Progress In Electromagnetics Research, Vol. 120, 481–498, 2011

SIMPLIFIED PARTICLE PHD FILTER FOR MULTIPLE-TARGET TRACKING: ALGORITHM AND ARCHITECTURE

S. H. Hong[1,*], L. Wang[1], Z. G. Shi[2], and K. S. Chen[2]

[1]Department of Communication Engineering, Xiamen University, Xiamen, Fujian 361005, China

[2]Department of Information and Electronic Engineering, Zhejiang University, Hangzhou, Zhejiang 310027, China

Abstract—In this paper, we propose a simplified particle probability hypothesis density (PHD) filter and its hardware implementation for multiple-target tracking (MTT). In the proposed algorithm, the update step of particle PHD filter is simplified and the time-varying number of measurements is arranged in combination series/parallel mode. This may result in fixed hardware architecture and therefore present a convenient hardware implementation of particle PHD filter. Simulation results indicate that for the MTT problems, this proposed simplified algorithm shows similar performance with the standard particle PHD filter but has faster processing rate. Experiment study shows that the proposed simplified algorithm can be efficiently implemented in hardware and can effectively solve the MTT problems.

1. INTRODUCTION

Radar tracking systems extract information pertaining to locations or velocities of targets upon receiving the measurements of targets. Since radar tracking is of great importance to both civilian and military applications, multifarious new techniques are continuously applied in various radar tracking systems to improve the performance [1–7]. Among various radar tracking problems, multiple-target tracking (MTT) is an important topic with wide applications [8, 9]. The purpose of MTT is to identify targets and then to estimate the states of the

Received 19 August 2011, Accepted 26 September 2011, Scheduled 30 September 2011

* Corresponding author: Shaohua Hong (hongsh@xmu.edu.cn).

targets. There have been several methods used for MTT problems [10–13], such as joint probabilistic data association (JPDA) filter, multiple hypothesis tracking (MHT) filter, and so on. These methods do not directly get the number of targets and are mostly based on explicit data association. Thus, these algorithms entail high computational complexity. Recently, the probability hypothesis density (PHD) filter, which propagates only the first moment (or PHD) instead of the full multi-target posterior, has been shown to be a computationally efficient solution to the MTT problems [14–17]. Unfortunately, it is impossible to obtain closed-form solutions for the PHD filter since it still involves multiple integrals.

Particle filters (PFs) [18, 19], also known as Sequential Monte Carlo (SMC) filters, are commonly utilized for the approximation of intractable integrals and rely on the ability to draw random samples (or particles) from a probability distribution. The key idea is to represent the target posterior probability density function (PDF) of the state given the observations by a set of random particles with their associated weights. PFs have shown great promise in various target tracking problems [20–27] since they were proposed. However, for MTT problems, multi-target PF requires intensive computations and it is difficult to find an efficient importance density for the multi-target PF [15].

To have both the merits of PFs and PHD filter, starting from the fundamental idea of PF, The authors in [14, 15] proposed a particle PHD filter based on SMC method to approximate the PHD posterior intensity for tracking multiple targets. The principle of particle PHD filter is to use a large set of weighted samples (or particles) to represent the multi-target PHD. It has been shown that the particle PHD filter is suitable for tracking a set of time-varying number of targets in non-linear and non Gaussian situations. However, to our knowledge, all the works about particle PHD filter or even PHD filter focus on theoretical analysis, and no effort toward the hardware implementation has been exploited yet due to its high computational complexity and time-varying number of measurements. From theory to practice, the hardware implementation is an important topic. In this paper, we propose a simplified particle PHD filter algorithm and its hardware implementation to fill in this blank.

The rest of this paper is organized as follows. In Section 2, a brief review of the particle PHD filter is presented. In Section 3, we propose the simplified particle PHD filter. In Section 4, the corresponding hardware architecture is described in details. Simulation results and experimental study are given in Section 4 and we conclude this paper in Section 5.

2. BRIEF REVIEW OF PARTICLE PHD FILTER

The PHD function D_Ξ is the 1st order moment of the random finite set (RFS) Ξ and is defined by [14]

$$D_\Xi(x) \equiv E[\delta_\Xi(x)] = \int \delta_X(x) P_\Xi(dX) \tag{1}$$

where $\delta_\Xi(x) = \sum_{x\in\Xi} \delta_x$ is the random density representation of Ξ. P_Ξ is the probability distribution of the RFS Ξ. The PHD has the properties that, the integral $\int_S D_\Xi(x)\lambda(dx)$ is the expected number of targets in the measurable region S and the peaks of the PHD function give the estimates of the target states, where λ is the appropriate measure.

The PHD filter is a recursion of the PHD $D_{k|k}$ associated with the multi-target posterior density $p_{k|k}$. Given the RFS Ξ is Poisson, the recursion propagating $D_{k|k}$ is [14]

$$D_{k|k} = (\Psi_k \circ \Phi_{k|k-1})(D_{k-1|k-1}) \tag{2}$$

where "$\circ$" denotes composition of functions, $\Phi_{k|k-1}$ and Ψ_k are the PHD prediction and update operator respectively, which are defined as follows:

$$(\Phi_{k|k-1}\alpha)(x) = \gamma_k + \int \phi_{k|k-1}(x,\xi)\alpha(\xi)\lambda(d\xi) \tag{3}$$

$$(\Psi_k\alpha)(x) = \left[1 - P_D(x) + \sum_{z\in Z_k} \frac{\psi_{k,z}(x)}{\kappa_k(z) + \langle \psi_{k,z}, \alpha\rangle}\right]\alpha(x) \tag{4}$$

for any (integrable) function α on E_s, where

$$\begin{aligned}
\phi_{k|k-1}(x,\xi) &= b_{k|k-1}(x \mid \xi) + e_{k|k-1}(\xi) f_{k|k-1}(x \mid \xi) \\
\psi_{k,z}(x) &= p_D(x) g_k(z \mid x) \\
\kappa_z(z) &= \lambda_k c_k(z) \\
\langle f, g\rangle &= \int f(x) g(x) \lambda(dx)
\end{aligned}$$

and γ_k denotes PHD of the spontaneous RFS Γ_k; $b_{k|k-1}(\cdot \mid \xi)$ denotes the PHD of the RFS $B_{k|k-1}(\xi)$ spawned by a target with previous state ξ; $e_{k|k-1}(\xi)$ denotes the target survival probability with previous state ξ; $f_{k|k-1}(\cdot \mid \cdot)$ and $g_k(\cdot \mid \cdot)$ denote the transition probability density and the likelihood of individual targets, respectively; $p_D(\cdot)$ denotes the probability of detection; λ_k denotes the average number of Poisson clutter points per scan; and $c_k(\cdot)$ denotes the clutter probability density.

As described above, the PHD filter involves multiple integrals with no closed forms. SMC implementation of PHD filter (name as particle PHD filter) incorporates PF that utilizes a set of weighted particles to represent the PHD and has been proven to be suitable for MTT problems. Let L_k be the number of particles used for those targets surviving or spawned, and J_k be the number of particles used for new born targets.The Pseudo code of the particle PHD filter is described in Algorithm 1. For more details of discussion, please see [14, 15].

Algorithm 1 Pseudo code of PHD particle filter

$k = 0$;

- **Initialization**:
  - the posterior PHD $D_{0|0}$ is represented by a set of particles with their associated weights $\{x_0^{(i)}, w_0^{(i)}\}_{i=1}^{L_k}$.

For $k = 1, 2, 3, \ldots$

- **Prediction**:
  - For $i = 1, \ldots, L_{k-1}$
    - sample $\tilde{x}_k^{(i)} \sim q_k(\cdot \mid x_{k-1}^{(i)}, z_k)$
    - compute the predicted weights: $\tilde{w}_{k|k-1}^{(i)} = \frac{\phi_{k|k-1}(\tilde{x}_k^{(i)}, x_{k-1}^{(i)})}{q_k(\tilde{x}_k^{(i)} | x_{k-1}^{(i)}, z_k)} w_{k-1}^{(i)}$
  - For $i = L_{k-1} + 1, \ldots, L_{k-1} + J_k$
    - sample $\tilde{x}_k^{(i)} \sim p_k(\cdot \mid z_k)$
    - compute the predicted weights: $\tilde{w}_{k|k-1}^{(i)} = \frac{1}{J_k} \frac{\gamma_k(\tilde{x}_k^{(i)})}{p_k(\tilde{x}_k^{(i)} | z_k)}$
- **Update**:
  - For each $z \in Z_k$, compute: $C_k(z) = \sum_{i=1}^{L_{k-1}+J_k} \psi_{k,z}(\tilde{x}_k^{(i)}) \tilde{w}_{k|k-1}^{(i)}$
  - For $i = 1, \ldots, L_{k-1} + J_k$, update weights:

$$\tilde{w}_k^{(i)} = \left[1 - p_D(\tilde{x}_k^{(i)}) + \sum_{z \in Z_k} \frac{\psi_{k,z}(\tilde{x}_k^{(i)})}{\kappa_k(z) + C_k(z)} \right] \tilde{w}_{k|k-1}^{(i)}$$

- **Resample**:
  - compute the total mass: $\tilde{N}_{k|k} = \sum_{i=1}^{L_{k-1}+J_k} \tilde{w}_k^{(i)}$;
  - resample $\{\tilde{x}_k^{(i)}, \tilde{w}_k^{(i)} / \tilde{N}_{k|k}\}_{i=1}^{L_{k-1}+J_k}$ to get $\{x_k^{(i)}, w_k^{(i)} / \tilde{N}_{k|k}\}_{i=1}^{L_k}$;

  - rescale (multiply) the weights by $\tilde{N}_{k|k}$ to get $\{x_k^{(i)}, w_k^{(i)}\}_{i=1}^{L_k}$.
- **Target estimation**:
  - estimate the number of targets: $\hat{N}_k = round(\tilde{N}_{k|k})$;
  - use clustering algorithm to determine the $\hat{N}_k$ peaks of the posterior, which are the estimates of target states.

---

3. PROPOSED PARTICLE PHD FILTER

It is seen from Algorithm 1 that due to the time-varying number of measurements, the update step is difficult to be implemented in hardware. If one arranges the time-varying number of measurements in series mode, the latency will be unbearable and plenty of variables should be stored; if one arranges the time-varying number of measurements in parallel mode, the number of parallel implementation circuits will be difficult to be chosen and the efficiency of the hardware implementation would be affected. In this section, we propose a simplified version of particle PHD filter, which simplifies the update step. In the simplified update step, the time-varying number of measurements are skillfully arranged in combination series/parallel mode considering the compromise between the latency and efficiency of the hardware implementation.

In the update step of the particle PHD filter, given the probability of detection $p_D(\cdot)$, one can observe that if the $C_k(z)$ of a measurement is zero, that is, all the likelihoods of this measurement for the particles $g_k(\cdot \mid \cdot)$ are zeros, this measurement will make no contribution to the update weights. Obviously, the measurement can be discarded or replaced. Considering the compromise between the computational complexity and estimation accuracy, in this simplified algorithm, the $C_k(z)$ is compared with a threshold $T > 0$. The measurements with $C_k(z)$ greater than T are retained and the corresponding implementation circuits are kept for these measurements, whereas the measurements with $C_k(z)$ less than T can be replaced and the corresponding implementation circuits may be utilized for the other measurements. Thus we can arrange the time-varying number of measurements in combination series/parallel mode. This may remove the barrier of the hardware implementation and result in fixed hardware architecture with acceptable latency. Let m_k be the number of time-varying measurements and p be the number of parallel implementation circuits used; The details of the update step in this proposed simplified algorithm are shown in Algorithm 2.

Algorithm 2 The update step of the simplified particle PHD filter

---

Initialization: $u = 0$;
Compute C_k:
while $(u \leq m_k)$

- for $j = 1 : p$
  - if $(value(j) == 0)$
    - $u = u + 1$
    - if $(u > m_k)$
      - break;
    - end if
    - $value(j) = 1$;
    - $\psi_{k,z}(\tilde{x}_k^{(i)}) = p_D(\tilde{x}_k^{(i)}) g_k(z \mid \tilde{x}_k^{(i)})$
    - $C_k(j) = \sum_{i=1}^{L_{k-1}+J_k} \psi_{k,z}(\tilde{x}_k^{(i)}) \tilde{w}_{k|k-1}^{(i)}$
  - end if
- end for
- for $j = 1 : p$
  - if $(C_k(j) > T)$
    - if $(pos(j) == 0)$
      - $pos(j) = 1$;
    - end if
  - else
    - $value(j) = 0$;
  - end if
- end for

end while
update weights:

$$\tilde{w}_k^{(i)} = \left[1 - p_D(\tilde{x}_k^{(i)}) + \sum_{j=1}^{p} \frac{\psi_{k,z}(\tilde{x}_k^{(i)})}{\kappa_k(z) + C_k(j)} \right] \tilde{w}_{k|k-1}^{(i)}$$

---

where $value(j)$ for $1 \leq j \leq p$ represents whether the corresponding implementation circuit is running, and $pos(j)$ for $1 \leq j \leq p$ shows whether the variable $C_k(j)$ is updated. Clearly the choice of the number of parallel implementation circuits p should be larger than the maximum number of targets and is compromise. Too large a value would result in the low efficiency of hardware implementation but too small a value would discard some of the measurements generated by the targets.

From the Algorithm 2, one can observe that the time-varying number of measurements is arranged in combination series/parallel mode and therefore the update step can be achieved in fixed hardware architecture.

4. HARDWARE IMPLEMENTATION OF PARTICLE PHD FILTER

According to the description in Section 2, the hardware implementation of the particle PHD filter contains four important processing steps, namely *prediction* module, *update* module, *resample* module and *target estimation* module, as shown in Fig. 1, where *prediction* module samples particles for those targets surviving or spawned and generates particles for new born targets; *update* module updates the weights of particles given the measurements; *resample* module resamples the particles to reduce sample impoverishment and to prevent exponential growth of the size of the particle ensemble; and *target estimation* module estimates the number and the corresponding state of targets.

Resample module, the hardware implementation of which has been widely investigated [28, 29], can be designed according to the references. In this paper, we utilize the systematic resampling (SR) algorithm [28]. *Target estimation* module mainly consists of clustering algorithm to estimate the states of targets, which is very mature and already used in variety of fields. The K-means clustering that is a very popular clustering technique is used in this paper. There have been considerable interests in the FPGA implementation of K-means clustering algorithm [30, 31]. Certainly, considering the implementation complexity due to the loops of the clustering algorithm, the target estimation can also be processed with software, hardware/software co-design or DSP. Thus, in what follows, we focus on the *prediction* module and *update* module.

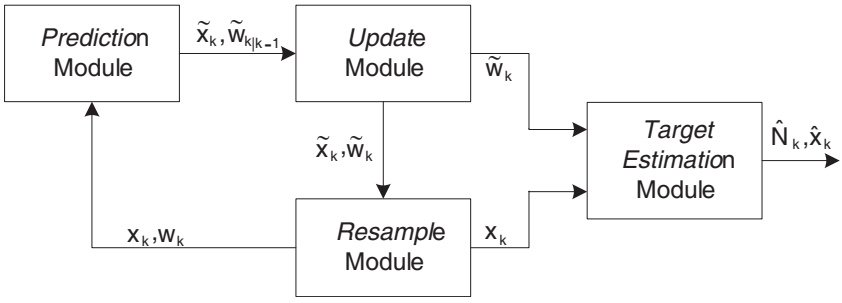

Figure 1. The module of particle PHD filter.

4.1. Prediction Module

Figure 2 shows the architecture of *prediction* module for the particles. Once the resample step is done, the memory *MEM* with depth L_k contains the replicated particles' indexes and the FIFO with depth $L_k + J_k$ stores the discarded indexes (since the number of discarded particles is nondeterministic). The memory *MEM* is then read sequentially and the indexes are utilized as addresses to the read port of the memory *PMEM* with depth $L_k + J_k$ storing the particles. Due to the nature of the SR algorithm and sequential read from the memory *MEM*, a replication signal *Rep* can be detected by comparing the current read index with the previous one stored in a temporary register *Reg1*. When an index is read from the memory *MEM* for the first time, the corresponding particle is read from the memory *PMEM* and stored in another temporary register *Reg2*. After propagation this predicted particle is written to its original location in the memory *PMEM*. In the following cycle, when the same index is read from the memory *MEM*, replication signal *Rep* is detected. Since the location in the memory *PMEM* has been overwritten by the propagated particle, the particle is read from the temporary register *Reg2* rather than from the memory *PMEM*. Meanwhile, a discarded index obtained form the FIFO simultaneously is used as address to the write port of memory *PMEM* to write the replicated particle after propagation. When all the replicated indexes are read off, particles for new born targets are generated and then writen into the memogy *PMEM* whose address is also the discarded index obtained from the FIFO (see Fig. 2).

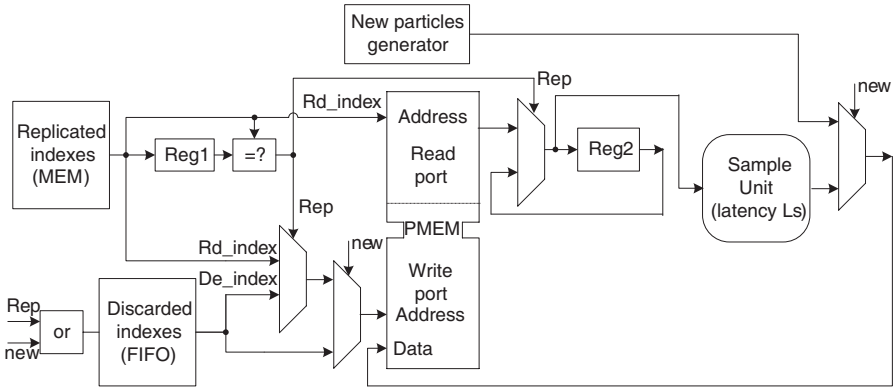

Figure 2. Architecture of prediction module.

4.2. Update Module

The architecture of *update* module, which updates the weights according to the measurements, is shown in Fig. 3. In this architecture, the number of parallel implementation circuits is set as p. Since all the p implementation circuits are to compute the $\frac{\psi_{k,z}(\tilde{x}_k^{(i)})}{\kappa_k(z)+C_k(z)}$, one structure is designed and is used for p times, which are actually p circuits in hardware implementation. Besides, there is a controller to generate control signals that arrange the time-varying number of measurements in combination series/parallel mode.

When the uth ($u \leq m_k$) measurement is ready, determine whether the p implementation circuits are free. If all the implementation circuits are busy, the measurement is kept until that one implementation circuit is free. If there are some free implementation circuits, the first free circuit j is obtained by judging $value(j) == 0$ for $1 \leq j \leq p$ and set $value(j) = 1$ to indicate that this implementation circuit is running. Then the likelihood function $g_k(z \mid \tilde{x}_k^{(i)})$ of every particle given this measurement is computed and multiplies by the probability of detection $p_D(\tilde{x}_k^{(i)})$ to get the $\psi_{k,z}(\tilde{x}_k^{(i)})$, which will be stored in a *RAM* for future use. Simultaneously the predicted weight $\tilde{w}_{k|k-1}^{(i)}$ multiplies by the $\psi_{k,z}(\tilde{x}_k^{(i)})$ corresponding to particle $\tilde{x}_k^{(i)}$ and the product is summed in an accumulator to get $C_k(z)$, which is used to compared with the threshold (T). The comparison results of the $C_k(z)$ and the threshold (T) are used to determine whether to retain the measurement. If the $C_k(z)$ is greater than the

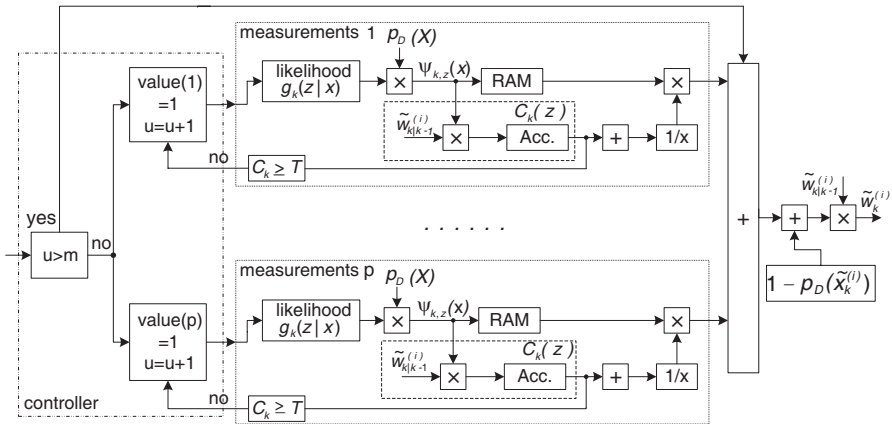

Figure 3. Architecture of update module.

threshold (T), reciprocal of the sum of the $C_k(z)$ and $\kappa_k(z)$ is computed and multiplies by the $\psi_{k,z}(\tilde{x}_k^{(i)})$ stored in RAM to get the $\frac{\psi_{k,z}(\tilde{x}_k^{(i)})}{\kappa_k(z)+C_k(z)}$ for this measurement. If the $C_k(z)$ is less than the threshold (T), that is, this measurement may make no contribution to the update weights, readjust $value(j) = 0$ to indicate that the implementation circuit is free and can be utilized for other measurements. For the rest of the measurements, similar operation is conducted.

After all the measurements are utilized in the update module, the values of $\frac{\psi_{k,z}(\tilde{x}_k^{(i)})}{\kappa_k(z)+C_k(z)}$ for all the p implementation circuits are obtained (for the free implementation circuit, $\frac{\psi_{k,z}(\tilde{x}_k^{(i)})}{\kappa_k(z)+C_k(z)} = 0$). Then all the values of $\frac{\psi_{k,z}(\tilde{x}_k^{(i)})}{\kappa_k(z)+C_k(z)}$ are added together and add $1 - p_D(\tilde{x}_k^{(i)})$. The results are used to multiply by the predicted weights $\tilde{w}_{k|k-1}^{(i)}$ and then updated weights $\tilde{w}_k^{(i)}$ are obtained.

5. SIMULATION RESULTS AND EXPERIMENTAL STUDY

To validate the proposed filter, we consider a two-dimensional scenario with an unknown and time-varying number of targets [15]. Each target moves according to the following dynamics

$$\mathbf{x}_k = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{x}_{k-1} + \begin{bmatrix} T^2/2 & 0 \\ T & 0 \\ 0 & T^2/2 \\ 0 & T \end{bmatrix} \mathbf{w}_k \tag{5}$$

where $\mathbf{x}_k = [x_k \ \dot{x}_k \ y_k \ \dot{y}_k]^T$ is target state vector at time kT (k is the time index) and the sampling period is $T = 1$. $\mathbf{w}_k = [w_k^x \ w_k^y]^T$ is the vector of independent zero-mean Gaussian white noise with standard deviations of $\begin{bmatrix} 1 & 0 \\ 0 & 0.1 \end{bmatrix}$. Targets can appear or disappear in the scene at any time. The probability of target survival is $e_{k|k-1}(\cdot) = 0.95$ and no spawning is considered for simplicity. Assume that target birth follows a Poisson model with the intensity $\gamma_k = 0.2N(\cdot; \bar{\mathbf{x}}, Q_{\mathbf{x}})$, where $N(\cdot; \bar{\mathbf{x}}, Q_{\mathbf{x}})$ denotes a normal density with mean $\bar{\mathbf{x}}$ and covariance $Q_{\mathbf{x}}$ and $\bar{\mathbf{x}} = \begin{bmatrix} 0 \\ 3 \\ 0 \\ -3 \end{bmatrix}$, $Q_{\mathbf{x}} = \begin{bmatrix} 10 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$.

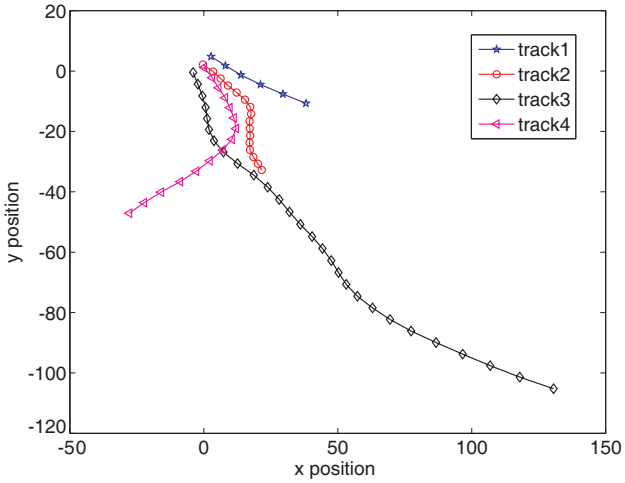

Figure 4. Positions of 4 tracks over 40 scans.

Figure 4 plots the true trajectories of 4 tracks over 40 scans. It can be seen that these 4 tracks start in the vicinity of the origin and move radially outwards. The corresponding start and finish times of the tracks can be found from Fig. 5.

The measurements originated from a target are given by

$$\mathbf{y}_k = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \mathbf{x}_k + \begin{bmatrix} v_{1,k} \\ v_{2,k} \end{bmatrix} \tag{6}$$

where $v_{1,k}$ and $v_{2,k}$ are independent zero-mean Gaussian measurement noise with standard deviations of $\sigma_{v_1} = \sigma_{v_2} = 2.5$. The probability of detection is set as $p_D(\mathbf{x}_k) = 1$. Clutter is uniformly distributed over the observation space of $[-100, 100] \times [-100, 100]$ with an average rate of $\gamma = 6$ points per scan.

In numerical simulations and hardware implementation, the number of particles used is $L_k = 1024$ and $J_k = 1024$. The importance sampling densities used are $q_k = f_{k|k-1}$ and $N(\cdot; \bar{\mathbf{x}}, Q_{\mathbf{x}})$, respectively. The number of parallel implementation circuits and the threshold are set as $p = 8$ and $T = 2^{-16}$, respectively. For comparison, the results of tracking the multiple trajectories using standard particle PHD filter are presented to gauge the performance.

Simulation results of the proposed simplified algorithm and the standard particle PHD filter are shown in Fig. 5, which plots the individual x and y coordinates of the tracks and estimated targets for each time step. It can be found that the estimated positions based

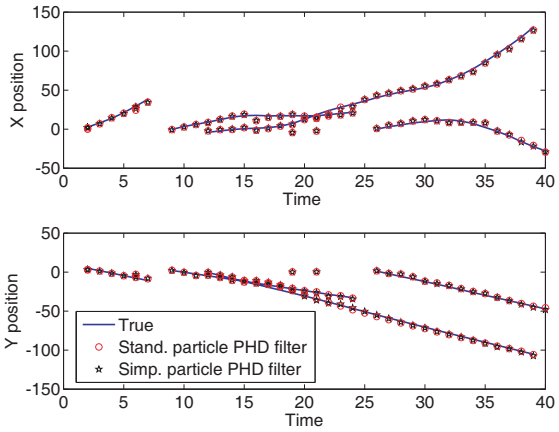

Figure 5. Comparison of the proposed simplified particle PHD filter and the standard particle PHD filter.

on the proposed simplified particle PHD filter and standard particle PHD filter are similar and they are all close to the true tracks.

In multi-target tracking, it was proposed in [32] to use the Wasserstein distance as a multi-target miss-distance between the true and the estimated sets of multi-target states, which is given by

$$d_p\left(\hat{X}_k, X_k\right) = \min_{C} \sqrt[p]{\sum_{i=1}^{|\hat{X}_k|}\sum_{j=1}^{|X_k|} C_{i,j}\|\hat{x}_{i,k} - x_{j,k}\|^p} \tag{7}$$

where X_k and $\hat{X}_k$ are the true and estimated set of the target states and the minimum is taken over the set of all transportation matrices C, whose entries $C_{i,j}$ satisfy $C_{i,j} \leq 0$, $\sum_{j=1}^{|X_k|} C_{i,j} = 1/ \leq |\hat{X}_k|$, $\sum_{j=1}^{|\hat{X}_k|} C_{i,j} = 1/|X_k|$.

Figure 6 plots the estimated targets against truth in terms of target number and Wasserstein multi-target miss-distance of the two algorithms at each time step for comparison. It can be found that again the proposed simplified particle PHD filter shows similar performance with standard particle PHD filter and the multi-target miss-distance exhibits peaks at the instances where the estimated number is incorrect. When the estimated number of targets is correct, the Wasserstein miss-distance is small.

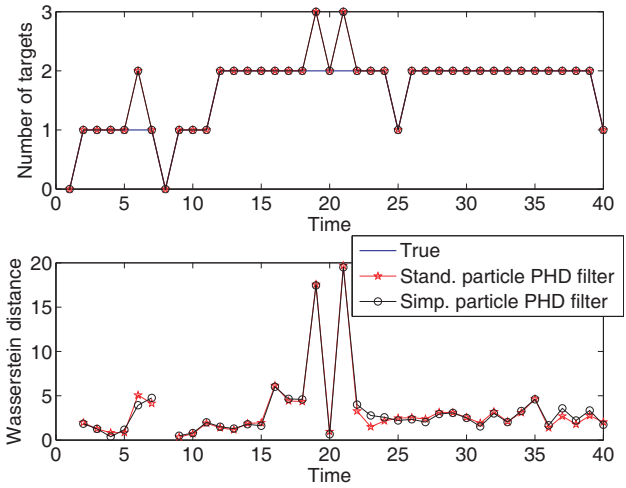

Figure 6. Target number estimate and multi-target miss-distance of the proposed simplified PHD filter and the standard particle PHD filter.

Table 1. The correct ratio of estimated number of targets and the multi-target miss-distance of the two filter algorithms.

Filter algorithm	The correct estimations	
	Correct ratio of number of targets	Wasserstein miss-distance
The standard particle PHD filter	86.81%	2.646
The proposed simplified particle PHD filter	86.78%	2.641

The correct estimation parameters including the correct ratio of estimated number of targets and the average multi-target miss-distance with the estimated number of targets correct are listed in Table 1. All the data in Table 1 is obtained through 1000 Monte Carlo simulations. From the Table 1, one can find that for the estimated number of targets, the standard particle PHD filter has the correct ratio of 86.81% while the proposed simplified one has 86.78%, and when the estimated number of targets is correct, the average Wasserstein miss-distance is 2.646 and 2.641, respectively. Comparison results confirm that the proposed simplified particle PHD filter shows similar performance with the standard one.

Table 2 shows the computational time in Matlab 7.7.0 on a Core 2 Duo 3.16-GHz PC with 4 GB RAM. Both the two filters run 1000 rounds of simulation and their computational time are recorded. Then the average time is computed. From the Table 2, it can be found that compared with the standard particle PHD filter, the proposed simplified particle PHD filter has faster processing rate. The reasons can be explained that the proposed simplified particle PHD filter discards the measurements with $C_k(z)$ less than the threshold T in the update step. This simplified update step can reduce the complexity and computational time. Further, the simplified algorithm removes the barrier of the hardware implementation and results in fixed hardware architecture for the implementation of particle PHD filter.

Except that the clustering step is done using Matlab, the hardware architectures of the rest are described in Verilog HDL and verified using Modelsim. The results of implementing the proposed simplified particle PHD filter algorithm on a Xilinx Virtex II device (XC2VP70FF1517) for the MTT problems are shown in Fig. 7 and

Table 2. The computational time of the two filter algorithms.

Filter algorithm	The computational time(s)
The standard particle PHD filter	2.856
The proposed simplified particle PHD filter	2.652

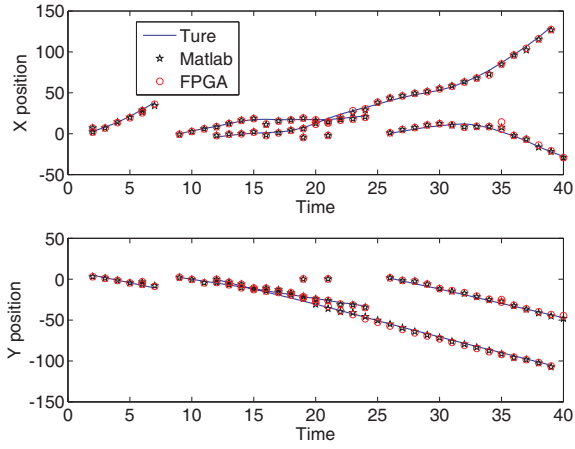

Figure 7. Comparison of the hardware results and matlab simulation of the proposed simplified particle PHD filter.

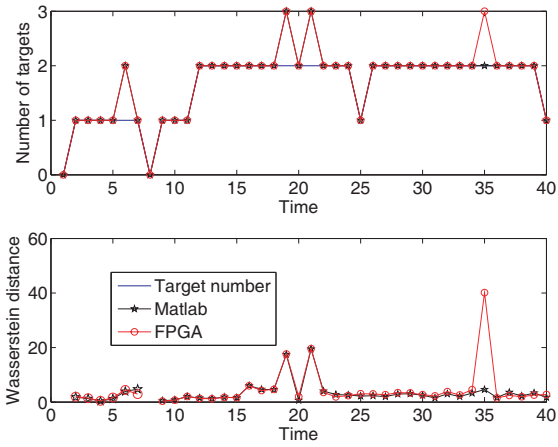

Figure 8. Target number estimate and multi-target miss-distance of the proposed simplified particle PHD filter by Matlab and FPGA.

Fig. 8, where Fig. 7 shows the individual x and y coordinates of the tracks and estimated targets at each time step and Fig. 8 plots the estimated targets against truth in terms of target number and Wasserstein multi-target miss-distance.

It is seen that the result from the hardware experiment agrees well with the simulated one. Detailed comparison shows that there are some small differences along with trajectory, which is mainly due to the different precision effects, time-series random noises and uniform random-numbers in simulation and hardware implementation. Thus, one can draw the conclusion that the simplified particle PHD filter can be applied to the MTT problems effectively.

6. CONCLUSION

In this paper, a simplified particle PHD filter and its hardware implementation are proposed for MTT problems. Numerical simulation results and experiment study indicate that compared with the standard particle PHD filter, the proposed simplified algorithm, which simplifies the update step and then removes the barrier of the hardware implementation, exhibits similar performances, has faster processing rate, and can be efficiently implemented in hardware. In this way, the proposed method can be used to effectively solve the MTT problems.

REFERENCES

1. Jiang, T., S. Qiao, Z. G. Shi, L. Peng, J. Huangfu, W. Z. Cui, W. Ma, and L. X. Ran, "Simulation and experimental evaluation of the radar signal performance of chaotic signals generated from a microwave Colpitts oscillator," *Progress In Electromagnetics Research*, Vol. 90, 15–30, 2009.
2. Li, Y., Y. J. Gu, Z. G. Shi, and K. S. Chen, "Robust adaptive beamforming based on particle filter with noise unknown," *Progress In Electromagnetics Research*, Vol. 90, 151–169, 2009.
3. Choi, G.-G., S.-H. Park, H.-T. Kim, and K.-T. Kim, "ISAR imaging of multiple targets based on particle swarm optimization and hough transform," *Journal of Electromagnetic Waves and Applications*, Vol. 23, No. 14–15, 1825–1834, 2009.
4. Guo, K.-Y, Q. Li, and X.-Q, Sheng, "A Precise recognition method of missile warhead and decoy in multi-target scene," *Journal of Electromagnetic Waves and Applications*, Vol. 24, No. 5–6, 641–652, 2010.
5. Le Marshall, N. W. D. and A. Z. Tirkel, "Modified matrix pencil algorithm for termite detection with high resolution radar," *Progress In Electromagnetics Research C*, Vol. 16, 51–67, 2010.
6. Li, M., G. Liao, and J. Liu, "Robust elevation prefiltering method for non-side looking airborne radar," *Journal of Electromagnetic Waves and Applications*, Vol. 24, No. 16, 2191–2205, 2010.
7. Chen, H. W., X. Li, J. Yang, W. Zhou, and Z. Zhuang, "Effects of geometry configurations on ambiguity properties for bistatic MIMO radar," *Progress In Electromagnetics Research B*, Vol. 30, 117–133, 2011.
8. Blackman, S. S., *Multiple Target Tracking with Radar Application*, Artech House, Boston, 1986.
9. Bar-Shalom, Y. and X. R. Li, *Multitarget-Multisensor Tracking: Principles and Techniques*, YBS Publishing, Storrs, CT, 1995.
10. Chang, K. C. and Y. Bar-Shalom, "Joint probabilistic data association for multitarget tracking with possibly unresolved measurements and maneuvers," *IEEE Transactions on Automatic Control*, Vol. 29, No. 7, 585–594, 1984.
11. Roecker, J. A., "Suboptimal joint probabilistic data association," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 29, No. 2, 510–517, 1993.
12. Blackman, S. S., "Multiple hypothesis tracking for multiple target tracking," *IEEE Aerospace and Electronic Systems Magazine*, Vol. 19, No. 1, 5–18, 2004.

13. Joo, S. W. and R. Chellappa, "A multiple-Hypothesis approach for multiobject visual tracking," *IEEE Transactions on Image Processing*, Vol. 16, No. 11, 2849–2854, 2007.
14. Mahler, R., "Multi-target Bayes filtering via first-order multi-target moment," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 39, No. 4, 1152–1178, 2003.
15. Vo, B. N., S. Singh, and A. Doucet, "Sequential monte carlo methods for multi-target filtering with random finite sets", *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 41, No. 4, 1224–1245, 2005.
16. Clark, D., I. T. Ruiz, Y. Petillot, and J. Bell, "Particle PHD filter multiple target tracking in sonar image," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 43, No. 1, 409–416, 2007.
17. Wang, Y. D., J. K. Wu, A. A. Kassim, and W. M. Huang, "Data-driven probability hypothesis density filter for visual tracking," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 18, No. 8, 1085–1095, 2008.
18. Arulampalam, M. S., S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian bayesian tracking," *IEEE Transactions on Signal Processing*, Vol. 50, No. 2, 174–187, 2002.
19. Ristic, B., S. Arulampalam, and N. Gordon, *Beyond the Kalman Filter: Particle Filter for Tracking Applications*, Artech House, Boston, 2004.
20. Zang, W., Z. G. Shi, S. C. Du, and K. S. Chen, "Novel roughening method for reentry vehicle tracking using particle filter," *Journal of Electromagnetic Waves and Applications*, Vol. 21, No. 14, 1969–1981, 2007
21. Hong, S. H., Z. G. Shi, and K. S. Chen, "Novel roughening algorithm and hardware architecture for bearings-only tracking using particle filter," *Journal of Electromagnetic Waves and Applications*, Vol. 22, No. 2–3, 411–422, 2008.
22. Shi, Z. G., S. H. Hong, and K. S. Chen, "Tracking airborne targets hidden in blind doppler using current statistical model particle filter," *Progress In Electromagnetics Research*, Vol. 82, 227–240, 2008.
23. Chen, J. F., Z. G. Shi, S. H. Hong, and K. S. Chen, "Grey prediction based particle filter for maneuvering target tracking," *Progress In Electromagnetics Research*, Vol. 93, 237–254, 2009.
24. Liu, H. Q., H. C. So, F. K. W. Chan, and K. W. K. Lui, "Distributed particle filter for target tracking in sensor networks,"

Progress In Electromagnetics Research C, Vol. 11, 171–182, 2009.

25. Liu, H.-Q. and H.-C. So, "Target tracking with line-of-sight identification in sensor networks under unknown measurement noises," *Progress In Electromagnetics Research*, Vol. 97, 373–389, 2009.
26. Wang, Q., J. Li, M. Zhang, and C. Yang, "H-infinity filter based particle filter for maneuvering target tracking," *Progress In Electromagnetics Research B*, Vol. 30, 103–116, 2011.
27. Wang, X. F., J. F. Chen, Z. G. Shi, and K. S. Chen, "Fuzzy-control-based particle filter for maneuvering target tracking," *Progress In Electromagnetics Research*, Vol. 118, 1–15, 2011.
28. Athalye, A., M. Bolic, S. Hong, and P. M. Djuric, "Generic hardware architectures for sampling and resampling in particle filters," *EURASIP Jounal on Applied Signal Processing*, Vol. 2005, No. 17, 2888–2902, 2005.
29. Hong, S. H., Z. G. Shi, and K. S. Chen, "A low-power memory-efficient resampling architecture for particle filters", *Circuit, System, and Signal Processing*, Vol. 29, 155–167, 2010.
30. Estlick, M., M. Leeser, J. Theiler, and J. J. Szymanshi, "Algorithmic transformations in the implementation of K-means clustering on reconfigurable hardware," *International Symposium on Field Programmable Gate Arrays*, 103–110, 2001.
31. Saegusa, T. and T. Maruyama, "An FPGA implementation of real-time K-means clustering for color images", *Journal of Real-Time Image Processing*, Vol. 2, No. 4, 309–318, 2007.
32. Hoffman, J. R. and R. P. S. Mahler, "Multitarget miss distance via optimal assignment," *IEEE Transactions on Systems, Man and Cybernetics-Part A: Systems and Humans*, Vol. 34, No. 3, 327–336, 2004.