POSE: Design of Hardware-Friendly Particle-Based Observation Selection PHD Filter

Zhiguo Shi, *Member, IEEE*, Yongkang Liu, Shaohua Hong, *Member, IEEE*, Jiming Chen, *Senior Member, IEEE*, and Xuemin (Sherman) Shen, *Fellow, IEEE*

***Abstract*—Particle probability hypothesis density (PHD) filtering is a promising technology for the multitarget-tracking problem. Traditional particle PHD filter solutions usually have high computational complexity, and the lack of dedicated hardware has seriously limited their usages in real-time industrial applications. The hardware implementation difficulty of the particle PHD filtering in field-programmable gate array (FPGA) platforms lies in that the number of observations for filtering is time varying while the number of parallel processing units in circuit is fixed. To overcome this challenge, we propose a novel particle-based observation selection (POSE) PHD filter algorithm and its hardware implementation in this paper. Specifically, we opportunistically select a fixed number of observations out of a varying number of observations for filtering, where the approximation error is proved to be negligible by adapting the circuit budget to the environment accordingly. To implement the proposed POSE PHD filter, the hardware design issues are addressed in depth. Extensive simulations demonstrate that the POSE PHD filter has a comparable performance with the traditional one while its hardware implementation challenge is overcome. The hardware experiment results of the POSE PHD filter on a Xilinx Virtex-II Pro FPGA platform match the simulation ones well. Furthermore, the execution time of the implemented hardware circuit is evaluated, and the results show that it can achieve a processing rate of 6.892 kHz with a 50-MHz system clock.**

***Index Terms*—Field-programmable gate array (FPGA) platform, hardware design, multitarget tracking (MTT), particle probability hypothesis density (PHD) filter, real-time performance.**

Manuscript received August 15, 2012; revised February 18, 2013; accepted April 11, 2013. Date of publication May 13, 2013; date of current version September 19, 2013. This work was supported in part by the National Science Foundation of China under Grant 61171149, by the Research Foundation of the Chinese State Key Laboratory of Industrial Control Technology under Grant ICT1119, NCET-11-0445, and 863 High-Tech Project under Grant 2011AA040101-1, by the Fundamental Research Funds for the Chinese Central Universities under Grant 2013xzzx008-2, and by Ontario Research Fund–Research Excellence (ORF-RE), Ontario, Canada.

Z. Shi is with the Department of Information and Electronic Engineering, Zhejiang University, Hangzhou 310027, China, and also with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: shizg@zju.edu.cn).

Y. Liu and X. Shen are with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: y257liu@bbcr.uwaterloo.ca; xshen@bbcr.uwaterloo.ca).

S. Hong is with the Department of Communication Engineering, Xiamen University, Xiamen 361005, China (e-mail: hongsh@xmu.edu.cn).

J. Chen is with the State Key Laboratory of Industrial Control Technology, Zhejiang University, Hangzhou 310027, China (e-mail: jmchen@ieee.org).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIE.2013.2262753

I. Introduction

Multi-target tracking (MTT) is a crucial issue for many industrial applications [1]–[4]. For example, in vehicular ad hoc networks (VANETs), it usually needs to track the states of multiple vehicles on the roads based on clutter-contaminated observations for regulators to predict and manage traffic as well as for drivers to achieve driving-safety and route planning [5]–[8]. Since clutters are usually caused by terrain factors, weather systems, and/or unrelated moving objects, such as birds [9], the number of observations varies at different sensing moments. Thus, the major challenge in MTT is how to effectively differentiate the observations generated by real targets from those which are generated by clutters in an observation set and estimate the state of each target from the available observations.

In recent years, probability hypothesis density (PHD) filtering has emerged as a promising technology for the MTT problem [10]–[13]. PHD filters represent the targets and observations as random finite sets (RFSs) and use the finite set statistics (FISST) to solve the MTT problem under the Bayesian framework. Although there are many variations of PHD filters, such as the cardinalized PHD (CPHD) and the multitarget multi-Bernoulli (MeMBer) filter [13]–[16], it is generally impossible to get closed-form solutions for PHD filters as the recursion involves multiple integral calculations. Among the attempts for approximate algorithms of PHD filters, the particle PHD filter shows its potentials as a generic method in solving the MTT problem [11]. Based upon the similar idea of particle filters (PFs) [17], [18], the particle PHD filter uses the sequential Monte Carlo (MC) (SMC) method to approximate the PHD posterior density and can be applied to nonlinear and non-Gaussian MTT problems in denser clutter environments. However, the SMC method has high computational complexity intrinsically and thus limits the usage of the particle PHD filter in real-time industrial applications [7], [19].

On the contrary, the demands of "real-time" MTT have been increasing [20], [21]. To improve the processing speed, one of the main methods is to implement filters in dedicated hardware circuits such as field-programmable gate array (FPGA) platforms [22]–[26], which are very flexible for the hardware design in the prototype phase and can be easily converted to application-specific integrated circuits for final commercialized products [27]. Specifically, the FPGA platforms offer the possibility of fully parallel implementation of digital circuits by allowing concurrent processing in both time and space dimensions to speed up the processing. The FPGA-based hardware implementation has been proved to achieve significant real-time performance improvement for the PFs which use the SMC method to represent the posterior probability density, by utilizing distributed concurrent processing [28], [29]. For the traditional particle PHD filter [11], however, these designs cannot be directly applied due to its unique challenges as follows.

First, the hardware implementation requires a fixed number of observation processing elements to work in parallel with a deterministic processing delay. To achieve this, in the traditional particle PHD filter where the number of observations is time varying, an algorithm is needed to select a fixed number of observations from the available observations. In addition, the design of observation selection in hardware circuit is another issue since the failure of selecting the target-oriented observations would have negative impacts on the tracking performance. Furthermore, the hardware design of the update module in the traditional particle PHD filter is fairly complex and challenging due to various complicated arithmetic operations, such as accumulation operation, likelihood calculation, and division.

Recently, some research efforts have been put on the implementation of the particle PHD filter. Hong *et al.* [30] propose a simplified particle PHD filter and its hardware implementation, where the update step is simplified to reduce the implementation complexity. However, such modification introduces longer execution delay, and whether such simplification will cause performance degradation has not been analyzed. In addition, Miao *et al.* [31] implement a particle PHD filter in hardware for a real-time closed-loop tracking with unknown neural sources, yet they assume that the number of observations is fixed, which does not always hold in the real case. More recently, Shi *et al.* propose a threshold-based resampling for high-speed particle PHD filtering [32]. However, the observation selection for hardware implementation of the particle PHD filter remains a fundamental problem which needs more efforts. In this paper, we try to address this difficulty in depth from algorithm to implementation. The main contributions of this paper can be summarized as follows.

- First, we propose a novel particle-based observation selection (POSE) PHD filter which overcomes the difficulty of implementing the traditional particle PHD filter in FPGA platforms, and we present the mathematic analysis to show that the observation selection error probability (OSEP) is negligible when a proper number of selected observations is set according to the environment accordingly.
- Second, for the hardware implementation of the POSE PHD filter, we propose the design of the observation selection module into the implementation of the traditional particle PHD filter architecture and present the customization of other hardware modules in it.
- Third, by extensive simulations and circuit experiments, we validate the effectiveness of the POSE PHD filter and show that it can achieve a processing rate of 6.892 kHz with a 50-MHz system clock.

The remainder of this paper is organized as follows. The problem formulation of a multiple tracking in an industrial application is presented in Section II. The algorithmic design and hardware implementation of the POSE PHD filter are proposed in Section III. The simulation results and hardware implementation of the proposed POSE PHD filter are given in Section IV, followed by the concluding remarks in Section V.

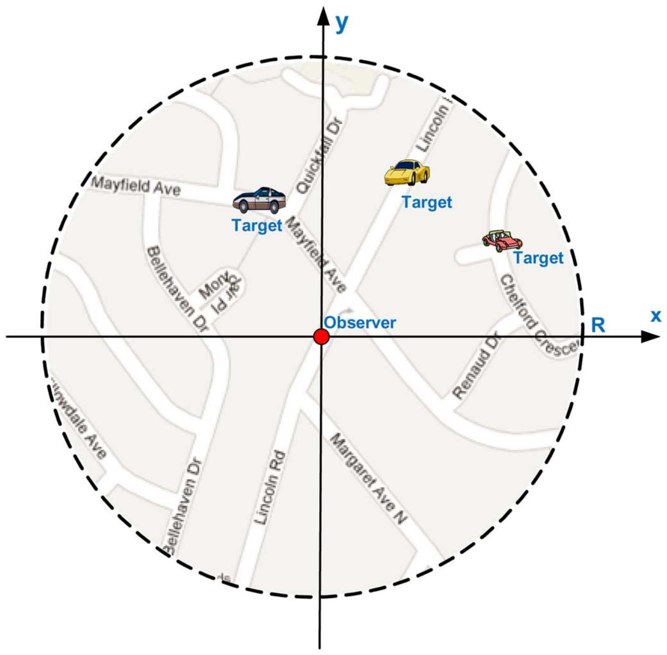


Fig. 1. MTT in VANET.

II. Problem Formulation

We consider the MTT problem in VANET in a circular observation region with an observer, e.g., a roadside unit, located at the origin, as shown in Fig. 1. Vehicles, i.e., targets, move into and out of the observation region continuously in any arbitrary directions, and no road information is available for the observer. The state of a target at the moment k, $\boldsymbol{x}_k = [x_k, \dot{x_k}, y_k, \dot{y_k}]^T$, contains the position information of the target, (x_k, y_k), and its velocities, $(\dot{x_k}, \dot{y_k})$, in the x- and y-directions, respectively. The state equation of the target in the $x-y$ plane is described by

$$\boldsymbol{x}_k = \begin{bmatrix} 1 & \delta T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \delta T \\ 0 & 0 & 0 & 1 \end{bmatrix} \boldsymbol{x}_{k-1} + \begin{bmatrix} \frac{\delta T^2}{2} & 0 \\ \delta T & 0 \\ 0 & \frac{\delta T^2}{2} \\ 0 & \delta T \end{bmatrix} \begin{bmatrix} \kappa_{1,k} \\ \kappa_{2,k} \end{bmatrix} \quad (1)$$

where δT is the sensing period and $\kappa_{1,k}$ and $\kappa_{2,k}$ are independent zero-mean white Gaussian noises with standard deviations σ_{κ_1} and σ_{κ_2}, respectively [11]. As many symbols are used in this paper, Table I summarizes the important ones.

For the observer, the number of targets in the region varies with time. Thus, the tracked targets at each scan can be categorized into two types: survival targets and newborn targets, accordingly. The survival targets are those which appear in previous scans while the newborn targets are those newly detected in the region. We consider that, at most, one newborn target can appear in a short δT. For the newborn targets, they can appear spontaneously according to a Poisson point process with intensity function $\gamma_k = 0.2N(\cdot|\bar{\mathbf{x}}_0, Q)$, which denotes a normal distribution with mean $\bar{\mathbf{x}}_0$ and covariance Q. For the survival targets,

TABLE I
SOME IMPORTANT NOTATIONS

$\overline{d}_p^{(c)}$	Optimal Sub-Pattern Assignment (OSPA)
$e_{k\|k-1}$	Target survival probability from time $k-1$ to k
$m, \breve{N}_k$	$m = \breve{N}_k$ is the number of targets
$\breve{N}_{k\|k}$	Sum of weights of particles
p_D	Probability of detection
δT	Sensing period
$w_k^{(i)}$	The associated weight of particle $\boldsymbol{x}_{\boldsymbol{k}}^{(i)}$
$\tilde{w}_k^{(i)}$	Weight of particle $\tilde{\boldsymbol{x}}_{k\|k-1}^{(i)}$ in update step
$\hat{w}_{k\|k-1}^{(i)}$	Weight of particle $\tilde{\boldsymbol{x}}_{k\|k-1}^{(i)}$ in prediction step
$\bar{\mathbf{x}}_0$	Mean of Gaussian distribution for new born target
$\bar{\mathbf{x}}_1$	One-step state prediction of $\bar{\mathbf{x}}_0$
$\boldsymbol{x}_{\boldsymbol{k}}$	Target state at time k, is a vector $[x_k, \dot{x}_k, y_k, \dot{y}_k]$
$\boldsymbol{x}_k^{(i)}$	The i-th particle at time k after resampling
$\breve{\boldsymbol{x}}_k^{(i)}$	State of survival targets
$\tilde{\boldsymbol{x}}_{k\|k-1}^{(i)}$	The i-th particle in time k in prediction from $k-1$
$\breve{\boldsymbol{X}}_k$	Survival target set
$\boldsymbol{z}_k$	Observations after selection
$\boldsymbol{Z}_k$	Observation set after selection
$\tilde{\boldsymbol{z}}_k$	Observations before selection
$\tilde{\boldsymbol{Z}}_k$	Observation set before selection
$\breve{\boldsymbol{Z}}_k$	Predicted observation set from $\breve{\boldsymbol{X}}_k$
$\zeta_{r,k}$	Zero-mean white Gaussian noise for distance r_k
$\zeta_{\theta,k}$	Zero-mean white Gaussian noise for bearing θ_k
$\kappa_{i,k}$	Zero mean white Gaussian noise for state equation
σ_{κ_i}	Standard deviations of $\kappa_{i,k}$
γ_k	PHD of the RFS for new born targets
σ_r	Standard deviations of $\zeta_{r,k}$
σ_θ	Standard deviations of $\zeta_{\theta,k}$
λ	the expected number of clutters per scan

the probability of target survival from time $k-1$ to time k is $e_{k|k-1}$. No target spawning is considered in this paper [10]. The maximum number of the targets at each sensing moment is denoted as P. The probability of detection is denoted as p_D.

At each sensing moment, the observer obtains observations, $\tilde{\boldsymbol{Z}}_k$, from the targets and clutters in the circular region with a radius R. The target-oriented observation equation is given by

$$\boldsymbol{z}_k = H(\boldsymbol{x}_k) = \begin{bmatrix} r_k \\ \theta_k \end{bmatrix} \tag{2}$$

where

$$r_k = \left\| \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \mathbf{x}_k \right\| + \zeta_{r,k} \tag{3}$$

$$\theta_k = \arctan\left(\frac{[0 \;\; 0 \;\; 1 \;\; 0]\, \mathbf{x}_k}{[1 \;\; 0 \;\; 0 \;\; 0]\, \mathbf{x}_k} \right) + \zeta_{\theta,k}. \tag{4}$$

The observations consist of both the target range r_k and bearing θ_k. $\zeta_{r,k}$ and $\zeta_{\theta,k}$ are independent zero-mean white Gaussian noises with standard deviations σ_r and σ_θ.

Clutters are uniformly distributed in the observation region, and the number of clutter points per scan obeys a Poisson distribution with an average rate of λ [9].

III. ALGORITHMIC DESIGN AND HARDWARE IMPLEMENTATION

In this section, we first propose the POSE PHD filter design which is suitable for hardware implementation in FPGA

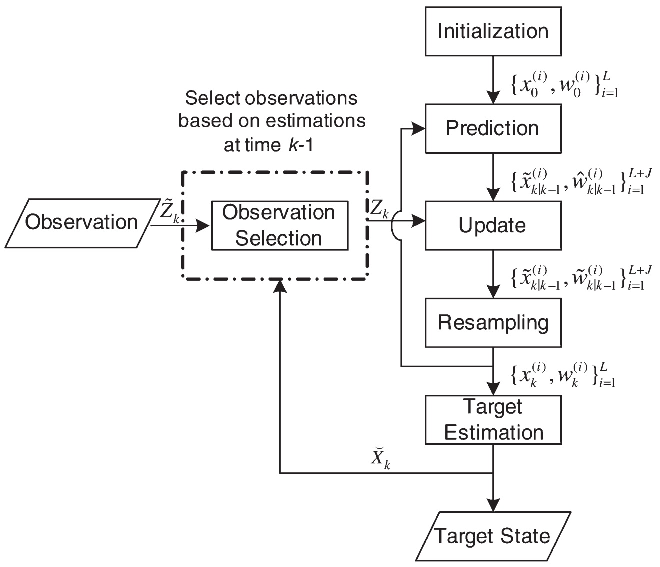

Initialization
$\{x_0^{(i)}, w_0^{(i)}\}_{i=1}^{L}$
Select observations based on estimations at time k-1
Prediction
$\{\tilde{x}_{k|k-1}^{(i)}, \hat{w}_{k|k-1}^{(i)}\}_{i=1}^{L+J}$
Observation
$\tilde{Z}_k$
Observation Selection
Z_k
Update
$\{\tilde{x}_{k|k-1}^{(i)}, \tilde{w}_{k|k-1}^{(i)}\}_{i=1}^{L+J}$
Resampling
$\{x_k^{(i)}, w_k^{(i)}\}_{i=1}^{L}$
Target Estimation
$\breve{X}_k$
Target State

Fig. 2. Processing flow of the POSE PHD filter.

platforms. We then analyze the observation selection process and show that the OSEP is negligible in the POSE PHD filter. Furthermore, we address the hardware implementation issues of the POSE PHD filter. Finally, we propose a heuristic method to select the proper particle number based on the "lost tracking ratio" parameter and discuss the computational complexity problem.

A. POSE PHD Filter Overview

The processing flow of the POSE PHD filter contains six steps, namely, *Initialization*, *Prediction*, *Observation Selection*, *Update*, *Resampling*, and *Target Estimation*, as shown in Fig. 2. In comparison with the traditional particle PHD filter, we introduce a novel *Observation Selection* module to preprocess the observations to meet the hardware design requirements. In the processing flow, *Initialization* is executed at the beginning to initiate the processing parameters, while the other five steps are executed recursively each time new observations arrive at time $k = 1, 2, 3\ldots$.

1) *Initialization*: At time $k = 0$, the posterior PHD of targets is represented by a set of particles (samples) $\{\boldsymbol{x}_0^{(i)}, w_0^{(i)}\}_{i=1}^{L}$, where $\boldsymbol{x}_0^{(i)}$ denotes the ith particle at time $k = 0$ and $w_0^{(i)}$ denotes the associated weight of the particle $\boldsymbol{x}_0^{(i)}$.
2) *Prediction*: The algorithm generates L particles $\{\tilde{\boldsymbol{x}}_{k|k-1}^{(i)}\}_{i=1}^{L}$ from the proposal density $q_k(\cdot|\boldsymbol{x}_{k-1}^{(i)}, \boldsymbol{Z}_k)$ for the surviving targets and generates J particles $\{\tilde{\boldsymbol{x}}_{k|k-1}^{(i)}\}_{i=L+1}^{L+J}$ from the proposal density $p_k(\cdot|\boldsymbol{Z}_k)$ for the newborn targets. The prediction weight of the ith particle at time k is given by

$$\hat{w}_{k|k-1}^{(i)} = \begin{cases} \dfrac{\phi_{k|k-1}\left(\tilde{\boldsymbol{x}}_{k|k-1}^{(i)}, \boldsymbol{x}_{k-1}^{(i)}\right)}{q_k\left(\tilde{\boldsymbol{x}}_{k|k-1}^{(i)} | \boldsymbol{x}_{k-1}^{(i)}, \boldsymbol{z}_k\right)} w_{k-1}^{(i)} & i = 1, \ldots, L \\ \dfrac{1}{J} \dfrac{\gamma_k\left(\tilde{\boldsymbol{x}}_{k|k-1}^{(i)}\right)}{p_k\left(\tilde{\boldsymbol{x}}_{k|k-1}^{(i)} | \boldsymbol{z}_k\right)} & i = L+1, \ldots L+J \end{cases} \tag{5}$$

where $\phi_{k|k-1}(\boldsymbol{x},\boldsymbol{\xi}) = e_{k|k-1}(\boldsymbol{\xi})f_{k|k-1}(\boldsymbol{x}|\boldsymbol{\xi})$. $e_{k|k-1}(\boldsymbol{\xi})$ is the target survival probability given that the previous state was $\boldsymbol{\xi}$, and $f_{k|k-1}(\boldsymbol{x}|\boldsymbol{\xi})$ is the transition probability of the survival target from previous state $\boldsymbol{\xi}$ to the sampled particle $\boldsymbol{x}$.

3) *Observation Selection*: For the survival targets, it generates the Euclidean distance D_{sur} and selects M_1 targets from the observation set $\tilde{\boldsymbol{Z}}_k$ with elements $\tilde{\boldsymbol{z}}_k$. For the newborn targets, it generates the Euclidean distance D_{new} and then selects M_2 targets from $\tilde{\boldsymbol{Z}}_k$. The selected observation set is denoted as $\boldsymbol{Z}_k$. We will address the details of this step later.

4) *Update*: Using the selected observations $\boldsymbol{Z}_k$ at time k, it updates the weights of particles

$$\tilde{w}_k^{(i)} = \left[1 - p_D + \sum_{\boldsymbol{z}\in\boldsymbol{Z}_k} \frac{\psi_{k,\boldsymbol{z}}\left(\tilde{\boldsymbol{x}}_{k|k-1}^{(i)}\right)}{\kappa_k(\boldsymbol{z}) + C_k(\boldsymbol{z})}\right] \hat{w}_{k|k-1}^{(i)} \tag{6}$$

where

$$C_k(\boldsymbol{z}) = \sum_{i=1}^{L+J} \psi_{k,\boldsymbol{z}}\left(\tilde{\boldsymbol{x}}_{k|k-1}^{(i)}\right) \hat{w}_{k|k-1}^{(i)} \tag{7}$$

with $\psi_{k,\boldsymbol{z}}(\boldsymbol{x}) = p_D g_k(\boldsymbol{z}|\boldsymbol{x})$ and $\kappa_k(\boldsymbol{z}) = \lambda c_k(\boldsymbol{z})$. $g_k(\boldsymbol{z}|\boldsymbol{x})$ denotes the likelihood of individual targets, and $c_k(\boldsymbol{z})$ is the probability density of clutters.

5) *Resampling*: By computing the sum of all weights, i.e., $\tilde{N}_{k|k} = \sum_{i=1}^{L+J} \tilde{w}_k^{(i)}$, it first estimates the number of targets $\breve{N}_k = round(\tilde{N}_{k|k})$. Then, it resamples the particle set $\{\tilde{x}_{k|k-1}^{(i)}, \tilde{w}_k^{(i)}/\breve{N}_k\}_{i=1}^{L}$ and rescales the weights by $\breve{N}_k$ to get L particles $\{x_k^{(i)}, w_k^{(i)}\}_{i=1}^{L}$.

6) *Target Estimation*: Using clustering algorithms, such as the K-means clustering [33], we determine the $\breve{N}_k$ peaks of the posterior and obtain the state estimation set $\breve{\boldsymbol{X}}_k$, where each element represents a target state estimate $\breve{\boldsymbol{x}}_k^{(i)}$, $i = 1, 2, \ldots, \breve{N}_k$.

B. Observation Selection

In the following, we discuss the detailed design of the *observation selection* scheme. In the POSE PHD filter, observations are first processed by the observation selection module to obtain a fixed number of observations M, which agrees with the fixed number of parallel observation processing units in the hardware implementation. Denote the time-varying number of received observations as U. If $U < M$, some NULL observations are added. Otherwise, when $U > M$, $M = M_1 + M_2$ observations are selected from U, where M_1 and M_2 observations are used for the survival targets and the newborn targets, respectively.

Observation Selection for Survival Targets: For the survival targets, the estimated state $\breve{\boldsymbol{x}}_{k-1}$ at time $k-1$ is used to compute the predicted state at time k, i.e., $A\breve{\boldsymbol{x}}_{k-1}$. Then, the Euclidean distance is computed between the predicted observation $\breve{\boldsymbol{z}}_k = H(A\breve{\boldsymbol{x}}_{k-1})$ and the observation $\tilde{\boldsymbol{z}}_k$. Suppose that, at time $k-1$, there are $m = \breve{N}_{k-1}$ targets and the predicted observation set $\breve{\boldsymbol{Z}}_k$ at the current moment has m elements $\breve{\boldsymbol{z}}_i$, $i = 1, 2, \ldots, m$, accordingly. Meanwhile, the received observation set $\tilde{\boldsymbol{Z}}_k$ has U elements $\tilde{\boldsymbol{z}}_j$, $j = 1, 2, \ldots, U$. Thus, a m by U matrix D_{sur} of the Euclidean distance is presented by

$$D_{sur} = \begin{bmatrix} d_{11}, d_{12}, \ldots, d_{1U} \\ \vdots \qquad\qquad \vdots \\ d_{m1}, d_{m2}, \ldots, d_{mU} \end{bmatrix}_{m\times U} \tag{8}$$

where the element d_{ij} denotes the Euclidean distance between the predicted observation $\breve{\boldsymbol{z}}_i$ and the observation $\tilde{\boldsymbol{z}}_j$, i.e.,

$$d_{ij} = \|\breve{\boldsymbol{z}}_i - \tilde{\boldsymbol{z}}_j\|. \tag{9}$$

Distances in each row of the matrix D_{sur} are sorted into a new matrix E

$$E = \begin{bmatrix} e_{11}, e_{12}, \ldots, e_{1U} \\ \vdots \qquad\qquad \vdots \\ e_{m1}, e_{m2}, \ldots, e_{mU} \end{bmatrix}_{m\times U} \tag{10}$$

with an accompanying matrix A_{cc} which is presented by

$$A_{cc} = \begin{bmatrix} a_{11}, a_{12}, \ldots, a_{1U} \\ \vdots \qquad\qquad \vdots \\ a_{m1}, a_{m2}, \ldots, a_{mU} \end{bmatrix}_{m\times U} \tag{11}$$

where a_{ij} records the index of the observation that generates the sorted distance e_{ij}. Interested readers can refer to [34] for the design of the sorting algorithms and hardware circuits. In (10), the relationship $e_{i,1} \leqslant e_{i,2} \leqslant \cdots \leqslant e_{i,U}$ holds for all $1 \leqslant i \leqslant m$. Using (10) and (11), M_1 different observations with smaller Euclidean distances are selected while the others are discarded by using Algorithm 1, where $\boldsymbol{z}'_i$ denotes the ith selected observation.

Algorithm 1 Observation Selection for Survival Targets

```
BEGIN:
  01: SET $i = 1$;
  02: FOR $j = 1, 2, \ldots, U$
  03:   SET $Flag(j) = 0$;
  04: END FOR
  05: WHILE $(i <= M_1)$
  06:   FOR $j = 1, 2, \ldots, U$
  07:     FOR $k = 1, 2, \ldots, m$
  08:       IF $(Flag(a_{kj}) == 0)$ THEN
  09:         SET $z'_s = z_{a_{kj}}$;
  10:         SET $Flag(a_{kj}) = 1$;
  11:         SET $i = i + 1$;
  12:       END IF
  13:     END FOR
  14:   END FOR
  15: End WHILE
END;
```

In Algorithm 1, $Flag(j)$ denotes whether the jth observation has been selected (1) or not (0). $Flag(j)$ is set to 0 at the beginning for all j. The WHILE loop is used to control whether M_1 observations have been selected. Within the WHILE loop, the outer FOR loop is used to search the column of the matrix E, while the inner FOR loop is used to search each element in the column. In this algorithm, we select the observation through a column-by-column manner, not by the absolute value of all sorted Euclidean distances. It is because we want to give each survival target the same priority to have a new observation(s) if it does have one.

Observation Selection for Newborn Targets: For the newborn targets, we select the observations based on the information of $\bar{\mathbf{x}}_0$ and $\bar{\mathbf{x}}_1$, where $\bar{\mathbf{x}}_1 = [x_0\ \dot{x}_0\ y_0\ \dot{y}_0]^T + [\dot{x}_0\ 0\ \dot{y}_0\ 0]^T \delta T$ is the one-step state prediction of $\bar{\mathbf{x}}_0$. We calculate two Euclidean distance sets: One contains the distances between the received observations and $H(\bar{\mathbf{x}}_0)$, and the other contains the distances between the received observations and $H(\bar{\mathbf{x}}_1)$. D_{new} is presented by

$$D_{new} = \begin{bmatrix} d_{11}, d_{12}, \ldots, d_{1U} \\ d_{21}, d_{22}, \ldots, d_{2U} \end{bmatrix}_{2\times U} \tag{12}$$

where

$$d_{ij} = \left\| H(\bar{\mathbf{x}}_{i-1}) - \tilde{\boldsymbol{z}}_j \right\|. \tag{13}$$

Since, at each sensing moment, only one newborn target may appear, we select two measurements from the received measurements, i.e., $M_2 = 2$ for the possible newborn targets. Following a similar strategy as that of Algorithm 1 for the survival targets, two observations for the newborn targets can be selected.

OSEP: We analyze the impact of the observation selection by a metric named OSEP P_{ose}, which is the probability that any target-oriented observation is missed given that M observations are selected.

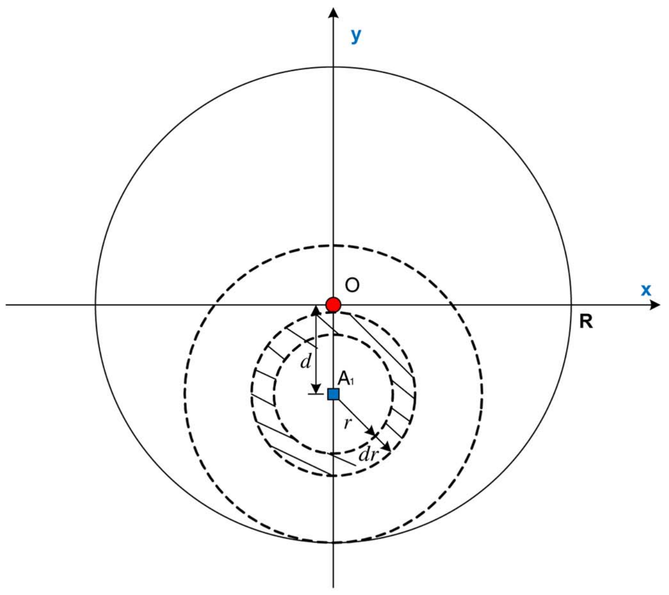
At a sensing moment, the observations are obtained from real targets and clutters as stated in Section II. Supposing that the number of the received observations is U, then the probability that the number of observations from clutter is $X = U - m$ can be presented by

$$P(X = U - m) = \frac{\exp(-\lambda)\lambda^{(U-m)}}{(U-m)!}. \tag{14}$$

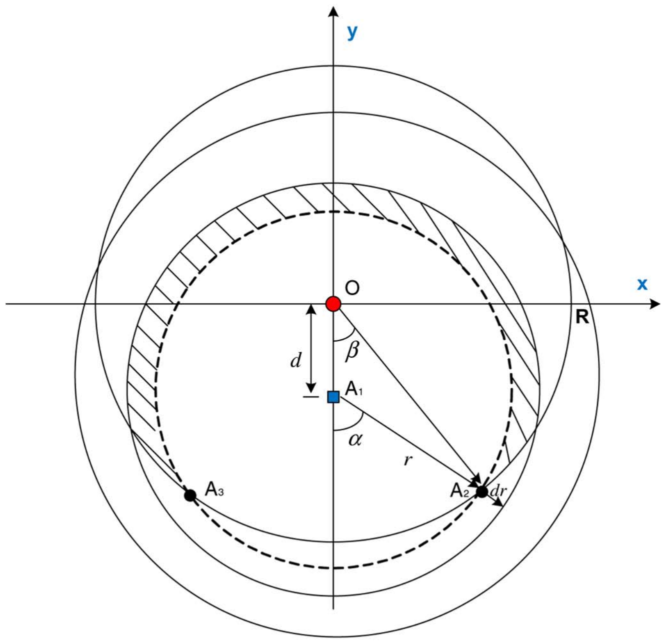
Thus, if M out of U observations are selected by the *Observation Selection* module, the observation selection error occurs when more than $M - m$ clutters are selected.

First, we analyze the probability P_{e1} that any single clutter-oriented observation is selected. The observer selects the observation generated from a clutter other than the one from a real target only when the predicted observation, i.e., the estimated target position, is closer to the one from the clutter than the one from real targets.

Without loss of generality, we consider the predicted observation located at the square point A_1 which is away from the origin O with a distance d, i.e., the coordinate of the point A_1 is $(0, -d)$ where $d \geqslant 0$, as shown in Fig. 3. For the target-oriented observations, from (2), the probability density function



(a)



(b)

Fig. 3. Geometric illustration for evaluation of P_{e1}. (a) Case 1: $r \leqslant R - d$. (b) Case 2: $r \geqslant R - d$.

of the Euclidean distance, r, between the observation from a target to the predicted observation is given by

$$p_t(r) = \frac{1}{\pi\sigma_r^2} \exp\left(-\left[r^2/\sigma_r^2\right]\right). \tag{15}$$

Case 1—$r \leqslant R - d$: As shown in Fig. 3(a), the probability that the Euclidean distance from the clutter and the predicted observation is bigger than a given value r is presented by

$$P_1 = (\pi R^2 - \pi r^2)/\pi R^2 = (R^2 - r^2)/R^2. \tag{16}$$

Thus, in this situation, the probability that the observation generated from targets is closer to the estimated observation than the clutter is

$$P_2 = \int_0^{R-d} 2\pi r p_t P_1 \, dr. \tag{17}$$

Case 2—$r > R - d$: As shown in Fig. 3(b), the coordinates of the intersection A_2 and A_3 of the circle with radius R and the circle with radius r are denoted as (x_{A_2}, y_{A_2}) and (x_{A_3}, y_{A_3}), respectively, where

$$\begin{aligned} x_{A_2} &= -x_{A_3} \\ &= \frac{\sqrt{-r^4 - R^4 - d^4 + 2r^2R^2 + 2r^2d^2 + 2d^2R^2}}{2d} \end{aligned} \tag{18}$$

$$y_{A_2} = y_{A_3} = \frac{r^2 - R^2 - d^2}{2d}. \tag{19}$$

Therefore, the probability that the Euclidean distance between the clutter and the predicted observation is bigger than a given value r is formed as

$$P_3 = \frac{\pi R^2 \frac{\pi-\beta}{\pi} + d x_{A_2} - \pi r^2 \frac{\pi-\alpha}{\pi}}{\pi R^2} = 1 - \frac{\beta}{\pi} + \frac{d x_{A_2}}{\pi R^2} - \frac{r^2}{R^2}\left(1 - \frac{\alpha}{\pi}\right) \tag{20}$$

where

$$\alpha = \arccos\left(\frac{R^2 - r^2 - d^2}{2dr}\right), \quad \alpha \in [0, 2\pi] \tag{21}$$

$$\beta = \arccos\left(\frac{R^2 + d^2 - r^2}{2dR}\right), \quad \beta \in [0, 2\pi]. \tag{22}$$

Moreover, the probability that the observation generated from targets is closer to the estimated observation than the clutter is

$$P_4 = \int_{R-d}^{R+d} 2(\pi - \alpha) r p_t P_3 dr. \tag{23}$$

Then, the probability that one clutter is selected other than the observation from its correlated target is

$$P_5 = 1 - \int_0^{R-d} 2\pi r p_t P_1 dr - \int_{R-d}^{R+d} 2(\pi - \alpha) r p_t P_3 dr. \tag{24}$$

Since all targets are independent with each other, the probability P_{e1} can be formulated as

$$P_{e1} = mP_5. \tag{25}$$

When U observations are received, the observation selection error occurs with the probability

$$P_{e2} = \frac{\exp(-\lambda)\lambda^{(U-m)}}{(U-m)!} \sum_{j=M-m+1}^{M} (P_{e1})^j \tag{26}$$

which includes all the cases when it selects more clutter-oriented observations than expected.

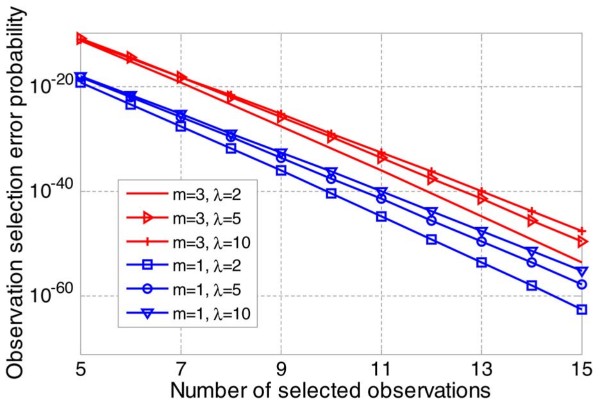


Fig. 4. OSEP under different numbers of selected observations.

To this end, counting all possible values of U, the OSEP is presented by

$$P_{ose} = \sum_{U=M-m+1}^{\infty} \left[\frac{e^{-\lambda}\lambda^{(U-m)}}{(U-m)!} \sum_{j=M-m+1}^{M} (P_{e1})^j \right]. \tag{27}$$

Suppose that the radius of the circular region $R = 200$ and the standard deviations of the observation noise $\sigma_r = 2.5$, as used in the performance evaluations section. It is found that, within the region of $d \leqslant (19/20)R$, wherever the predicted observations are, the relationship $p_{e1} \simeq 0.00032$ holds. (We omit the situation when $d > (19/20)R$ as it means that the targets will soon get out of the observation region.) In this situation, the OSEP is evaluated and shown in Fig. 4 where the maximum number of targets $P = 3$, i.e., $m \leq 3$ at any time. It can be seen that the OSEP decreases in a nearly linear logarithmic manner with the increasing number of the selected observations M. Meanwhile, as the number of real targets m increases, the OSEP also increases. In addition, as the expected clutter number λ in the field increases, the OSEP degrades accordingly. It is concluded that the selection error is negligible as long as the number of selected observations is larger than a boundary value in the studied environment, e.g., $P_{ose} < 10^{-10}$ with $M = 5$ even in the worst case of $m = 3$ and $\lambda = 10$ in Fig. 4. Therefore, the proposed observation selection scheme can be applied to the POSE PHD filter which is suitable for hardware implementation with little loss in the tracking performance.

C. Hardware Implementation

The processing modules of the POSE PHD filter are implemented in the hardware platform accordingly as shown in Fig. 2, namely, *initialization*, *observation selection*, *prediction*, *update*, *resampling*, and *target estimation*.

Specifically, the *initialization* module is responsible for initializing the operating state of the *prediction* module, and it requires no specific hardware processing elements. To implement the particle PHD filters in the hardware, the *prediction* module and the *resampling* module can be combined to further improve the total hardware resource utilization [30]. In our design, the *prediction* module generates $J = 1024$ particles to represent the newborn targets plus another $L = 1024$ particles selected by the *resampling* module which represent the survival

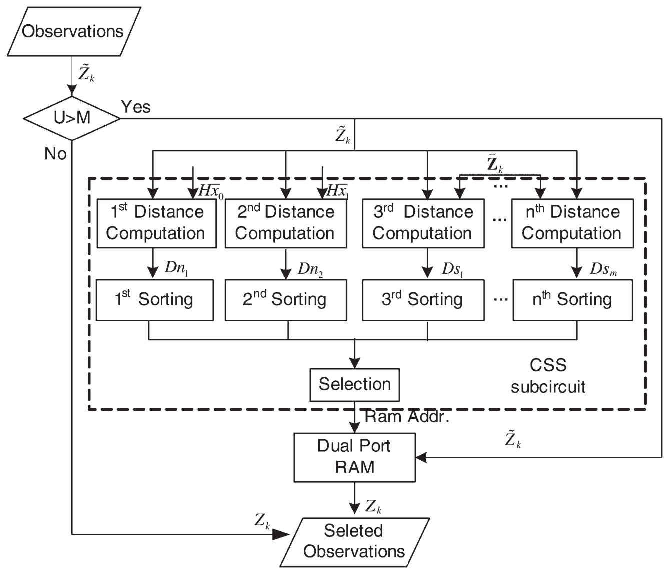


Fig. 5. Hardware structure of *observation selection* module.

targets. We use the K-means cluster algorithm in the *target estimation* module which is pipelined with other processing modules in a Texas Instruments (TI) DSP and contributes little to the real-time filtering performance. Interested readers can refer to [30] for the detailed design of the architecture and the aforementioned modules.

In the following, we focus on the issues related to the implementation of the *observation selection* module and the *update* module, which is specific in the POSE PHD filter.

According to the processing flow shown in Fig. 2, the observations for the POSE PHD filter are first sent to the *observation selection* module. Fig. 5 shows the hardware architecture of the *observation selection* module. These observations are preprocessed in a comparison circuit where the number U of the newly arrived observations is compared with the predefined M. At most, M observations will be sent to the *update* module for the next module processing while the extra observations will be discarded. In case of $U > M$, the observations will be duplicated into two copies. One copy is stored in a dual port RAM, while the other one is sent to the distance computation, sorting, and selection (CSS) subcircuit for future processing.

In the CSS subcircuit, the observations of survival targets and newborn targets are processed by n separate distance computation units by (8) and (12) accordingly. The unit number n is fixed as $n = P + 2$, where P is the maximum number of the targets that appear within the observation region at any time. Thus, the first and second distance computation units compute the distances between the received observations and $H(\bar{\mathbf{x}}_0)$ and $H(\bar{\mathbf{x}}_1)$, respectively. For the remaining P distance computation units, they compute the distances between the observations $\tilde{\boldsymbol{z}}_k$ and $H(\breve{\boldsymbol{x}}_k)$ which are the predicted observations for survival targets. The sorting process is after the distance computation, and interested readers can refer to [34] for the hardware design of the sorting units. The sorting index generated by each distance computation unit is then collected to the selection unit. The number of active distance computation units and sorting units can be adjusted adaptively to the number of currently survival targets.

The CSS subcircuit performs as follows.

Step 1: The first distance computation unit and its corresponding sorting unit are enabled. The index of the observation corresponding to the minimum distance is used as the output of the selection unit. This index will be used as the read address of the dual port RAM unit to obtain the first selected observation.

Step 2: The second distance computation unit and its corresponding sorting unit are enabled. The index of the observation corresponding to the minimum distance is used as the output of the selection unit.

Steps 3, 4, …: The other m out of P distance computation units and the corresponding sorting units are enabled by the number of targets m at time $k - 1$. For example, if there are three targets at time $k - 1$, i.e., $m = 3$, three distance computation units and their sorting units are enabled. Thus, we have three sets of indices to be fed to the selection unit.

The selection follows the algorithm described in Algorithm 1, where one step of the loop corresponds to one clock cycle processing in hardware circuit. The process continues until all M observations are selected. The steps are processed concurrently in the CSS subcircuit. After M indices are selected, they are used as the read address of the dual − port RAM unit to fetch the observations in it and then send the selected observations to the *update* module.

The *update* module updates the weight of each particle according to (6) in a recursive manner by using the selected observations. To update the weight of the ith particle, the *update* module processes the M observations in parallel as shown in Fig. 6.

For each observation, the *update* module first calculates the value of $(\psi_{k,\boldsymbol{z}}(\tilde{\boldsymbol{x}}_{k|k-1}^{(i)}))/(\kappa_k(\boldsymbol{z}) + C_k(\boldsymbol{z}_j))$ individually and then sends the sum of the results from all M subcircuits plus $(1 - p_D)$ to the product unit where it is rescaled by the predicted weight $\hat{w}_{k|k-1}^{(i)}$ from the *prediction* module. Moreover, the output is stored as the updated weight $\tilde{w}_k^{(i)}$. The module will run $L + J$ times to update all particle weights. Specifically, in the subcircuit for the individual observation, the likelihood function $g_k(\boldsymbol{z}_j|\tilde{\boldsymbol{x}}_{k|k-1}^{(i)})$ is computed first, and then, it is multiplied by the probability of detection p_D to obtain $\psi_{k,\boldsymbol{z}}(\tilde{\boldsymbol{x}}_{k|k-1}^{(i)})$, which is stored in a RAM. Meanwhile, the predicted weight $\hat{w}_{k|k-1}^{(i)}$ from the *prediction* module times $\psi_{k,\boldsymbol{z}}(\tilde{\boldsymbol{x}}_{k|k-1}^{(i)})$ and the product is sent to an accumulator (the Acc. unit). Once the products of all particles are collected, $C_k(\boldsymbol{z}_j)$ is obtained by (7). The reciprocal of the sum of $C_k(\boldsymbol{z}_j)$ and $\kappa_k(\boldsymbol{z})$ is calculated which then times $\psi_{k,\boldsymbol{z}}(\tilde{\boldsymbol{x}}_{k|k-1}^{(i)})$. The result of $(\psi_{k,\boldsymbol{z}}(\tilde{\boldsymbol{x}}_{k|k-1}^{(i)}))/(\kappa_k(\boldsymbol{z}) + C_k(\boldsymbol{z}_j))$ calculation is stored in the RAM for the next step of summing up all observations to obtain the ith particle weight.

D. Discussions on Particle Number and Computational Complexity

The performance of particle PHD filters is largely affected by the number of particles. Generally, more particles lead to

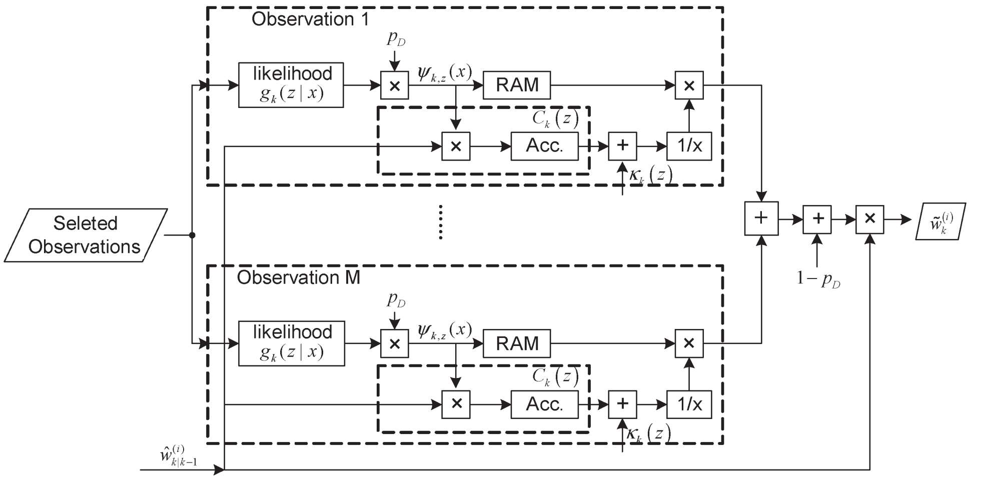


Fig. 6. Hardware structure of *update* module.

more accurate tracking performance at the increasing cost in the hardware units, particularly the RAM resources, proportionally. Meanwhile, the marginal improvement becomes less after the particle number is over a large value. In addition, in the *update* module, the weight computation of all particles produces a significant processing delay. Furthermore, the delay caused by the sequential resampling is proportional to the particle number as in most resampling schemes [35], [36].

Therefore, we make a tradeoff in our design among the expected tracking performance, hardware resource utilization, and time delay. Specifically, proper values of L and J are selected in the proposed POSE PHD filter. We introduce the "lost tracking ratio" as the criterion in the selection of a proper particle number. The lost tracking ratio is defined as the ratio of the number of tracking lost runs and the total number of MC simulation runs. Herein, one tracking lost run means that, in a single simulation run which contains a large number of iterations, the number of targets is wrongly estimated for several (we used four in this paper) consecutive iterations. Generally, the lost tracking ratio will decrease rapidly with the increment of the particle number when the particle number is relatively small, and then, it tends to decrease more and more slowly with the increase of the particle number when it becomes larger enough. Proper values of L and J can be set according to the lost tracking ratio. According to preliminary simulations, we choose $L = 1024$ and $J = 1024$ as the number of particles for use in both the algorithm and hardware design.

In the POSE PHD filter algorithm, the *observation selection* processing is added, and the computational complexity in each recursion becomes $O(mU + (L + J)M)$, while the computational complexity of the traditional particle PHD filter is $O((L + J)U)$. Since $mU + (L + J)M$ can be rewritten as $(L + J)U((m/L + J) + M/U)$, where $m \ll L + J$ and $M < U$ in general, the complexity of the POSE PHD filter is lower than that of the traditional one. Even in the worst case that $M \geq U$, the computational complexities of both the filters are of the same order. In addition, from the hardware implementation complexity perspective, although the observation selection module is added in the POSE PHD filter, which seems to make the hardware implementation more complex, it totally overcomes the difficulty of the implementation of the traditional PHD filter which requires the design of varying number of hardware modules.

IV. Performance Evaluations

In this section, we evaluate the performance of the proposed POSE PHD filter in simulation and hardware experiments, respectively. Specifically, we first compare the tracking performance of the POSE PHD filter with that of the traditional PHD filter. Then, we present the results of implementing the POSE PHD filter in a hardware platform. Finally, we analyze the execution timing of the POSE PHD filter to evaluate the real-time performance.

The parameter settings in the evaluation are shown as follows, if not specified otherwise. The radius of the circular region is $R = 200$, and the observer is located at the center of the region. The dynamic of targets follows (1) where the sensing period $\delta T = 1$ and the independent zero-mean Gaussian white noises $\kappa_{1,k}$ and $\kappa_{2,k}$ have standard deviations $\sigma_{\kappa_1} = 1$ and $\sigma_{\kappa_2} = 0.1$, respectively. The observation equation follows (2) where the standard deviations of the observation noise are $\sigma_r = 2.5$ and $\sigma_\theta = 0.005$. The maximum number of targets in the observation region is $P = 3$ for any sensing moment. The probability of detection is $p_D(x_k) = 1$, i.e., there is no miss detection or detection error. The probability of target survival is $e_{k|k-1} = 0.95$. The expected number of the Poisson distribution for the clutter is $\lambda = 10$. For the intensity function in the Poisson point process

$$\bar{\mathbf{x}}_0 = \begin{bmatrix} 0 \\ 3 \\ 0 \\ -3 \end{bmatrix}, \text{ and } Q = \begin{bmatrix} 10 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

A. Matlab Simulation Results

The comparison of the tracking performance between the POSE PHD filter and the traditional particle PHD filter is shown in Fig. 7. The trajectories estimated by both filters and the real one are shown in Fig. 7(a), and the estimated positions at

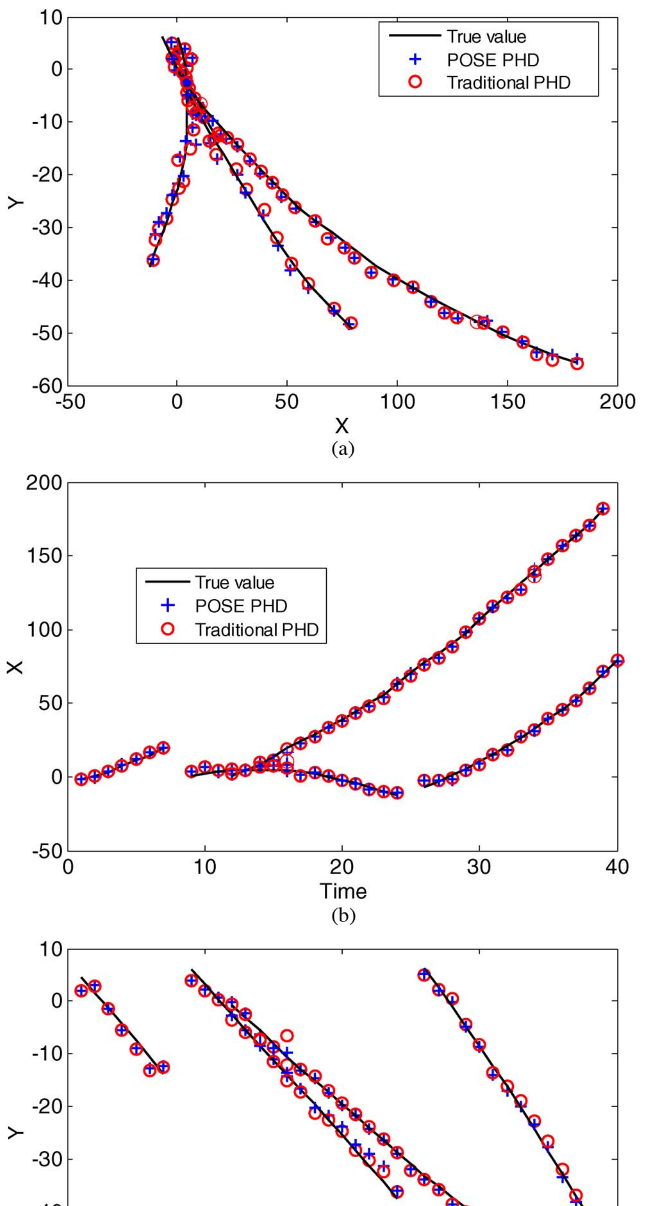


Fig. 7. Target trajectories estimated by the POSE PHD filter and the traditional particle PHD filter. (a) Estimated trajectories and true trajectories. (b) Estimated trajectories and true trajectories in x-axis direction. (c) Estimated trajectories and true trajectories in y-axis direction.

different times in the x- and y-axes are shown in Fig. 7(b) and (c), respectively. It is seen that both the POSE PHD filter and the traditional PHD filter have a good tracking performance even when the targets overlap in some region. Although the proposed POSE PHD filter reduces the observation set to a fixed number, e.g., $M = 8$ in the simulation, it has achieved approximately the same level of tracking performance as the traditional particle PHD filter.

To further quantify the tracking performance, we study the estimated number of targets, their multitarget miss distance, and

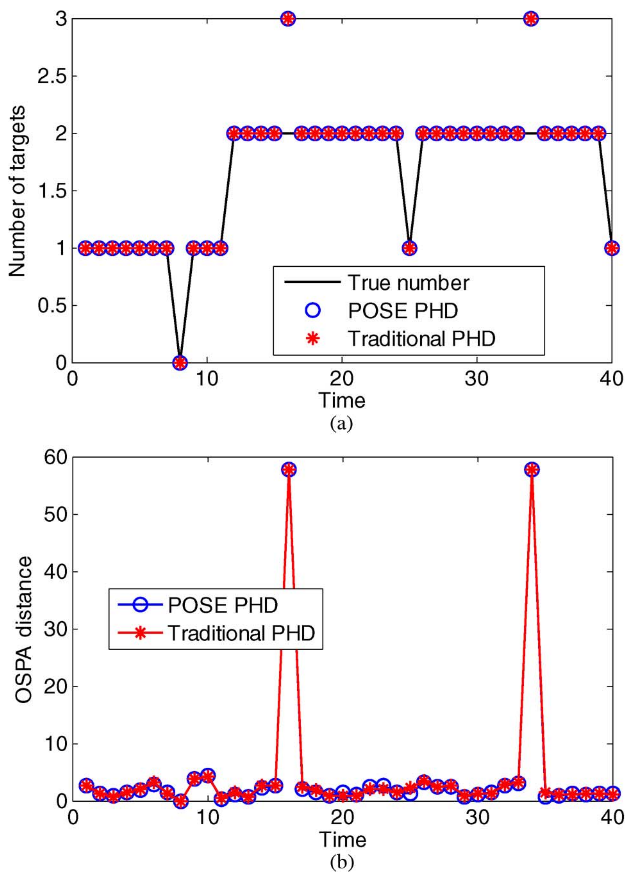


Fig. 8. Estimated number of targets and multitarget miss distance by the POSE PHD filter and the traditional PHD filter. (a) Estimated number of targets and the true number of targets. (b) Multitarget miss distance.

the lost tracking ratio in the following. The optimal subpattern assignment (OSPA) [37] is used as the multitarget miss distance metric, and the parameters in it are set as $p = 2$ and $c = 20$ in our evaluation.

Fig. 8(a) shows the estimation of the number of targets by the two filters along with the true values. It is observed that, under the same simulation conditions, the POSE PHD filter and the traditional one achieve the same results in the target number estimation. Specifically, at the time indices of 16 and 34, both make a wrong estimation of the target number, while for the rest of the time indices, they correctly estimate the target number. Fig. 8(b) shows the OSPA distance of the two filters. The OSPA distance jointly captures the differences in cardinality and individual elements between two finite sets in a mathematically consistent yet intuitively meaningful way. Again, from this figure, it is seen that the POSE PHD filter has a similar performance as the traditional particle PHD filter. Specifically, when the estimated number of targets is correct, both filters have small OSPA distances, approximately under the value of 5. However, the OSPA distance tends to deteriorate if the true target set and the estimated target set are of different cardinalities, which occurs when the estimated target number is incorrect.

Previous results are obtained in a single simulation run, e.g., Fig. 8. In Table II, we study the lost tracking ratio parameter in multiple runs of simulation and show the tracking lost times recorded in 5000 simulation runs for the POSE PHD filter and the traditional one. The proposed POSE PHD filter has

TABLE II
5000 MC SIMULATIONS OF TWO PHD FILTERS

Filter	Number of tracking lost	lost tracking ratio
Traditional PHD	556	11.12%
POSE PHD	578	11.56%

578 tracking lost times while the traditional particle PHD filter has 556 times, with lost tracking ratios of 11.12% and 11.56%, respectively. It further confirms our statement that the POSE PHD algorithm has approximately the same level of tracking performance as the traditional one.

B. Hardware Experiment Results

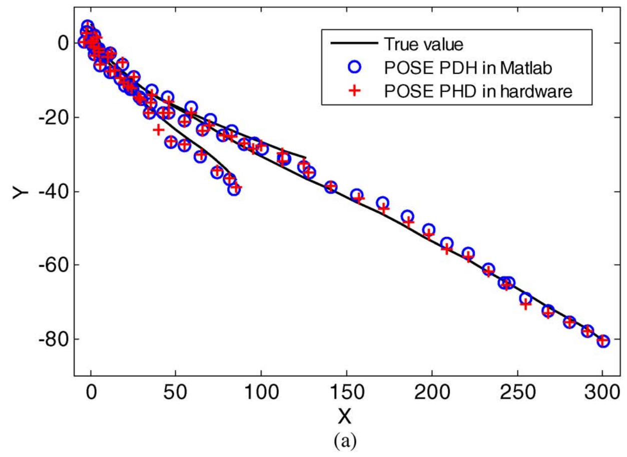
The hardware design of the POSE PHD filter is implemented in a Xilinx Virtex-II Pro FPGA platform with the main chip XC2VP700. The hardware design was described in Verilog hardware description language and verified by using the Xilinx on-chip debug tool "Chipscope Pro." In the hardware resource budgeting, we set the word length of position to 21 b with 1 b for sign, 9 b for integer, and 11 b for decimal fraction. The word length of velocity is 17 b with 1 b for sign, 5 b for integer, and 11 b for decimal fraction. The word length of weight is 16 b with 1 b for integer and 15 b for decimal fraction. The results from "Chipscope Pro" are fed to a TI DSP for clustering to obtain the target states.

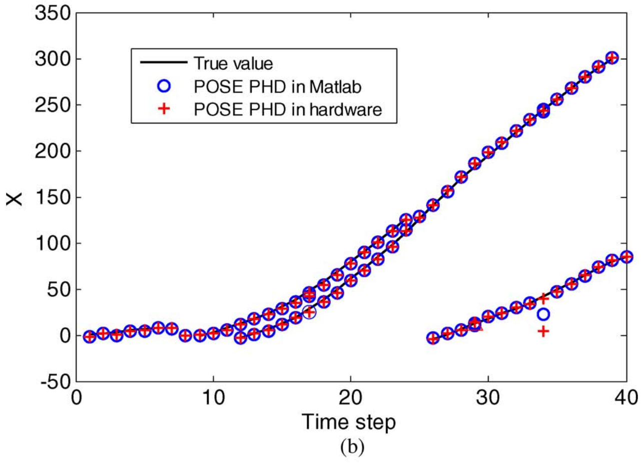
We compare the hardware experimental results with the ones obtained from Matlab simulation as shown in Fig. 9 based on the same set of observations. The estimated trajectories are shown in Fig. 9(a) along with the real target trajectories, and the estimated positions at different sensing times in the x- and y-axes are shown in Fig. 9(b) and (c), respectively. It is seen that the experiment results from the "Chipscope Pro" are very close to the ones from simulation, and both are very close to the true value. This illustrates that the hardware implementation of the POSE PHD filter can have the same level of tracking performance as the algorithm simulation.

For quantitative comparison, the number of targets and multitarget miss distance estimated by the hardware circuit and the Matlab simulation are shown in Fig. 10. During the period of target dynamic in Fig. 10(a), the results from both the hardware circuit and the Matlab simulation return the accurate estimation of the real target movement with only a few discrete failures. In Fig. 10(b), the OSPA distances are plotted. It can bee seen that the POSE PHD filter implemented in hardware shows approximately the same level of OSPA performance as that simulated in Matlab with only very slight difference, which is caused by the finite word length effect of the hardware circuit. From the aforementioned quantitative comparison, it is seen that the hardware implementation of the POSE PHD filter can be applied to some real-world MTT scenarios.

C. Real-Time Performance Analysis

Next, we assess the real-time performance of the propose POSE PHD filter implementation. Fig. 11 depicts the execution time for one recursion of the proposed POSE PHD filter, where the *target estimation (clustering)* module is not included since it can be implemented in a pipelined manner with the





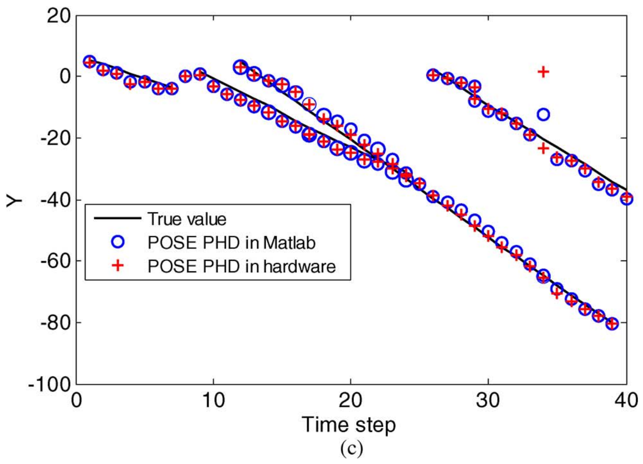


Fig. 9. Target trajectories by the hardware circuit and Matlab simulation of the proposed POSE PHD filter. (a) Estimated trajectories by hardware circuit and Matlab simulation along with true trajectories. (b) Estimated position by hardware circuit and Matlab simulation along with true trajectories in x-axis direction. (c) Estimated position by hardware circuit and Matlab simulation along with true trajectories in y-axis direction.

prediction of the next recursion and thus does not affect the total processing speed. The total cycle time is $T_{POSE} = (L_p + L_{update} + L_{res})T_{clk}$, where L_p is the latency of *prediction* preparation and *prediction*, L_{update} is the latency of *update*, L_{res} is the number of cycles needed for *resampling*, and T_{clk} is the cycle time of the system clock.

The latency of *update* in the POSE PHD filter is mainly caused by the calculation of $C_k(z)$. All weights of the 2048

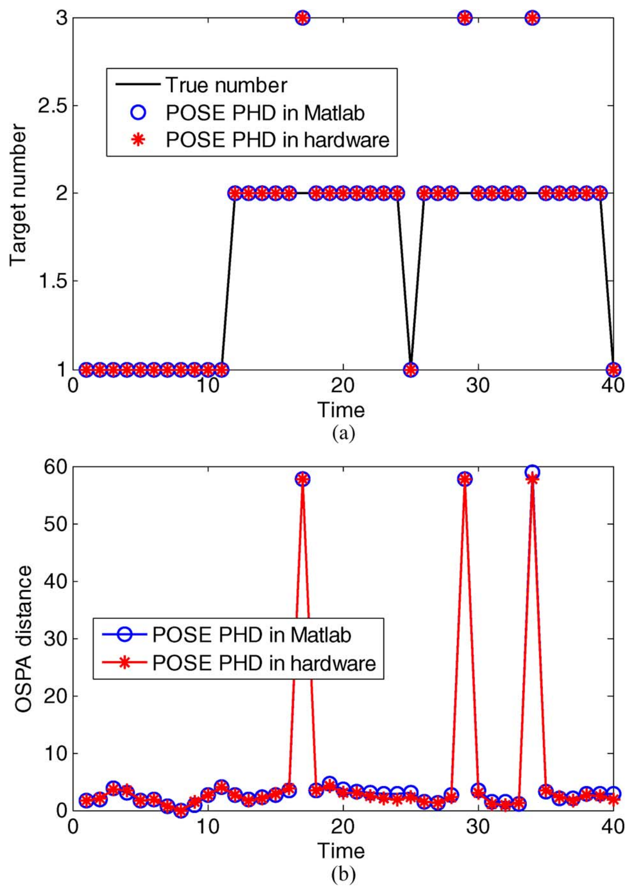


Fig. 10. Estimated number of targets and multitarget miss distance of the POSE PHD filter implemented in hardware and in Matlab. (a) Estimated number of targets versus the true number of targets. (b) Multitarget miss distance.

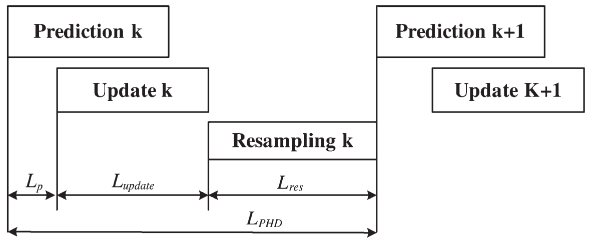


Fig. 11. Timing of the POSE PHD filter.

particles are required to compute $C_k(z)$ in (7), which results in a latency of 2048 cycles. As the systematic resampling (SR) is adopted [35], the sum of updated weights must be computed before *resampling*, which requires another 2048-cycle latency. Thus, we have $L_{update} = (2048 + 2048 + L_{other})$ cycle latency, where L_{other} includes the exponential, division, multiplication, sum, subtraction, and registration latency. In the proposed structure, $L_{other} = 78$, so we have $L_{update} = 4174$.

The *resampling* uses the SR and is supposed to have $(p + q - 1)T_{clk}$ latency, where p is the number of all particles before resampling and q is the number of particles after resampling. Here, $p = 2048$, and $q = 1024$. In addition, another four cycles are needed for auxiliary operation, so totally, we have $L_{res} = 3075L_{clk}$ cycle latency. After resampling, it requires six cycles to prepare for sensing at the next moment; thus, we have $L_p = 6$.

The total time of one recursion of the POSE PHD filter implementation besides the *target estimation (clustering)* module equals

$$\begin{aligned} T_{POSE} &= (L_p + L_{update} + L_{res})T_{clk} \\ &= (6 + 4174 + 3075)T_{clk} \\ &= 7255T_{clk}. \end{aligned}$$

The Xilinx Synthesis Technology (XST) tool shows that the hardware design can support clock frequencies up to 72 MHz with the FPGA chip XC2VP70. When using a system clock frequency of 50 MHz, the proposed design can achieve a processing speed $1/T_{POSE} = 6.892$ kHz, which can be further improved by using more fast speed grade FPGA.

V. Conclusion

In this paper, we have proposed a novel MTT particle PHD filter with observation selection for practical hardware implementation. Simulation results have shown that the proposed POSE PHD filter achieves the same level of tracking performance as the traditional particle PHD filter. Furthermore, the feasibility of implementing the POSE PHD filter in hardware is verified on an FPGA platform, and the real-time performance is evaluated. For the future work, we will formulate the utilization of hardware resources together with the real-time performance issues into a joint optimization problem, to further improve the implementation of PHD filters for real-time applications. In addition, we will extend the proposed observation selection scheme to the implementation of multiple variations/developments of PHD filters such as the CPHD filter and the MeMBer filter.

References

[1] M. Alam and A. Bal, "Improved multiple target tracking via global motion compensation and optoelectronic correlation," *IEEE Trans. Ind. Electron.*, vol. 54, no. 1, pp. 522–529, Feb. 2007.

[2] O. Gerelli and C. G. Lo Bianco, "Nonlinear variable structure filter for the online trajectory scaling," *IEEE Trans. Ind. Electron.*, vol. 56, no. 10, pp. 3921–3930, Oct. 2009.

[3] Z. Wang and D. Gu, "Cooperative target tracking control of multiple robots," *IEEE Trans. Ind. Electron.*, vol. 59, no. 8, pp. 3232–3240, Aug. 2012.

[4] P. Cheng, J. Chen, F. Zhang, Y. Sun, and X. Shen, "A distributed TDMA scheduling algorithm for target tracking in ultrasonic sensor networks," *IEEE Trans. Ind. Electron.*, vol. 60, no. 9, pp. 3836–3845, Sep. 2013.

[5] M.-S. Lee and Y.-H. Kim, "An efficient multitarget tracking algorithm for car applications," *IEEE Trans. Ind. Eletron.*, vol. 50, no. 2, pp. 397–399, Apr. 2003.

[6] Y. Bi, L. Cai, X. Shen, and H. Zhao, "Efficient and reliable broadcast in intervehicle communication networks: A cross-layer approach," *IEEE Trans. Veh. Technol.*, vol. 59, no. 5, pp. 2404–2417, Jun. 2010.

[7] Y. Chen, B. Wu, H. Huang, and C. Fan, "A real-time vision system for nighttime vehicle detection and traffic surveillance," *IEEE Trans. Ind. Electron.*, vol. 58, no. 5, pp. 2030–2044, May 2011.

[8] J. Scharcanski, A. de Oliveira, P. Cavalcanti, and Y. Yari, "A particle-filtering approach for vehicular tracking adaptive to occlusions," *IEEE Trans. Veh. Technol.*, vol. 60, no. 2, pp. 381–389, Feb. 2011.

[9] S. Haykin, W. Stehwien, C. Deng, P. Weber, and R. Mann, "Classification of radar clutter in an air traffic control environment," *Proc. IEEE*, vol. 79, no. 6, pp. 742–772, Jun. 1991.

[10] R. Mahler, "Multitarget Bayes filtering via first-order multitarget moments," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 39, no. 4, pp. 1152–1178, Oct. 2003.

[11] B. N. Vo, S. Singh, and A. Doucet, "Sequential Monte Carlo methods for multitarget filtering with random finite sets," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 41, no. 4, pp. 1224–1245, Oct. 2005.

[12] B. N. Vo and W. Ma, "The Gaussian mixture probability hypothesis density filter," *IEEE Trans. Signal Process.*, vol. 54, no. 11, pp. 4091–4104, Nov. 2006.

[13] B. Ristic, D. Clark, B. N. Vo, and B. T. Vo, "Adaptive target birth intensity for PHD and CPHD filters," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 48, no. 2, pp. 1656–1668, Apr. 2012.

[14] R. Mahler, "PHD filters of higher order in target number," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 43, no. 4, pp. 1523–1543, Oct. 2007.

[15] B. T. Vo, B. N. Vo, and A. Cantoni, "The cardinality balanced multi-target multi-Bernoulli filter and its implementations," *IEEE Trans. Signal Process.*, vol. 57, no. 2, pp. 409–423, Feb. 2009.

[16] Y. Zheng, Z. Shi, R. Lu, S. Hong, and X. Shen, "An efficient data-driven particle PHD filter for multi-target tracking," *IEEE Trans. Ind. Informat.*, to be published.

[17] M. 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, Feb. 2002.

[18] J. Lim and D. Hong, "Cost reference particle filtering approach to high-bandwidth tilt estimation," *IEEE Trans. Ind. Electron.*, vol. 57, no. 11, pp. 3830–3839, Nov. 2010.

[19] Z. Liang, S. Feng, D. Zhao, and X. Shen, "Delay performance analysis for supporting real-time traffic in a cognitive radio sensor network," *IEEE Trans. Wireless Commun.*, vol. 10, no. 1, pp. 325–335, Jan. 2011.

[20] C. Chang and H. Lie, "Real-time visual tracking and measurement to control fast dynamics of overhead cranes," *IEEE Trans. Ind. Electron.*, vol. 59, no. 3, pp. 1640–1649, Mar. 2012.

[21] C. Lee, S. Lin, C. Lee, and C. Yang, "An efficient camera hand-off filter in real-time surveillance tracking system," *Int. J. Innov. Comput., Inf. Control*, vol. 8, no. 2, pp. 1397–1417, Feb. 2012.

[22] L. Wu, P. Shi, H. Gao, and C. Wang, "H∞ filtering for 2D Markovian jump systems," *Automatica*, vol. 44, no. 7, pp. 1849–1858, Jul. 2008.

[23] L. Wu and D. Ho, "Fuzzy filter design for Itô stochastic systems with application to sensor fault detection," *IEEE Trans. Fuzzy Syst.*, vol. 17, no. 1, pp. 233–242, Feb. 2009.

[24] S. Hong, Z. Shi, and K. Chen, "Easy-hardware-implementation MMPF for maneuvering target tracking: Algorithm and architecture," *J. Signal Process. Syst.*, vol. 61, no. 3, pp. 259–269, Dec. 2010.

[25] L. Miao, J. Zhang, C. Chakrabarti, and A. Papandreou-Suppappola, "Algorithm and parallel implementation of particle filtering and its use in waveform-agile sensing," *J. Signal Process. Syst.*, vol. 65, no. 2, pp. 211–227, Nov. 2011.

[26] X. Su, P. Shi, L. Wu, and Y. Song, "A novel approach to filter design for T-S fuzzy discrete-time systems with time-varying delay," *IEEE Trans. Fuzzy Syst.*, vol. 20, no. 6, pp. 1114–1129, Dec. 2012.

[27] C. Lin, A. Ting, C. Hsu, and C. Chung, "FPGA-based robust adaptive control of BLDC motors using fuzzy cerebellar modal articulation controller," *Int. J. Innov. Comput., Inf. Control*, vol. 8, no. 5(A), pp. 3411–3429, May 2012.

[28] M. Bolić, A. Athalye, S. Hong, and P. Djurić, "Study of algorithmic and architectural characteristics of Gaussian particle filters," *J. Signal Process. Syst.*, vol. 61, no. 2, pp. 205–218, Nov. 2010.

[29] S. Hong, Z. Shi, J. Chen, and K. Chen, "A low-power memory-efficient resampling architecture for particle filters," *Circuits, Syst., Signal Process.*, vol. 29, no. 1, pp. 155–167, Feb. 2010.

[30] S. Hong, L. Wang, Z. Shi, and K. Chen, "Simplified particle PHD filter for multiple-target tracking: Algorithm and architecture," *Progr. Electromagn. Res.*, vol. 120, pp. 481–498, Nov. 2011.

[31] L. Miao, J. Zhang, C. Chakrabarti, A. Papandreou-Suppappola, and N. Kovvali, "Real-time closed-loop tracking of an unknown number of neural sources using probability hypothesis density particle filtering," in *Proc. IEEE Workshop SiPS*, 2011, pp. 367–372.

[32] Z. Shi, Y. Zheng, X. Bian, and Z. Yu, "Threshold-based resampling for high-speed particle PHD filter," *Progr. Electromagn. Res.*, vol. 136, pp. 369–383, 2013.

[33] M. Gupta, L. Jin, and N. Homma, *Static and Dynamic Neural Networks*. Hoboken, NJ, USA: Wiley, 2003.

[34] K. Batcher, "Sorting networks and their applications," in *Proc. Spring Joint Comput. Conf.*, 1968, pp. 307–314.

[35] A. Athalye, M. Bolic, S. Hong, and P. Djuric, "Generic hardware architectures for sampling and resampling in particle filters," *EURASIP J. Appl. Signal Process.*, vol. 2005, no. 17, pp. 2888–2902, Jan. 2005.

[36] A. Sankaranarayanan, A. Srivastava, and R. Chellappa, "Algorithmic and architectural optimizations for computationally efficient particle filtering," *IEEE Trans. Image Process.*, vol. 17, no. 5, pp. 737–748, May 2008.

[37] D. Schuhmacher, B. T. Vo, and B. N. Vo, "A consistent metric for performance evaluation of multi-object filters," *IEEE Trans. Signal Process.*, vol. 56, no. 8, pp. 3447–3457, Aug. 2008.

Zhiguo Shi (M'10) received the B.S. and Ph.D. degrees in electronic engineering from Zhejiang University, Hangzhou, China, in 2001 and 2006, respectively.

From 2006 to 2009, he was an Assistant Professor with the Department of Information and Electronic Engineering, Zhejiang University, where he is currently an Associate Professor. From September 2011, he began a two-year visit to the Broadband Communications Research Group, University of Waterloo, Waterloo, ON, Canada. His research interests include radar data and signal processing, wireless communication, and security.

Dr. Shi was the recipient of the Best Paper Award of the IEEE Wireless Communications and Networking Conference 2013 in Shanghai, China, and the Best Paper Award of the IEEE Wireless Communications and Signal Processing 2012 in Huangshan, China. He was the recipient of the Scientific and Technological Award of Zhejiang Province, China, in 2012. He serves as an Editor of KSII Transactions on Internet and Information Systems. He also serves as a Technical Program Committee member for the IEEE Vehicular Technology Conference 2013 Fall, IEEE International Conference on Communications in China 2013, Mobile Ad-hoc and Sensor Networks 2013, IEEE International Conference on Computer Communications 2014, IEEE International Conference on Computing, Networking and Communications 2014, etc.

Yongkang Liu is currently working toward the Ph.D. degree in the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada.

He is currently a Research Assistant with the Broadband Communications Research Group, University of Waterloo. His general research interests include protocol analysis and resource management in wireless communications and networking, with special interest in spectrum- and energy-efficient wireless communication networks.

Mr. Liu was the recipient of the Best Paper Award from the IEEE GLOBECOM 2011 in Houston, TX, USA.

Shaohua Hong (M'12) received the B.Sc. degree in electronics and information engineering from Zhejiang University, Hangzhou, China, in 2005 and the Ph.D. degree in electronics science and technology from Zhejiang University, Hangzhou, in 2010.

He is currently an Assistant Professor with the Department of Communication Engineering, Xiamen University, Xiamen, China. His research interests include image processing, joint source and channel coding, Bayesian target tracking, and nonlinear signal processing.

Jiming Chen (M'08–SM'11) received the B.Sc. and Ph.D. degrees in control science and engineering from Zhejiang University, Hangzhou, China, in 2000 and 2005, respectively.

He was a Visiting Researcher at the Institut National de Recherche en Informatique et en Automatique in 2006, the National University of Singapore, Singapore, in 2007, and the University of Waterloo, Waterloo, ON, Canada, from 2008 to 2010. He is currently a Full Professor with the Department of Control Science and Engineering, the Coordinator of the Group of Networked Sensing and Control in the State Key Laboratory of Industrial Control Technology, and the Vice-Director of the Institute of Industrial Process Control at Zhejiang University.

Dr. Chen currently serves an Associate Editor for several international Journals, including the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEM, IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, IEEE NETWORK, *IET Communications*, etc. He was a Guest Editor of the IEEE TRANSACTIONS ON AUTOMATIC CONTROL, *Computer Communication* (Elsevier), *Wireless Communication and Mobile Computer* (Wiley), and *Journal of Network and Computer Applications* (Elsevier). He also served/serves as an Ad Hoc and Sensor Network Symposium Cochair of the IEEE Globecom 2011, a General Symposium Cochair of the Association for Computing Machinery (ACM) International Wireless Communications and Mobile Computing (IWCMC) 2009 and ACM IWCMC 2010, an International Wireless Internet Conference (WiCON) 2010 Medium Access Control (MAC) Track Cochair, an IEEE International Conference on Mobile Ad hoc and Sensor Systems (MASS) 2011 Publicity Cochair, an IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS) 2011 Publicity Cochair, an IEEE International Conference on Distributed Computing Systems (ICDCS) 2012 Publicity Cochair, an IEEE ICCC 2012 Communications Quality of Service and Reliability Symposium Cochair, an IEEE SmartGridComm The Whole Picture Symposium Cochair, an IEEE MASS 2013 Local Chair, a Wireless Networking and Applications Symposium Cochair, and a TPC member for the IEEE ICCC 2013, IEEE ICDCS'10,'12,'13, IEEE MASS'10,'11,'13, IEEE International Conference on Sensing, Communication, and Networking (SECON'11),'12, IEEE INFOCOM'11,'12,'13.

Xuemin (Sherman) Shen (M'97–SM'02–F'09) received the B.Sc. degree in electrical engineering from Dalian Maritime University, Dalian, China, in 1982 and the M.Sc. and Ph.D. degrees in electrical engineering from Rutgers University, Newark, NJ, USA, in 1987 and 1990, respectively.

He is a Professor and the University Research Chair of the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. He was the Associate Chair for Graduate Studies from 2004 to 2008. His research focuses on resource management in interconnected wireless/wired networks, wireless network security, wireless body area networks, and vehicular ad hoc and sensor networks. He is a coauthor/editor of six books and has published more than 600 papers and book chapters in wireless communications and networks, control, and filtering.

Dr. Shen served as the Technical Program Committee Chair for the IEEE VTC'10 Fall, the Symposium Chair for the IEEE ICC'10, the Tutorial Chair for the IEEE VTC'11 Spring and IEEE ICC'08, the Technical Program Committee Chair for the IEEE Globecom'07, the General Cochair for Chinacom'07 and QShine'06, and the Chair for the IEEE Communications Society Technical Committee on Wireless Communications and on P2P Communications and Networking. He also serves/served as the Editor-in-Chief for the IEEE NETWORK, Peer-to-Peer Networking and Application, and IET Communications; a Founding Area Editor for the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS; an Associate Editor for the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, Computer Networks, ACM/Wireless Networks, etc.; and the Guest Editor for the IEEE JSAC, IEEE Wireless Communications, IEEE COMMUNICATIONS MAGAZINE, and ACM Mobile Networks and Applications, etc. He was the recipient of the Excellent Graduate Supervision Award in 2006 and the Outstanding Performance Award in 2004, 2007, and 2010 from the University of Waterloo, the Premier's Research Excellence Award in 2003 from the Province of Ontario, Canada, and the Distinguished Performance Award in 2002 and 2007 from the Faculty of Engineering, University of Waterloo. He is a Registered Professional Engineer of Ontario, Canada, a fellow of the Engineering Institute of Canada, a fellow of the Canadian Academy of Engineering, and a Distinguished Lecturer of the IEEE Vehicular Technology Society and Communications Society.