TRACKING EXAMPLE

In this section, the proposed double space-resampling PF is evaluated using a robot target tracking example. The target moves an elliptical motion in the plane. The initial position of the target and the position of the observation point are (0, 37.7) and (0, -70), respectively. We choose the starting grids size $L_{star} = 1.5$, the smallest threshold $L_{min} = 0.03$, and the upper value of particle density of grids $\alpha = 7$. The simulation result is shown as Fig. 8. From the result, one can see that the proposed method can track the real trajectory of the target. The proposed double space-resampling PF is effectiveness.

V. CONCLUSION

In this paper, the problems for sample impoverishment and computational efficiency of PF methods are solved. A double space-resampling method based particle filters is presented with respect to the practical features of robot localization. The particle space sampling technique is used in double space-resampling, which adjusts the sample size according to not only the weights of samples but also spatial distribution. The first space resampling is introduced to improve the robustness of particle filters and the second resampling is introduced to reduce its time consumption. Simulation results are given to demonstrate that the estimation accuracy and robustness of the proposed approach, which is improved compared to traditional PF methods.

REFERENCES

[1] D. Crisan and K. Li, "Generalised particle filters with Gaussian mixtures," *Stochastic Processes Appl.*, vol. 125, no. 7, pp. 2643–2673, 2015.
[2] X. Wang, T. Li, S. Sun, and J. Corchado, "A survey of recent advances in particle filters and remaining challenges for multitarget tracking," *Sensors*, vol. 17, no. 12, p. 2707, Nov. 2017.
[3] "A tutorial on particle filtering and smoothing: Fifteen years late," in *Handbook of Nonlinear Filtering*. Cambridge, U.K.: Cambridge Univ. Press, 2009.
[4] T. Schon, F. Gustafsson, and P.-J. Nordlund, "Marginalized particle filters for mixed linear/nonlinear state-space models," *IEEE Trans. Signal Process.*, vol. 53, no. 7, pp. 2279–2289, Jul. 2005.
[5] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Trans. Signal Process.*, vol. 50, no. 2, pp. 174–188, Aug. 2002.
[6] M. Khaki, M. S. Filmer, W. E. Featherstone, M. Kuhn, L. K. Bui, and A. L. Parker, "A sequential Monte Carlo framework for noise filtering in InSAR time series," *IEEE Trans. Geosci. Remote Sens.*, vol. 8, no. 3, pp. 1904–1912, Nov. 2020.

[7] W. Li, R. Chen, and Z. Tan, "Efficient sequential Monte Carlo with multiple proposals and control variates," *J. Amer. Statal Assoc.*, vol. 111, pp. 1–44, Jun. 2015.
[8] N. Chopin, "Central limit theorem for sequential Monte Carlo methods and its application to Bayesian inference," *Ann. Statist.*, vol. 32, no. 6, pp. 2385–2411, Dec. 2004.
[9] J. Olsson, O. Cappé, R. Douc, and É. Moulines, "Sequential Monte Carlo smoothing with application to parameter estimation in nonlinear state space models," *Bernoulli*, vol. 14, no. 1, pp. 155–179, Feb. 2008.
[10] T. Li, M. Bolic, and P. M. Djuric, "Resampling methods for particle filtering: Classification, implementation, and strategies," *IEEE Signal Process. Mag.*, vol. 32, no. 3, pp. 70–86, May 2015.
[11] T.-C. Li, G. Villarrubia, S.-D. Sun, J. M. Corchado, and J. Bajo, "Resampling methods for particle filtering: Identical distribution, a new method, and comparable study," *Frontiers Inf. Technol. Electron. Eng.*, vol. 16, no. 11, pp. 969–984, Nov. 2015.
[12] T. Li, S. Sun, T. P. Sattar, and J. M. Corchado, "Fight sample degeneracy and impoverishment in particle filters: A review of intelligent approaches," *Expert Syst. Appl.*, vol. 41, no. 8, pp. 3944–3954, Jun. 2014.
[13] D. Fox, "Adapting the sample size in particle filters through KLD-sampling," *Int. J. Robot. Res.*, vol. 22, no. 12, pp. 985–1003, Dec. 2003.
[14] T. Li, S. Sun, and T. P. Sattar, "Adapting sample size in particle filters through KLD-resampling," *Electron. Lett.*, vol. 49, no. 12, pp. 740–742, Jun. 2013.
[15] W. Xu, R. Jiang, L. Xie, X. Tian, and Y. Chen, "Adaptive square-root transformed unscented FastSLAM with KLD-resampling," *Int. J. Syst. Sci.*, vol. 48, no. 6, pp. 1322–1330, Apr. 2017.
[16] A. Soto, "Self adaptive particle filter," in *Proc. Int. Joint Conf. Artif. Intell.*, 2005, pp. 1398–1406.
[17] M. Orton and W. Fitzgerald, "A Bayesian approach to tracking multiple targets using sensor arrays and particle filters," *IEEE Trans. Signal Process.*, vol. 50, no. 2, pp. 216–223, Feb. 2002.
[18] P. Pan and D. Schonfeld, "Dynamic proposal variance and optimal particle allocation in particle filtering for video tracking," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 9, pp. 1268–1279, Sep. 2008.
[19] J. R. Norris, *Markov Chains*. Cambridge, U.K.: Cambridge Univ. Press, 2012.
[20] T. Li, T. P. Sattar, and S. Sun, "Deterministic resampling: Unbiased sampling to avoid sample impoverishment in particle filters," *Signal Process.*, vol. 92, no. 7, pp. 1637–1645, Jul. 2012.
[21] T.-C. Li and S.-D. Sun, "Double-resampling based Monte Carlo localization for mobile robot," *Acta Automatica Sinica*, vol. 36, no. 9, pp. 1279–1286, Dec. 2010.
[22] P. D. Moral, *Feynman-Kac Formulae: Genealogical and Interacting Particle Systems With Applications* (Series: Probability and Applications). New York, NY, USA: Springer, 2005.

ZHENG GONG received the B.S. and M.S. degrees from the School of Astronautics, Northwestern Polytechnical University, in 2004 and 2007, respectively, where he is currently pursuing the Ph.D. degree. His research interests include guidance and control for air-to-air missiles.

GANG GAO received the Ph.D. degree from the School of Engineering, Peking University, in 2015. He is currently working as an Engineer with the Luoyang Optoelectro Technology Development Center. His research interest includes flight control and guidance.

MINGANG WANG was born in 1958. He received the Ph.D. degree from the School of Astronautics, Northwestern Polytechnical University, Xi'an, China. He is currently working as a Professor and the Ph.D. Candidate Supervisor with Northwestern Polytechnical University. His research interests include guidance and control, computer control and simulation, and image processing.