Journal of Computational Information Systems 7: 6 (2011) 1838-1845
Available at http://www.Jofcis.com

Parallel Resampling Particle Filter Algorithm

Jun BI[1,†], Yufai FUNG[2], Tinkin HO[2], Baohua MAO[1]

[1] *School of Traffic and Transportation, Beijing Jiaotong University, Beijing 100044, China*
[2] *Department of Electrical Engineering, The Hong Kong Polytechnic University*

Abstract

Resampling in the particle filter algorithm can solve the algorithm's degeneracy problem. In order to decrease the execution time of the particle filter, the parallel resampling particle filter algorithm is proposed. In the algorithm, firstly all weights of the particles are sorted according to the ascending order. Secondly the particles space is classified into two independent sets. Finally the particles that will be resampled from two sets respectively are found parallelly according to the random search method. According to the theoretical analysis and the experiment results, the algorithm can reduce the search space for resampling and can shorten the search time, so it has high efficiency in the implementation. What is more, the algorithm can overcome the blindness of resampling, and can better embody the basic idea of resampling which is a good weight particle to be reproduced more, so it has better filter and estimation performance.

Keywords: Particle Filter; Resampling; Particle Degeneracy; Parallel Combination

1. Introduction

Particle filter (PF) is a promising solution to the general nonlinear and /or non-Gaussian filtering problem [1-3]. The central idea of PF is to represent the posterior probability density functions (pdf) of state by a set of particles with associated weights, and the estimate can be computed as the expected value of the discrete pdf. The resampling step is critical in every implementation of PF because without it, the variance of the particle weights quickly increases, i.e., very few normalized weights are substantial. Then, the inference is degraded because it is made by using only a very small number of particles. The idea of resampling is to remove the particle trajectories with small weights and replicate the trajectories with large weights. Resampling was proposed for use in particle filtering in various works including [4-7].

However, the application of PF in real-time systems is limited due to their inherent computational complexity[8-10]. The main design goal of this paper is to shorten the execution time of the PF. This is done through exploiting a parallel resampling method. We show that the parallel resampling method can greatly increase the execution speed of PF, which is suitable for the real-time embedded computer systems.

This paper is organized as follows. We provide a brief overview of PF theory in Section II. In Section III, we propose a parallel resampling method in PF. Section IV presents some simulation results, and Section V is a summary of the work in this paper.

2. Introduction of PF Theory

According to the following discrete-time stochastic model of the dynamic system

$$x_k = f_k(x_{k-1}, w_{k-1}) \tag{1}$$

$$y_k = h_k(x_k, v_k) \tag{2}$$

[†] Corresponding author.
Email address: bilinghc@163.com (Jun BI)

Where x_k represents the state vector at discrete time k . y_k represents the observation vector at discrete time k . f_k is nonlinear state transition function and h_k is nonlinear observation function. The stochastic processes w_k and v_k represent the state and measurement noise processes. It is assumed that the initial probability density function of the state vector $p(x_0 \mid D_0) = p(x_0)$ is available. The available information at time step k is the set of measurements $D_k = \{y_i : i = 1, 2, ..., k\}$ .According to the Bayesian estimation theory,

$$p(x_k \mid D_{k-1}) = \int p(x_k \mid x_{k-1}) p(x_{k-1} \mid D_{k-1}) dx_{k-1} \tag{3}$$

$$p(x_k \mid D_k) = \frac{p(y_k \mid x_k) p(x_k \mid D_{k-1})}{p(y_k \mid D_{k-1})} \tag{4}$$

$$p(y_k \mid D_{k-1}) = \int p(y_k \mid x_k) p(x_k \mid D_{k-1}) dx_k \tag{5}$$

Because it is very difficult to solve the integration in the equation(3), PF uses the Monte Carlo techniques and Sequence Importance Sampling(SIS)methods to solve the problem. PF estimates the probability density function of state according to the number N samples from the important distribution function, that is

$$p(x_k \mid D_k) = \sum_{i=1}^{N} w_k^i \delta(x_k - x_k^i) \tag{6}$$

$$w_k^i = \frac{\overline{w}_k^i}{\sum_{i=1}^{N} \overline{w}_k^i} \tag{7}$$

$$\overline{w}_k^i = \overline{w}_{k-1}^i \frac{p(y_k \mid x_k^i) p(x_k^i \mid x_{k-1}^i)}{q(x_k^i \mid x_{k-1}^i, y_k)} \tag{8}$$

Where w_k^i is the weight of every particle and w_k^i satisfies the relation of $0 \le w_k^i \le 1$. q is important distribution function. To get good estimation, the important distribution function should be close to the real posterior distribution of state. But in order to simplify the computation of PF, define the important distribution function,

$$q(x_k \mid x_{k-1}^i, z_k) = p(x_k \mid x_{k-1}^i) . \tag{9}$$

So PF is also called Bootstrap filter.

3. Parallel Resampling PF

3.1. Algorithm Description

3.1.1. Initialization

When the time k is zero, suppose that samples $\{x_0^i,\ i = 1, 2, ..., N\}$ are available from a density function $p(x_0)$. Then set k =1.

3.1.2. SIS

Set $q(x_k \mid x_{k-1}^i, z_k) = p(x_k \mid x_{k-1}^i)$, and $\{x_k^i, i = 1, 2, ..., N\}$ are from a density function $p(x_k \mid x_{k-1}^i)$ .According to the equation (8), the important weight is,

$$\overline{w}_k^i = \overline{w}_{k-1}^i p(y_k \mid x_k^i) . \tag{10}$$

According to the equation (7), the normalized weight w_k^i of each particle is got.

3.1.3. Parallel Resampling

Step1. Sort N particles' weights according to the ascending order, that is, $w_k^1 = \min(w_k^i \mid_{i=1,2,\ldots,N})$, $w_k^N = \max(w_k^i \mid_{i=1,2,\ldots,N})$. At the same time, sort the x_k^i according to the corresponding weight, stilly the weight of x_k^i is w_k^i. Save N particles' weights into the set W.

Step2. The set W is divided into two sets S_1 and S_2, S_1 is composed of every weight w_k^{2i-1}, $i = 1, 2, \ldots,$ and $2i-1 \le N$. S_2 is composed of every weight w_k^{2i}, $i = 1, 2, \ldots,$ and $2i \le N$. There are k_1 elements in S_1 and k_2 elements in S_2

Step3. Set $c_1 = \sum w_k^i,\ w_k^i \in S_1$, $c_2 = \sum w_k^i,\ w_k^i \in S_2$, and $0 < c_1 < 1, 0 < c_2 < 1, c_1 + c_2 = 1$. Set $t = \max(sum_1, sum_2)$, $d = \min(sum_1, sum_2)$, $a = t / d$ is a coefficient.

Step4. Assume that n_1 is the number of copied particles in the set with minimum sum of weights. $n_1 = \text{int}(\frac{N}{1+a})$.Here int represents round numbers operation. Assume that n_2 is the number of copied particles in the set with maximum sum of weights.

Step5. Search from S_1 and S_2 sets respectively at the same time to get particle $\overset{\sim i}{x_k}$ through resampling. The detailed methods are shown in step 5-1 and step 5-2.

Step5-1. Search from S1 to get particle $\overset{\sim i}{x_k}$ through resampling as follows

```
For $i = 1 : n_1$
    Drawing a random sample $u_i$ from the uniform distribution over $(0, c_1]$
    Set variable $j$ =1
    While $j \le k_1$
   {If $\sum_{j=1}^{m-1} w_k^j < u_i \le \sum_{j=1}^{m} w_k^j,\ w_k^j \in S_1$,
      {The number $m$ particle is resampled,
          $\overset{\sim i}{x_k} = x_k^m$
          $\overset{\sim i}{w_k} = w_k^m$
        Break
        }
    Else $j = j + 1$
    }
    End While
End For
```

Step5-2. Search from S2 to get particle $\overset{\sim i}{x_k}$ through resampling as follows

```
For $i = 1 : n_1$
    Draw a random sample $u_i$ from the uniform distribution over $(0, c_1]$
    Set variable $j$ =1
    While $j \le k_1$
    {If $\sum_{j=1}^{m-1} w_k^j < u_i \le \sum_{j=1}^{m} w_k^j$, $w_k^j \in S_1$,
        {The number $m$ particle is resampled,
                $\tilde{x}_k^i = x_k^m$
                $\tilde{w}_k^i = w_k^m$
         Break
        }
    Else $j = j + 1$
    }
    End While
End For
```

3.1.4. State Estimation

Set k to be increased itself by one. If k is less than its set value, the algorithm is end, else return to step 2.

$$\hat{x}_k = \sum_{i=1}^{N} \tilde{w}_k^i \tilde{x}_k^i$$

3.2. Algorithm Analysis

(1) From the viewpoint of execution time, the whole resampling process is divided into two small resampling tasks which can be executed at the same time, so the algorithm can greatly shorten search time and has good execution time efficiency.

(2) From viewpoint of space, the scope of whole particle space is N. After the whole space is divided into two independent set spaces, the scope of each set space is $N/2$ .Because each resampling task can be excuted parallelly in each set, so it has small search space.

(3)Parallel resampling algorithm is fit for the multithreading programme which can be excuted in the embedded hardware system with low performance and low cost.

4. Simulation

4.1. Univariate Nonstationary Growth Model Example

The model is often used to evaluate the filter's performance [4]. Consider the following nonlinear model

$$x_k = 0.5x_{k-1} + 25x_{k-1}/(1 + x_{k-1}^2) + 8\cos(1.2(k-1)) + w_k \tag{11}$$

$$y_k = x_k^2/20 + v_k \tag{12}$$

Where w_k and v_k are zero-mean Gaussian white noise with variances 10.0 and 1.0 respectively. This example is severely nonlinear, both in the system and the measurement equation. The initial state was taken to be xo = 0.1, and the number of particles N=500. In order to compare with bootstrap filter [4], Fig. 1 shows the 50 iteration step comparison result. In Fig.1, the horizontal axis represents iteration step, and the vertical axis represents the state value. The true state is represented by a solid line, the Bootstrap filter result is represented by a star, the parallel resampling PF result is represented by a plus. The variance of parallel resampling PF is 12.52, while the variance of bootstrap filter is 18.25.So the performance of resampling PF is similar to bootstrap filter. Fig. 2 and Fig.3 show estimation of the posterior density funciton from both the parallel resampling PF and the bootstrap filter at $k = 20$.

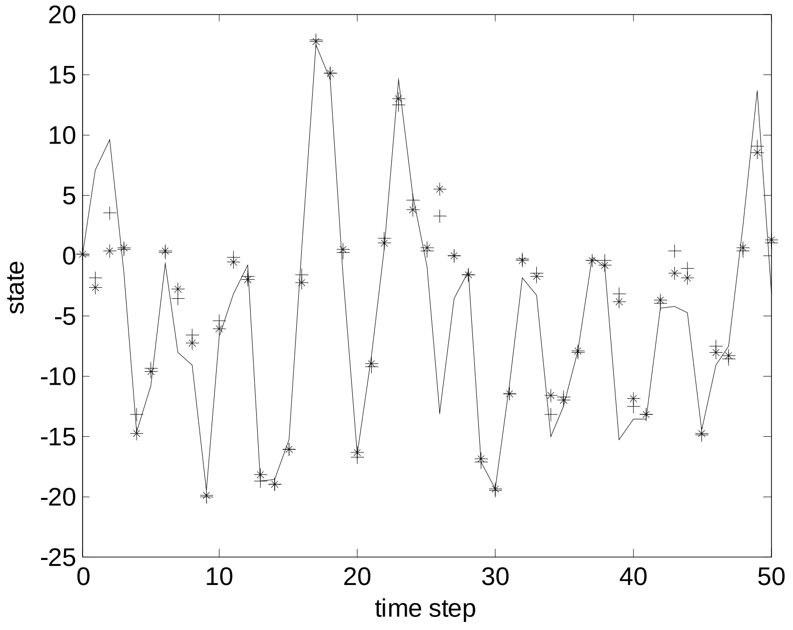


Fig.1 Comparison between Filter Algorithms

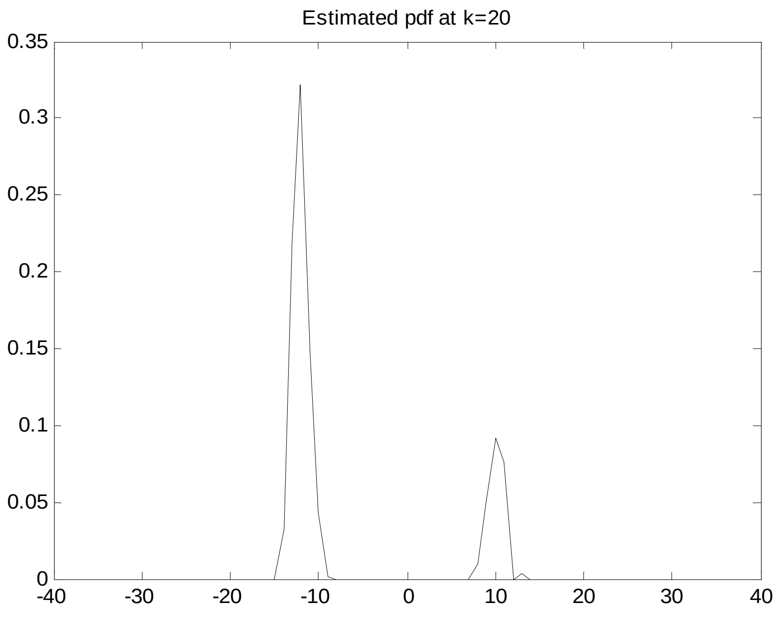


Fig.2 Pdf Value at k=20 Based on Parallel Resampling

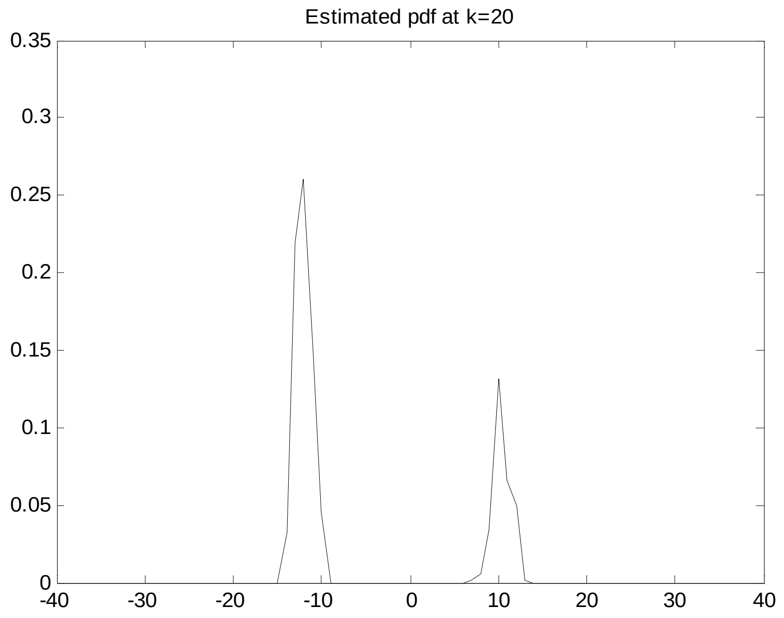


Fig.3 Pdf Value at k=20 Based on Bootstrap Algorithm

4.2. Visual Tracking of Moving Car

The child car is used to do the experiments. The speed and direction of car can be wirelessly controlled by the controller. The camera is fixed to record the trace of moving car.

The system state model of moving car

$$\mathrm{X}_k = \mathrm{AX}_{k-1} + \mathrm{B}w_{k-1}$$

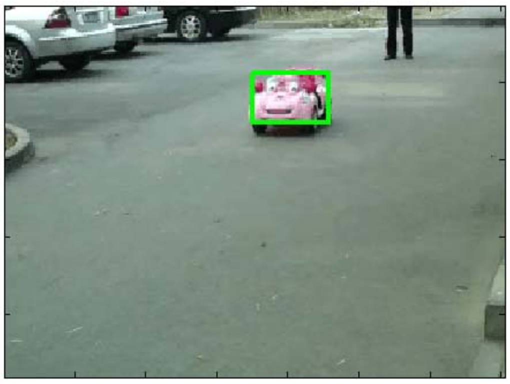
Where $\mathrm{X}_k = [x, v_x, a_x, y, v_y, a_y]'$,

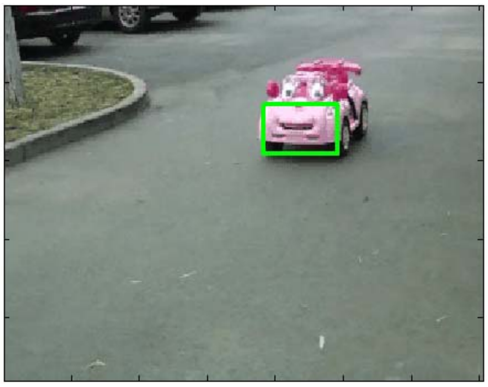
$$A = \begin{bmatrix} 1 & T & T^2/2 & 0 & 0 & 0 \\ 0 & 1 & T & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & T & T^2/2 \\ 0 & 0 & 0 & 0 & 1 & T \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \text{ and } B = [0 \quad 0 \quad T \quad 0 \quad 0 \quad T]'$$

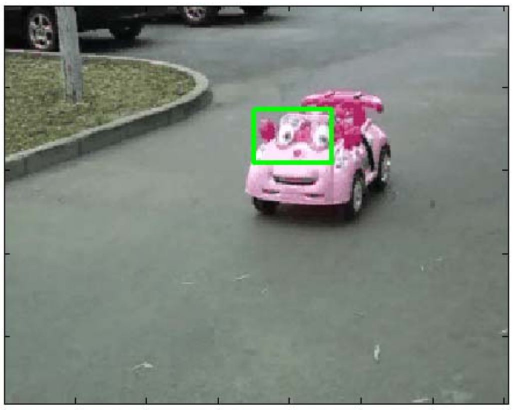
Here x and y denote the Cartesian co-ordinates of the target. x, v_x, a_x are respectively position, velocity, and acceleration along x direction at k time, while y, v_y, a_y are respectively position, velocity, and acceleration along y direction at k time. w_{k-1} is system noise. T is time interval.

The system observation value is the color characteristics of car's head. The color histograms method is used to extract the color characteristics. The Bhattacharyya coefficient has to be computed between the target histogram and the histogram from the state transition.

The parallel resampling PF is used in the camera controller based on the embedded system with ARM9 CPU. Fig.4 (a-d) shows the moving car tracking results by using the parallel resampling PF. In the algorithm, the number of particles $N = 300$, $T = 1s$ .The rectangle frame which locks the car's head in the figures is tracking frame, which shows the parallel resampling PF has good trakcing results.


(a) The No.35 Frame


(b) The No.130 Frame


(c) The No.163 Frame

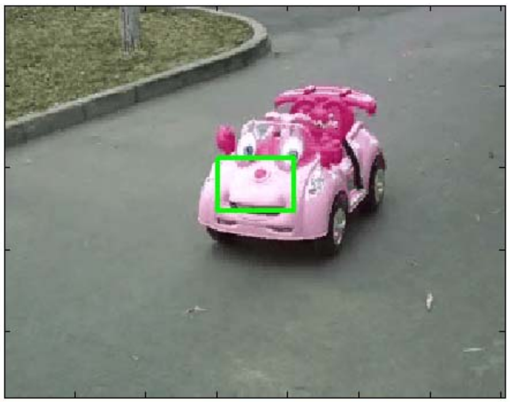

(d) The No.205 Frame

Fig.4 The Moving Car Tracking Results by Using The Parallel Resampling PF

5. Conclusion

Theoretical analysis and the experiment results show that the parallel resampling PF can reduce the search space for resampling, shorten the search time and has good execution efficiency. It is easily programmed by multithreading technology,which is suitable for the embedded hardware system platform.

Acknowledgement

This work is sponsored by PolyU Project (No.G-U499) and BJTU Project (No.2009JBM046 and No.151088522).

Reference

[1] P. M. Djuric, J. H. Kotecha, J. Zhang, Y. Huang, T. Ghirmai, M. F. Bugallo, and J. Miguez. Particle filtering. *IEEE Signal Proc. Mag.*,pages 19–38, 2003.

[2] B.Yuan, Z.X.Sun. Guide backgroud model update with object tracking.*Journal of Computational Information Systems*,4(4):1635-1642,2008.

[3] Y.M.Wang, J.X.Zhang, L.Wu, Z.Y.Zhou. Mean Shift Tracking algorithm based on multi-feature space and grey model.*Journal of Computational Information Systems*,6(11):3731-3739,2010.

[4] W. Fong, S. J. Godsill, A. Doucet, and M. West. Monte Carlo smoothing with application to audio signal enhancement. *IEEE Trans. Signal Process*, 50(2): 438–449, 2002.

[5] N. Gordon, D. Salmond, and A. Smith. Novel approach to non-linear/ non-Gaussian Bayesian state estimation. *IEE Proceedings-F*,14(2): 107–113, 1993.

[6] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P. J. Nordlund. Particle filters for positioning, navigation, and tracking. *IEEE Trans. Signal Process*, 50(2):425–437, 2002.

[7] Y. Huang and P. M. Djuric. A blind particle filtering detector of signals transmitted over flat fading channels. *IEEE Trans. Signal Process*, 52(7):1891–1900, 2004.

[8] Y.H.Yu,Q.S.Cheng. Particle filters for maneuvering target tracking problem.*Signal Processing*, pages 195-203, 2006.

[9] Ch.Cheng, R.Ansari. Kernel particle filter for visual tracking, *IEEE Signal Processing Letters*,12(3):242-245,2005.

[10] S.Hong, M.Bolic. An efficient fixed-point implementation of residual resampling scheme for high-speed particle filters, *IEEE Signal Processing Letters*,11(5):482-485,2004.