State Estimation in Nonlinear System Using Sequential Evolutionary Filter

Shen Yin, *Senior Member, IEEE*, Xiangping Zhu, *Student Member, IEEE*,
Jianbin Qiu, *Senior Member, IEEE*, and Huijun Gao, *Fellow, IEEE*

***Abstract*—As a commonly encountered problem in the particle filters (PFs), the particle impoverishment is caused partially by the reduction of particle diversity after resampling. In this paper, a novel particle filtering technique named sequential evolutionary filter (SEF) is introduced, by which the particle impoverishment problem can be effectively mitigated. SEF is proposed based on the genetic algorithm (GA). A GA-inspired strategy is designed and incorporated in SEF. With this strategy, the resampling used in most of the existing PFs is not necessary, and the particle diversity can be maintained. The experimental results also demonstrate the effectiveness of SEF.**

***Index Terms*—Genetic algorithm (GA), nonlinear system, particle filter (PF), sequential evolutionary filter (SEF), state estimation.**

I. Introduction

ACCURATE estimation of the system states is important for many practical applications, for example, process monitoring [1]–[4], target tracking [5]–[7], robot navigation [8], [9], controller design [10], [11], etc. From the Bayesian perspective, the posterior distribution of the system states contains the information from both the system model and sensor measurements. With the obtained posterior distribution, the system states can be easily estimated by certain methods, e.g., minimum mean-squared error (MMSE) and maximum *a posteriori* (MAP). Starting from this observation, many nonlinear filtering techniques have been developed under the Bayesian framework to derive the state posterior distribution. The extended Kalman filter (EKF) is the most commonly used nonlinear filter. It deals with the nonlinearity of the considered system with the linear approximation around the operation point at each time instant. The unscented Kalman filter (UKF) is also a nonlinear extension of the Kalman filter. Based on the unscented transformation, it approximates the posterior distribution of the system states more accurately than EKF. The effectiveness of EKF and UKF has been demonstrated in many literatures and lots of the improved versions have been proposed [10], [12], [13]. However, these nonlinear filters have the difficulties in dealing with the arbitrary distributions in the considered system.

The particle filter (PF) provides another effective tool for the state estimation problem of nonlinear systems. Under the Bayesian framework, the Monte Carlo simulation method is incorporated in PF. The posterior distribution is approximated by a number of weighted samples (or particles). Compared with EKF and UKF, PF offers several superior performances. For example, it can be used in the system with non-Gaussian distributions. And for the highly nonlinear system, it can generate more accurate state estimations than EKF and UKF. In the past two decades, PF has found a wide variety of applications in the practical problems. In [14], both the unconstrained and constrained state estimation problems using PF have been discussed. Das *et al.* proposed a visual tracking algorithm in [15] based on PF, and its effectiveness has been demonstrated in various face and vehicle tracking tasks. The topic of fault diagnosis and prognosis based on PF can be found in [16] and [17]. Besides, in the robotics community, PF has been studied in navigation or localization [8], [9], tracking [5], [7], health management [18], etc.

However, PF still suffers from several problems. Particle impoverishment is one of the commonly encountered problems. In PF, the resampling procedure is incorporated to replicate the large-weight particles to replace the small-weight ones, which are useless for approximating the posterior distribution. This results in the reduction of particle diversity, which is one of the reasons for the particle impoverishment problem. A simple strategy for mitigating this problem is to rough the over-centralized particles with Gaussian jitter noise [19]. Besides, in [20], a risk-sensitive PFs (RSPF) was proposed for mitigating particle impoverishment by constructing explicit risk functions. On the other hand, the filtering problem can also be treated as the sequential optimization problem. Tomoyuki revealed that PF and genetic algorithm (GA) share the similar structure [21]. Based on these observations, many efforts have also been devoted to mitigating the particle impoverishment problem with GA. In [21], Tomoyuki proposed a GA filter (GAF). Kwok *et al.* introduced an evolutionary PF for the prevention of particle impoverishment [22]. Yin *et al.* proposed an intelligent PF (IPF) by incorporating the genetic operators in PF [23]. In [24], the real-coded GA PFs have been proposed for solving the high-dimensional state-space problem. However, for most of these existing GA-based PFs, the proposed strategies often act as the additional part to be added in the generic PF. Compared

Manuscript received August 12, 2015; revised November 13, 2015 and December 28, 2015; accepted January 1, 2016. Date of publication January 27, 2016; date of current version May 10, 2016. This work was supported in part by the National Natural Science Foundation of China under Grant 61472104 and Grant 61304102, in part by the Fundamental Research Funds for the Central Universities under Grant HIT.BRETIV.201507, and in part by the Scientific Research Foundation for Returned Scholars of Heilongjiang Province under Grant LC2015023. *(Corresponding author: Xiangping Zhu.)*

The authors are with Harbin Institute of Technology, Harbin 150001, China (e-mail: shen.yin2011@googlemail.com; xiangping.zhu2010@gmail.com; jbqiu@hit.edu.cn; hjgao@hit.edu.cn).

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.2016.2522382

with the generic PF, although the particle impoverishment problem has been mitigated to some degree, most of these improved PFs have the more complex strategies and suffer from higher computational burdens. Besides the GA-based PF, there are also some other evolutionary inspired PFs. More information can be found in [6] and [25].

In this paper, a novel filter named sequential evolutionary filter (SEF) is proposed for mitigating the particle impoverishment. It is inspired from the idea of GA. The reduction of particle diversity introduced by resampling can be attributed to two reasons. The first one is the existence of too many replicates of large-weight particles. Second, if no particles located in the high probability regions of the posterior distribution, no resampled particles will be generated in these regions. Based on these two reasons, a GA-inspired strategy is designed in SEF. With this strategy, the particle diversity can be maintained. The small-weight particles can be modified into the particles assigned with larger weights. And at the same time, these modified small-weight particles will not overlap the large-weight ones. Different from most of the existing GA-based PFs, the resampling procedure is not necessary in SEF and is replaced by the proposed GA-inspired strategy. The effectiveness of SEF is demonstrated in the numerical example and three-tank system.

This paper is organized as follows. In Section II, the brief introductions of the generic PF and GA are presented. Section III presents the detailed descriptions of the proposed SEF. The results of the experiments on the numerical example and three-tank system are reported in Section IV. Finally, Section V concludes this paper.

II. Preliminaries On Generic PF (GPF) and GA

In this paper, the considered dynamic state-space model is

$$\begin{cases} \boldsymbol{x}_k = \mathrm{f}(\boldsymbol{x}_{k-1}, \boldsymbol{u}_k, \mathrm{w}_k) \\ \boldsymbol{y}_k = \mathrm{h}(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{\nu}_k), \qquad k = 1, 2, \ldots \end{cases} \tag{1}$$

where

$\boldsymbol{x}_k \in \mathbb{R}^{n_x}$ and $\boldsymbol{y}_k \in \mathbb{R}^{n_y}$ denote the state and measurement vectors;

$\boldsymbol{u}_k \in \mathbb{R}^{n_u}$ is the system input;

$\boldsymbol{w}_k \in \mathbb{R}^{n_w}$ and $\boldsymbol{\nu}_k \in \mathbb{R}^{n_\nu}$ denote the white process and measurement noises.

The functions $\mathrm{f} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_w} \to \mathbb{R}^{n_x}$ and $\mathrm{h} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_\nu} \to \mathbb{R}^{n_y}$ are the transition and measurement functions. The system model (1) can also be formulated in the statistical form, where the functions f and h, respectively, correspond to the transition and likelihood distributions, i.e., $p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$ and $p(\boldsymbol{y}_k|\boldsymbol{x}_k)$. The initial distribution of the state $\boldsymbol{x}_0$ is known as $p(\boldsymbol{x}_0)$. In general, the assumptions are made that $\boldsymbol{x}_k$ $(k = 1, 2, \ldots)$ subject to the first-order Markov process, and $\boldsymbol{y}_k$ $(k = 1, 2, \ldots)$ are conditionally independent given the states.

A. GPF

In order to give a brief introduction of GPF, the sequences of the states and measurements are denoted as $\boldsymbol{x}_{0:k} = \{\boldsymbol{x}_0, \ldots, \boldsymbol{x}_k\}$ and $\boldsymbol{y}_{1:k} = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_k\}$, respectively. As aforementioned, PF approximates the posterior distribution of the state $\boldsymbol{x}_k$ with a number of particles, and the approximated form of the joint posterior distribution $p(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{1:k})$ is

$$p(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{1:k}) \approx \sum_{i=1}^{N} w_k^i \delta(\boldsymbol{x}_{0:k} - \boldsymbol{x}_{0:k}^i) \tag{2}$$

in which $\delta(\cdot)$ denotes the Dirac delta function, N is the particle number, and w_k^i is the normalized importance weight (or particle weight) for the ith particle. With state $\boldsymbol{x}_{0:k}^i$ and weight w_k^i, the ith particle can be formulated as $\{\boldsymbol{x}_{0:k}^i, w_k^i\}$.

The definition of the importance weight w_k is

$$w_k \propto \frac{p(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{1:k})}{q(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{1:k})} \tag{3}$$

where $\propto$ means to be proportional to. $q(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{1:k})$ is the importance distribution. In practice, it is usually difficult to directly sample from $p(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{1:k})$ for its attributes such as multivariate and nonstandard. The importance distribution $q(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{1:k})$ is introduced to overcome this problem, and compared with $p(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{1:k})$, it can be easily sampled.

To derive the iterative formulation of the importance weight w_k, $q(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{1:k})$ is chosen to satisfy

$$q(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{1:k}) = q(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1}, \boldsymbol{y}_{1:k})q(\boldsymbol{x}_{0:k-1}|\boldsymbol{y}_{1:k-1}) \tag{4}$$

in which $q(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1}, \boldsymbol{x}_{1:k})$ is regarded as the marginal distribution of $q(\boldsymbol{x}_{0:k}|\boldsymbol{x}_{1:k})$.

Under the Bayesian framework, the joint posterior distribution $p(\boldsymbol{x}_{0:k}|\boldsymbol{x}_{1:k})$ can be written as

$$p(\boldsymbol{x}_{0:k}|\boldsymbol{x}_{1:k}) = \frac{p(\boldsymbol{x}_k|\boldsymbol{x}_{1:k-1}, \boldsymbol{x}_{0:k})p(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1})}{p(\boldsymbol{x}_k|\boldsymbol{x}_{1:k-1})} \times p(\boldsymbol{x}_{0:k-1}|\boldsymbol{y}_{1:k-1}). \tag{5}$$

The denominator part of (5) can be reformulated to

$$p(\boldsymbol{y}_k|\boldsymbol{y}_{1:k-1}) = \int p(\boldsymbol{y}_k|\boldsymbol{x}_k)p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k-1})d\boldsymbol{x}_k \tag{6}$$

in which

$$p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k-1}) = \int p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})p(\boldsymbol{x}_{k-1}|\boldsymbol{y}_{1:k-1})d\boldsymbol{x}_{k-1}. \tag{7}$$

Since $p(\boldsymbol{y}_k|\boldsymbol{x}_k)$ and $p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$ are known from the system model (1), and $p(\boldsymbol{x}_{k-1}|\boldsymbol{y}_{1:k-1})$ is the marginal posterior distribution at $(k-1)$th time step, $p(\boldsymbol{y}_k|\boldsymbol{y}_{1:k-1})$ can be regarded as a constant factor. Considering the assumptions made on the states $\boldsymbol{x}_k$ and measurements $\boldsymbol{y}_k$ $(k = 1, 2, \ldots)$, the joint posterior distribution $p(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{1:k-1})$ can be further simplified to

$$p(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{1:k}) \propto p(\boldsymbol{y}_k|\boldsymbol{x}_k)p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})p(\boldsymbol{x}_{0:k-1}|\boldsymbol{y}_{1:k-1}). \tag{8}$$

By substituting (4) and (8) into (3), a recursive formulation of the importance weight w_k can be derived as

$$w_k \propto \frac{p(\boldsymbol{y}_k|\boldsymbol{x}_k)p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})p(\boldsymbol{x}_{0:k-1}|\boldsymbol{y}_{1:k-1})}{q(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1}, \boldsymbol{y}_{1:k})q(\boldsymbol{x}_{0:k-1}|\boldsymbol{y}_{1:k-1})} \tag{9}$$

$$= \frac{p(\boldsymbol{y}_k|\boldsymbol{x}_k)p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})}{q(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1}, \boldsymbol{y}_{1:k})} w_{k-1}. \tag{10}$$

TABLE I
MULTINOMIAL RESAMPLING

Iteratively perform the following steps for $i = 1, \dots, N$

1) $u_i \sim U(0,1)$, in which $U(0,1)$ is the uniform distribution on $[0,1]$;
2) Search the variable $j \in \{1, \dots, N\}$ subjected to

$$\sum_{m=1}^{j-1} w_k^m < u_i \leq \sum_{m=1}^{j} w_k^m; \tag{15}$$

3) Store the $\mathbf{x}_k^j$ as an offspring particle, and set its particle weight to $1/N$.

In practical applications, the marginal importance distribution $q(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1}, \boldsymbol{y}_{1:k})$ is always chosen as

$$q(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1}, \boldsymbol{y}_{1:k}) = p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}). \tag{11}$$

By substituting (11) into (4), the formulation of w_k can be further simplified to satisfy

$$w_k \propto p(\boldsymbol{y}_k|\boldsymbol{x}_k) w_{k-1}. \tag{12}$$

With (12), the importance weight can be recursively calculated. If only the marginal posterior distribution $p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k})$ is required, the historical sequence state $\boldsymbol{x}_{0:k-1}$ is not necessary to be stored, and the marginal posterior distribution $p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k})$ can be approximated as

$$p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k}) \approx \sum_{i=1}^{N} w_k^i \delta(\boldsymbol{x}_k - \boldsymbol{x}_k^i). \tag{13}$$

In the following, only the marginal posterior distribution $p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k})$ will be considered as the posterior distribution of the state.

If the particles $\{\boldsymbol{x}_k^i, w_k^i\}$ $(i = 1, 2, \dots)$ are directly used for the $(k+1)$th generation, the particle population will suffer from serious degeneracy problem. After a few generations, all but one particles will be assigned with the small weights, which are negligible. The effective sample size N_{eff} [26], is proposed to measure the degree of the particle degeneracy. One commonly used approximation of N_{eff} is

$$N_{\text{eff}} \approx \frac{1}{\sum_{i=1}^{N} (w_k^i)^2}. \tag{14}$$

To solve the particle degeneracy problem, the resampling procedure is included in GPF. In each time step, the effective sample size of the particle population is calculated and compared with its threshold N_T, which can be set manually. If $N_{\text{eff}} < N_T$, the resampling is performed. By resampling, the small-weight particles are replaced by the duplicates of the large-weight particles. For the GPF introduced in [26], the mutinonial resampling is used, and it is briefly formulated in Table I.

B. Genetic Algorithm

GA is a category of population-based optimization algorithms. It is designed by mimicking the process of biological evolution. Corresponding to the individuals in nature, a number of potential optimal solutions are created from the solution space of the optimal problem, and they are updated in every generation. The main parts of GA consist of three genetic operators, i.e., selection, crossover, and mutation. The evaluation function evaluates the fitness of all potential solutions and assigns the fitness values to them. The selection operator simulates the principle of "survival of the fittest," and only the highly fitted potential solutions survive in every generation. These survived solutions can participate the following crossover procedure, in which the survived solutions act as the parent solutions and exchange part of their information to produce the offspring solutions. Just like the individuals in nature, the mutation may also happen on the offspring solutions by altering part of their information. The mutation operator effectively prevents the GA from tripping in the local optimization.

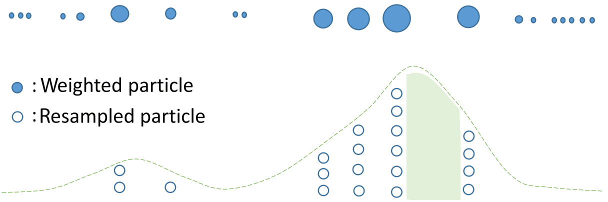


Fig. 1. Intuitively illustration of the particle impoverishment problem introduced by resampling procedure. The particle weight are presented as the size of the filled circle.

Due to the robustness and effectiveness in searching global solution of the complex optimization problems, the GA has been widely applied [27]–[29]. The traditional GAs involve coding the potential solutions into strings, for example, the commonly used binary strings. However, these GAs always encounter the expensive computation problem, especially when the space of the optimization problem is large. The real-coded GAs are another category of GAs, and there is no coding procedure needed. Compared with the binary-coded GAs, the real-coded GAs possess some superior properties, e.g., being more consistent, faster, and more precise. For more detailed descriptions on real-coded GAs, the reader can refer to [30], [31].

III. SEQUENTIAL EVOLUTIONARY FILTER

With the incorporated resampling procedure, the particle degeneracy problem is solved in the generic PF. The small-weight particles are effectively eliminated, and the duplicates of the large-weight particles are used to supplement these eliminated particles. However, as mentioned in [26], the resampling procedure also introduces new problems for the generic PF. One of the problems is the particle impoverishment, which results from the reduction of particle diversity after resampling. Fig. 1 intuitively shows the particle impoverishment problem introduced by the resampling procedure.

In the proposed SEF, the small-weight particles are changed into the large-weight ones. And at the same time, the particle diversity is also maintained. Compared with the generic PF, SEF mitigates the particle impoverishment problem. It is inspired from the idea of GA. An effective strategy, i.e., GA-inspired strategy, is implemented in SEF. In the following, this strategy will be first introduced, and then the full procedure of SEF will be presented.

A. GA-Inspired Strategy

The GA-inspired strategy is designed to maintain the particle diversity, and finally, mitigate the particle impoverishment problem encountered in GPF. Similar to GA, three operators, i.e., separation, crossover, and mutation, constitute the main part of the GA-inspired strategy. But, these three operators are different from the genetic operators in GA. They are developed according to the attributes of PF.

Assume that the particles have been obtained and they are formulated as $\{\boldsymbol{x}_k^i, w_k^i\} \quad (i = 1, 2, \ldots, N)$. The separation operator aims to separate the particle population into two categories. One is composed of the small-weight particles and the other is composed of the large-weight particles. To achieve this separation, all the particles are sorted first according to their weights

$$\tilde{\boldsymbol{\chi}} = \text{Sort}_{\text{desc.}}\left\{\{\boldsymbol{x}_k^1, w_k^1\}, \ldots, \{\boldsymbol{x}_k^i, w_k^N\}\right\} \tag{15}$$

$$= \left\{\{\tilde{\boldsymbol{x}}_k^1, \tilde{w}_k^1\}, \ldots, \{\tilde{\boldsymbol{x}}_k^i, \tilde{w}_k^N\}\right\} \tag{16}$$

where $\text{Sort}_{\text{desc.}}\{\cdot\}$ is the operation used to sort the particles in descending order according to their weights w_k^i. $\{\tilde{\boldsymbol{x}}_k^i, \tilde{w}_k^i\}$ represents the sorted particle. The separation is performed on these sorted particles

$$\begin{cases} \tilde{\boldsymbol{\chi}}_L = \left\{\{\tilde{\boldsymbol{x}}_k^1, \tilde{w}_k^1\}, \ldots, \{\tilde{\boldsymbol{x}}_k^m, \tilde{w}_k^m\}\right\} \\ \tilde{\boldsymbol{\chi}}_S = \left\{\{\tilde{\boldsymbol{x}}_k^{m+1}, \tilde{w}_k^{m+1}\}, \ldots, \{\tilde{\boldsymbol{x}}_k^N, \tilde{w}_k^N\}\right\} \end{cases} \tag{17}$$

in which $\tilde{\boldsymbol{\chi}}_L$ is the category including large-weight particles, and $\tilde{\boldsymbol{\chi}}_S$ is the category including the small-weight particles. m is a integer, and

$$m \leq \gamma N_{\text{eff}} < m + 1. \tag{18}$$

where N_{eff} is the effective sample size calculated by (14). $\gamma \in (0, 1]$ is a parameter that is set in advance.

In the real-coded GAs, a commonly used crossover is the arithmetic crossover [32]. Suppose that two of the selected parent potential solutions are represented as $\boldsymbol{x}_{\text{par}}^1$ and $\boldsymbol{x}_{\text{par}}^2$, and their offsprings are represented as $\boldsymbol{x}_{\text{off}}^1$ and $\boldsymbol{x}_{\text{off}}^2$. The arithmetic crossover is represented as

$$\boldsymbol{x}_{\text{off}}^1 = a\boldsymbol{x}_{\text{par}}^1 + (1 - a)\boldsymbol{x}_{\text{par}}^2 \tag{19}$$

$$\boldsymbol{x}_{\text{off}}^2 = a\boldsymbol{x}_{\text{par}}^2 + (1 - a)\boldsymbol{x}_{\text{par}}^1 \tag{20}$$

where a is a random number drawn from the interval $[0, 1]$.

The crossover operator used in the GA-inspired strategy is modified from the arithmetic crossover. It is designed to modify the small-weight particles into the particles with the large weights. Denote the state of the large-weight particle as $\tilde{\boldsymbol{x}}_{kL}^l$, and the state of the small-weight particle as $\tilde{\boldsymbol{x}}_{kS}^j$, in which $l = 1, \ldots, m$ and $j = 1, \ldots, N - m$. Hence, the $\tilde{\boldsymbol{\chi}}_L$ and $\tilde{\boldsymbol{\chi}}_S$ can be reformulated as

$$\begin{cases} \tilde{\boldsymbol{\chi}}_L = \left\{\{\tilde{\boldsymbol{x}}_{kL}^1, \tilde{w}_{kL}^1\}, \ldots, \{\tilde{\boldsymbol{x}}_{kL}^m, \tilde{w}_{kL}^m\}\right\} \\ \tilde{\boldsymbol{\chi}}_S = \left\{\{\tilde{\boldsymbol{x}}_{kS}^1, \tilde{w}_{kS}^1\}, \ldots, \{\tilde{\boldsymbol{x}}_{kS}^{N-m}, \tilde{w}_{kS}^{N-m}\}\right\} \end{cases}. \tag{21}$$

The state of the moved particle is represented by $\boldsymbol{x}_{kO}^j$. Then, the crossover operator in GA-inspired strategy is

$$\tilde{\boldsymbol{x}}_{kO}^j = \alpha_j \tilde{\boldsymbol{x}}_{kS}^j + (1 - \alpha_j)\tilde{\boldsymbol{x}}_{kL}^l. \tag{22}$$

Similar to the arithmetic crossover, α_j is also a random number, but it is drawn from the uniform distribution over $[0, \bar{\alpha}]$, i.e., $\alpha_j \sim U(0, \bar{\alpha})$. $\bar{\alpha}$ is the upper bound of α_j, and in every step, it can be calculated as

$$\bar{\alpha} = 1 - N_{\text{eff}}/N. \tag{23}$$

The value of α_j determines how much information will be transferred from $\tilde{\boldsymbol{x}}_{kL}^l$ to $\tilde{\boldsymbol{x}}_{kO}^j$. The smaller the value of α_j is the more information of $\tilde{\boldsymbol{x}}_{kL}^l$ will be transferred. For each $\tilde{\boldsymbol{x}}_{kS}^j$, $\tilde{\boldsymbol{x}}_{kL}^l$ is randomly selected from $\tilde{\boldsymbol{\chi}}_L$. Different from the arithmetic crossover, the modifications are only made on the small-weight particles, and only one offspring particle, i.e., $\tilde{\boldsymbol{x}}_{kO}^j$, is created by the crossover in the GA-inspired strategy. For the large-weight particles, no modifications are made on them, since they are important for approximating the posterior distribution $p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k})$.

In order to further improve the diversity of particle population, the mutation operator is designed to extend the state-search space of particles. The definition is

$$\tilde{\boldsymbol{x}}_{kM}^j = \begin{cases} 2\tilde{\boldsymbol{x}}_{kL}^l - \tilde{\boldsymbol{x}}_{kO}^j, & \text{if } r_j \leq p_M \\ \tilde{\boldsymbol{x}}_{kO}^j, & \text{if } r_j > p_M \end{cases} \tag{24}$$

where $\tilde{\boldsymbol{x}}_{kL}^l$ and $\tilde{\boldsymbol{x}}_{kO}^j$ are the particle states in (23). $r_j \sim U(0, 1)$, and $p_M \in [0, 1]$ is the mutation probability. When $r_j \leq p_M$, the mutation will be performed on $\tilde{\boldsymbol{x}}_{kO}^j$.

After performing the GA-inspired strategy, the small-weight particles will be modified into the ones with the large weights. In the next time step, all particles will be treated equally. Based on this idea, the particle weights will be reset to $1/N$ after the GA-inspired strategy as

$$\boldsymbol{\chi} = \left\{\{\tilde{\boldsymbol{x}}_{kL}^1, 1/N\}, \ldots, \{\tilde{\boldsymbol{x}}_{kL}^m, 1/N\}\right.$$
$$\left.\{\tilde{\boldsymbol{x}}_{kM}^1, 1/N\}, \ldots, \{\tilde{\boldsymbol{x}}_{kM}^{(N-m)}, 1/N\}\right\} \tag{25}$$
$$= \left\{\{\boldsymbol{x}_k^1, 1/N\}, \ldots, \{\boldsymbol{x}_k^N, 1/N\}\right\}. \tag{26}$$

To intuitively show the GA-inspired strategy, a diagram under the one dimension situation is presented in Fig. 2. The sizes of the circles represent the particle weights. Different colored circles denote the particles obtained from different stages. The circles filled with navy blue correspond to the parent particles in Fig. 1. The circles filled with red and orange denote the small- and large-weight particles, respectively. The circles filled with light blue are the particle created by mutation. And the circle filled with no color is the final particle, i.e., $\{\boldsymbol{x}_k^i, 1/N\}$, created by GA-inspired strategy.

Different from the resampling used in GPF, which only replicates the large-weight particles to replace the small-weight

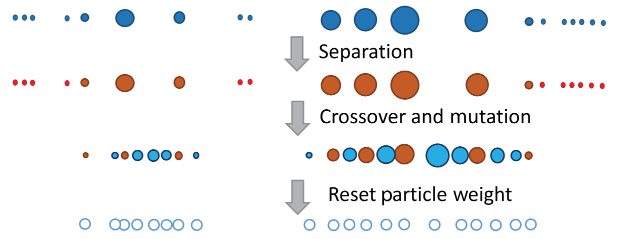


Fig. 2. Diagram of the GA-inspired strategy.

TABLE II
IMPLEMENTATION PROCEDURE OF SEF

1) Formulate N initial particles, i.e., $\{\boldsymbol{x}_0^i, 1/N\}$ $(i = 1, \cdots, N)$, in which $\boldsymbol{x}_0^i \sim p(\boldsymbol{x}_0)$
2) Perform the following steps for $k = 1, 2, \ldots$
   a) Sample from the distribution $p(\boldsymbol{x}_k | \boldsymbol{x}_{k-1}^i)$
   b) Calculate the normalized weights for all particles using equation (12). The particles are denoted as $\{\boldsymbol{x}_k^i, w_k^i\}$
   c) Estimate the hidden state $\boldsymbol{x}_k$ by an appropriate method, e.g., MMSE
   d) Perform the GA-inspired strategy to obtain the particle $\{\boldsymbol{x}_k^i, 1/N\}$

ones, the GA-inspired strategy modifies the the small-weight particles into the large-weight one. But, these modified small-weight particles will not overlap the particles with large weights. In this way, the diversity of the particle population can be maintained, and thus the particle impoverishment encountered in GPF is mitigated.

B. Sequential Evolutionary Filter

Once the weighted particles $\{\boldsymbol{x}_k^i, w_k^i\}$ $(i = 1, \ldots, N)$ have been obtained, the system state at the kth time can be estimated by appropriate methods, for example, MMSE and MAP. The definition of MMSE is

$$\hat{\boldsymbol{x}}_k^{MMSE} = \sum_{i=1}^{N} w_k^i \boldsymbol{x}_k^i \tag{27}$$

and the definition of MAP is

$$\hat{\boldsymbol{x}}_k^{MAP} = \arg\max_{\boldsymbol{x}_k^i} w_k^i. \tag{28}$$

The choice of MMSE or MAP depends on different estimation problems. In the experimental part of this work, the MMSE is used to estimate the system states.

In the GA-inspired strategy, a simpler formulation of the crossover and mutation operators can be derived. Assume that we have obtained the separated particles, i.e., $\{\tilde{\boldsymbol{x}}_{kL}^l, w_{kL}^l\} \in \tilde{\chi}_L$ and $\{\tilde{\boldsymbol{x}}_{kS}^j, w_{kS}^j\} \in \tilde{\chi}_S$. Then, the simpler formulation can be presented as

$$\tilde{\boldsymbol{x}}_{kM}^j = \begin{cases} (\alpha_j + 1)\tilde{\boldsymbol{x}}_{kL}^l - \alpha_j \tilde{\boldsymbol{x}}_{kS}^j, & \text{if } r_j \le p_M \\ \alpha_j \tilde{\boldsymbol{x}}_{kS}^j + (1 - \alpha_j)\tilde{\boldsymbol{x}}_{kL}^l, & \text{if } r_j > p_M \end{cases} \tag{29}$$

which combines the crossover and mutation operators.

Compared with the GPF, SEF replaces the resampling with the designed GA-inspired strategy, with which the particle diversity is maintained and the particle impoverishment in GPF is mitigated. A brief implementation procedure of SEF is presented in Table II.

IV. EXPERIMENTS

In this section, two experimental systems are used. The first one includes a commonly used numerical example [4], [20]. The highly nonlinear and bimodal characteristics of this example bring challenges for state estimation issues. The other one is the three-tank system that always acts as the benchmark in the process control [33], [34]. To evaluate the performances of SEF, three of other PFs are included for comparison. One is the GPF introduced in Section II-A. The second is the modified PF (RPF) introduced in [19]. Its main idea is to rough the over-centralized particles with the Gaussian jitter noise. The roughing procedure can also be recognized as performing the mutation operator of GA. The last one is called the real-coded GA PF (RGAPF) that is based on the real-coded GA [24].

A. Numerical Example

The process and measurement equations of the system can be formulated as

$$x_k = \frac{1}{2}x_{k-1} + \frac{25x_{k-1}}{1 + x_{k-1}^2} + 8\cos[1.2(k-1)] + \omega_k \tag{30}$$

$$y_k = \frac{1}{20}x_k^2 + \nu_k, \qquad k = 1, 2, \ldots \tag{31}$$

where $\omega_k \sim \mathcal{N}(0, \sigma_\omega^2)$ and $\nu_k \sim \mathcal{N}(0, \sigma_\nu^2)$ represent the process and measurement noises, respectively. In the experiment, the variances are set as $\sigma_\omega = 1, \sigma_\nu = \sqrt{2}$. x_k is the system state that needs to be estimated. Its initial value is set to $x_0 = 0.1$. The initial particles $\{x_0^i, 1/N\}$ $(i = 1, \ldots, N)$ are drawn from a Gaussian distribution, i.e., $x_0^i \sim \mathcal{N}(0.1, 2)$.

In this experiment, the particle number is set to $N = 500$. Both the thresholds of N_{eff} for GPF and RPF are set to $N_T = 0.5N$. The constant tuning parameter K in RGPF is set to $K = 0.2$ [19]. The parameters in SEF are set as $p_M = 0.5$ and $\gamma = 1$. A metric, i.e., the absolute error of state, is defined as

$$e = \frac{1}{n}\sum_{k=1}^{n} |x_k - \hat{x}_k| \tag{32}$$

where n is the number of the time steps. The experiment is repeatedly performed for 30 times. To evaluate the performance of these four PFs, the averaged absolute error $\bar{u}_e$ and variance of the absolute error V_{ar} are calculated. To estimate the computational burdens of four PFs, the averaged running time $\bar{u}_t$ is recorded.

The experiment results are presented in Table III. As shown in the table, four PFs give the comparative estimation accuracies which can be reflected by the average absolute error $\bar{u}_e$. The variances V_{ar} of four PFs are all very small. This means that these four PFs can offer stable state estimation results.

TABLE III
EXPERIMENTAL RESULTS OF FOUR PFS IN THE NUMERICAL EXAMPLE

	SEF	GPF	RPF	RGAPF
$\bar{u}_e$	1.720	1.725	1.760	1.712
Var	0.066	0.048	0.085	0.110
$\bar{u}_t$	0.036	0.164	0.339	0.382

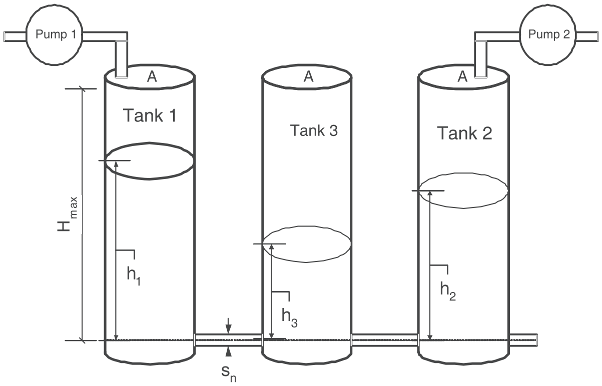


Fig. 3. Three-tank system.

However, the average running time $\bar{u}_t$ of four PFs are largely different from each other. Compared with GPF, the extra part, e.g., Gaussian jitter noise or genetic operators, is added in the RPF and RGAPF to mitigate the particle impoverishment. The added extra part directly results in the higher computational burden of RPF and RGAPF than GPF's. From the table, it can also be observed that the lowest computational burden among these four PFs belongs to SEF with $\bar{u}_t = 0.036$ s.

B. Three-Tank System

In the practical industrial application, the tank system always serves as the important part of the plant, especially for the chemical or biopharmaceutical process. The three-tank system introduced here is a simplified tank system used in the practical plant. It acts as a popular nonlinear system for the process control [33], [34]. The simplified form of the three-tank system is presented in Fig. 3. Three tanks constitute the main part of the system, and they share the same section area A. These three tanks are connected by pipelines. The water is pumped into the tanks 1 and 2 with the flow rate Q_1 and Q_2, respectively. According to the "mass balance" principle, the system can be mathematically formulated as

$$\frac{\mathrm{d}h_1}{\mathrm{d}t} = \frac{1}{A}(Q_1 - Q_{13}) \tag{33}$$

$$\frac{\mathrm{d}h_2}{\mathrm{d}t} = \frac{1}{A}(Q_2 + Q_{32} - Q_{20}) \tag{34}$$

$$\frac{\mathrm{d}h_3}{\mathrm{d}t} = \frac{1}{A}(Q_{13} - Q_{32}) \tag{35}$$

in which h_1, h_2, and h_3 are the water levels in three tanks. Q_{ij} denotes the flow rate in the pipeline from the ith to jth tank, and it can be modeled as

$$Q_{ij} = a_i s_i \mathrm{sgn}(h_i - h_j)\sqrt{2g|h_i - h_j|}. \tag{36}$$

In Fig. 3, the pipelines are numbered 1, 3, and 2 from left to right. s_i and a_i represent the sectional area and coefficient of flow for the ith pipeline, respectively. g is the acceleration due to gravity, and $g = 980\,\mathrm{cm/s^2}$. $\mathrm{sgn}(\cdot)$ is the sign function. Q_{20} is the flow rate in the pipeline, which outputs the water in tank 2, and it is modeled as

TABLE IV
PARAMETERS OF THE THREE-TANK SYSTEM

Parameters	Symbol	Value
Cross section area of tanks	A	154 cm^2
Cross section area of pipes	S_{13}, S_{23}, S_0	0.5 cm^2
Max. height of tanks	$H_{\max}$	62 cm
Max. flow rate of pump 1	$Q_{1\max}$	100 cm^3
Max. flow rate of pump 2	$Q_{2\max}$	100 cm^3
Coeff. of flow for pipeline 1	a_1	0.46
Coeff. of flow for pipeline 2	a_2	0.60
Coeff. of flow for pipeline 3	a_3	0.45

$$Q_{20} = a_2 s_2 \sqrt{2gh_2}. \tag{37}$$

Let $\boldsymbol{x} = [x_1, x_2, x_3]^T = [h_1, h_2, h_3]^T$ and $\boldsymbol{u} = [u_1, u_2]^T = [Q_1, Q_2]^T$. In this paper, both the process and measurement noises are considered, and the dynamic state-space model of the three-tank system can be formulated as

$$\dot{x}_1 = \frac{1}{A}\left(u_1 - a_1 s_1 \mathrm{sgn}(x_1 - x_3)\sqrt{2g|x_1 - x_3|}\right) + w_1 \tag{38}$$

$$\dot{x}_2 = \frac{1}{A}\left(u_2 + a_3 s_3 \mathrm{sgn}(x_3 - x_2)\sqrt{2g|x_3 - x_2|}\right.$$
$$\left. - a_2 s_2 \sqrt{2gx_2}\right) + w_2 \tag{39}$$

$$\dot{x}_3 = \frac{1}{A}\left(a_1 s_1 \mathrm{sgn}(x_1 - x_3)\sqrt{2g|x_1 - x_3|}\right.$$
$$\left. - a_3 s_3 \mathrm{sgn}(x_3 - x_2)\sqrt{2g|x_3 - x_2|}\right) + w_3 \tag{40}$$

$$\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{\nu} \tag{41}$$

in which $\boldsymbol{x} = [x_1, x_2, x_3]^T$, $\boldsymbol{y} = [y_1, y_2, y_3]^T$, and $\boldsymbol{\nu} = [\nu_1, \nu_2, \nu_3]^T$. $w_i (i = 1, 2, 3)$ are the process noises, and $\boldsymbol{\nu}$ is the measurement noise. Here, both w_i and $\boldsymbol{\nu}$ are assumed to be subject to the Gaussian distribution, i.e., $w_i \sim \mathcal{N}(0, 2)$ and $\boldsymbol{\nu} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_\nu)$, in which $\boldsymbol{\Sigma}_\nu = \mathrm{diag}(1, 1, 1)$. The parameters of the system are given in Table IV.

In this section, the performance of the introduced SEF will be evaluated on the three-tank system. The parameters of SEF are set as: $p_M = 0.5$ and $\gamma = 0.5$. The particle number for four discussed filters is set to $N = 1000$. As in the numerical example, the thresholds of N_{eff} for GPF and RPF are set to $N_T = 0.5N$. The initial particles are drawn from $p(\hat{\boldsymbol{x}}_0) = \mathcal{N}(\hat{\boldsymbol{x}}_0, \boldsymbol{\Sigma}_0)$, in which $\hat{\boldsymbol{x}}_0 = \boldsymbol{x}_0 = [50.93, 23.94, 37.73]^T$ and $\boldsymbol{\Sigma}_0 = \mathrm{diag}(2, 2, 2)$. $\mathrm{diag}(\cdot)$ denotes the diagonal matrix. The experiment is performed for 100 s and the data are collected by the sampling period of 0.1 s. The system input is

$$\boldsymbol{u} = \begin{cases} [10, 50]^T, & t = 20 \sim 60\,\mathrm{s} \\ [37, 28]^T, & \text{others} \end{cases} \tag{42}$$

in which $\boldsymbol{u} = [u_1, u_2]^T$.

The state estimation results of four filters are presented in Figs. 4–7. The black and red lines, respectively, represent the

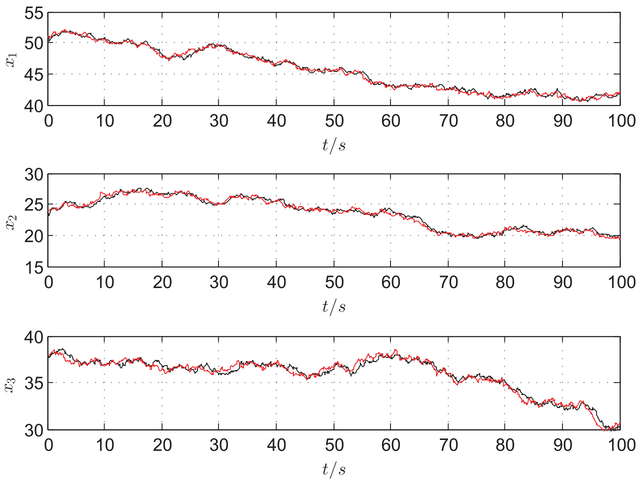


Fig. 4. State estimation results of SEF.

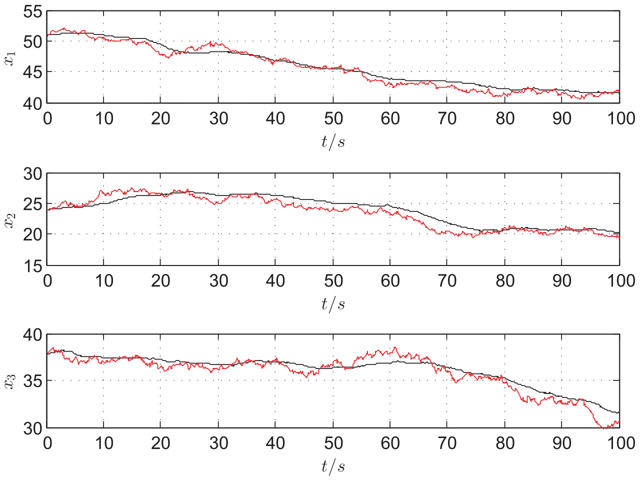


Fig. 5. State estimation results of GPF.

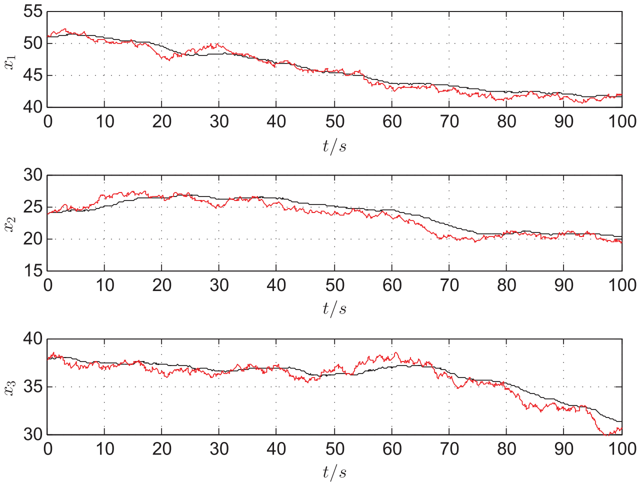


Fig. 6. State estimation results of RPF.

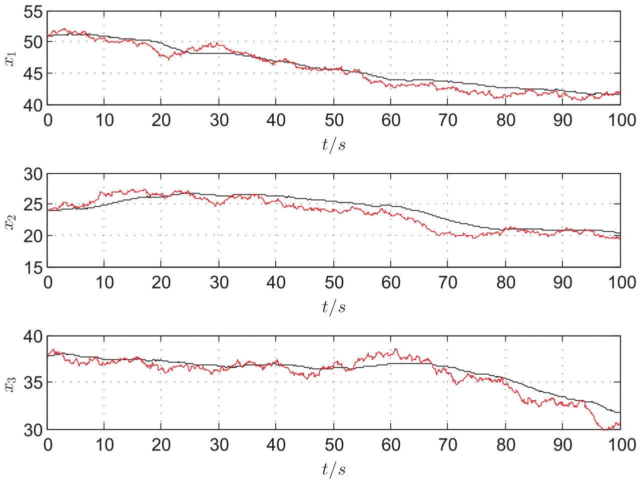


Fig. 7. State estimation results of RGAPF.

TABLE V
AVERAGED ABSOLUTE ERRORS OF SEF AND GPF

	SEF			GPF		
	x_1	x_2	x_3	x_1	x_2	x_3
$\bar{u}_e$	0.292	0.308	0.315	0.559	0.768	0.558
	RPF			RGAPF		
	x_1	x_2	x_3	x_1	x_2	x_3
$\bar{u}_e$	0.569	0.786	0.567	0.705	0.998	0.656

estimated and real states. From these figures, it can be observed that SEF provides the best state estimation. This mainly owes to the GA-inspired strategy. With this strategy, the particle diversity is maintained, and the particle impoverishment encountered in GPF is mitigated. GPF and RPF offer the similar state estimation results. The worst state estimation results belong to the RGAPF. The simulations are performed for 10 times, and the averaged absolute errors of four filters are presented in Table V. The table also shows that SEF offers the more accurate state estimation results than GPF, RPF, and RGAPF. In the experiment, we find that the variances of the absolute errors of all four filters for 10 times experiments approximate to 0. This means that all these four filters can offer stable state estimation results. The averaged running time $\bar{u}_t$ for SEF, GPF, RPF, and RGAPF are 14.573, 11.362, 14.591, and 15.053 s, respectively. As the averaged running time recorded in the numerical example, the $\bar{u}_t$ of RPF and RGAPF are larger than GPF's. But, for this MIMO system, the $\bar{u}_t$ of SEF is also greater than GPF's.

V. CONCLUSION

In this paper, an SEF technique, which is based on the PF and GA, has been introduced. It replaces the resampling in GPF with a GA-inspired strategy, by which the particle impoverishment encountered in GPF is mitigated. The main idea of the GA-inspired strategy is to modify the small-weight particles into the particles with large weights, and at the same time, maintain the diversity of the particle population. The performances of SEF are evaluated by two experiments. The experimental results show that compared with three other filters, SEF offers the better state estimation results.

REFERENCES

[1] A. Shenoy, J. Prakash, V. Prasad, S. Shah, and K. McAuley, "Practical issues in state estimation using particle filters: Case studies with polymer reactors," *J. Process Control*, vol. 23, no. 2, pp. 120–131, 2013.

[2] S. Yin, S. X. Ding, X. Xie, and H. Luo, "A review on basic data-driven approaches for industrial process monitoring," *IEEE Trans. Ind. Electron.*, vol. 61, no. 11, pp. 6418–6428, Nov. 2014.

[3] G. G. Rigatos, "A derivative-free Kalman filtering approach to state estimation-based control of nonlinear systems," *IEEE Trans. Ind. Electron.*, vol. 59, no. 10, pp. 3987–3997, Oct. 2012.

[4] S. Yin, X. Zhu, and O. Kaynak, "Improved PLS focused on key-performance-indicator-related fault diagnosis," *IEEE Trans. Ind. Electron.*, vol. 62, no. 3, pp. 1651–1658, Mar. 2015.

[5] I. Miller, M. Campbell, and D. Huttenlocher, "Efficient unbiased tracking of multiple dynamic obstacles under large viewpoint changes," *IEEE Trans. Robot.*, vol. 27, no. 1, pp. 29–46, Feb. 2011.

[6] X. Zhang, W. Hu, S. Maybank, X. Li, and M. Zhu, "Sequential particle swarm optimization for visual tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2008, pp. 1–8.

[7] P. Vadakkepat and L. Jing, "Improved particle filter in sensor fusion for tracking randomly moving object," *IEEE Trans. Instrum. Meas.*, vol. 55, no. 5, pp. 1823–1832, Oct. 2006.

[8] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with Rao-Blackwellized particle filters," *IEEE Trans. Robot.*, vol. 23, no. 1, pp. 34–46, Feb. 2007.

[9] M. M. Atia, J. Georgy, M. Korenberg, and A. Noureldin, "Real-time implementation of mixture particle filter for 3D RISS/GPS integrated navigation solution," *Electron. Lett.*, vol. 46, no. 15, pp. 1083–1084, Jul. 2010.

[10] S. Chen, "Kalman filter for robot vision: A survey," *IEEE Trans. Ind. Electron.*, vol. 59, no. 11, pp. 4409–4420, Nov. 2012.

[11] S. Yin, P. Shi, and H. Yang, "Adaptive fuzzy control of strict-feedback nonlinear time-delay systems with unmodeled dynamics," *IEEE Trans. Cybern.*, DOI: 10.1109/TCYB.2015.2457894.

[12] M. A. Khanesar, E. Kayacan, M. Teshnehlab, and O. Kaynak, "Extended Kalman filter based learning algorithm for type-2 fuzzy logic systems and its experimental evaluation," *IEEE Trans. Ind. Electron.*, vol. 59, no. 11, pp. 4443–4455, Nov. 2012.

[13] A. Germani, C. Manes, and P. Palumbo, "Polynomial extended Kalman filter," *IEEE Trans. Automat. Control*, vol. 50, no. 12, pp. 2059–2064, Dec. 2005.

[14] J. Prakash, S. C. Patwardhan, and S. L. Shah, "On the choice of importance distributions for unconstrained and constrained state estimation using particle filter," *J. Process Control*, vol. 21, no. 1, pp. 3–16, 2011.

[15] S. Das, A. Kale, and N. Vaswani, "Particle filter with a mode tracker for visual tracking across illumination changes," *IEEE Trans. Image Process.*, vol. 21, no. 4, pp. 2340–2346, Apr. 2012.

[16] A. S. Sarathi Vasan, B. Long, and M. Pecht, "Diagnostics and prognostics method for analog electronic circuits," *IEEE Trans. Ind. Electron.*, vol. 60, no. 11, pp. 5277–5291, Nov. 2013.

[17] E. Zio and G. Peloni, "Particle filtering prognostic estimation of the remaining useful life of nonlinear components," *Reliab. Eng. Syst. Safety*, vol. 96, no. 3, pp. 403–409, 2011.

[18] Z. Duan, Z. Cai, and J. Yu, "Fault diagnosis and fault tolerant control for wheeled mobile robots under unknown environments: A survey," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 2005, pp. 3428–3433.

[19] N. J. Gordon, D. J. Salmond, and A. F. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," *Proc. Inst. Elect. Eng. Radar Signal Process.*, vol. 140, no. 2, pp. 107–113, Apr. 1993.

[20] U. Orguner and F. Gustafsson, "Risk-sensitive particle filters for mitigating sample impoverishment," *IEEE Trans. Signal Process.*, vol. 56, no. 10, pp. 5001–5012, Oct. 2008.

[21] T. Higuchi, "Monte Carlo filter using the genetic algorithm operators," *J. Stat. Comput. Simul.*, vol. 59, no. 1, pp. 1–23, 1997.

[22] S. Park, J. P. Hwang, E. Kim, and H.-J. Kang, "A new evolutionary particle filter for the prevention of sample impoverishment," *IEEE Trans. Evol. Comput.*, vol. 13, no. 4, pp. 801–809, Aug. 2009.

[23] S. Yin and X. Zhu, "Intelligent particle filter and its application on fault detection of nonlinear system," *IEEE Trans. Ind. Electron.*, vol. 62, no. 6, pp. 3852–3861, Jun. 2015.

[24] M. S. Hussain, "Real-coded genetic algorithm particle filters for high-dimensional state spaces," Dept. Comput. Sci., Ph.D. dissertation, Univ. College London, London, U.K., 2014.

[25] J. Zhong and Y.-F. Fung, "Case study and proofs of ant colony optimisation improved particle filter algorithm," *IET Control Theory Appl.*, vol. 6, no. 5, pp. 689–697, Mar. 2012.

[26] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Trans. Signal Process.*, vol. 50, no. 2, pp. 174–188, Feb. 2002.

[27] M. Paulinas and A. Ušinskas, "A survey of genetic algorithms applications for image enhancement and segmentation," *Inf. Technol. Control*, vol. 36, no. 3, pp. 278–284, 2007.

[28] V. Roberge, M. Tarbouchi, and G. Labonté, "Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning," *IEEE Trans. Ind. Informat.*, vol. 9, no. 1, pp. 132–141, Feb. 2013.

[29] Y. Yoon and Y.-H. Kim, "An efficient genetic algorithm for maximum coverage deployment in wireless sensor networks," *IEEE Trans. Cybern.*, vol. 43, no. 5, pp. 1473–1483, Oct. 2013.

[30] K. Deep, K. P. Singh, M. Kansal, and C. Mohan, "A real coded genetic algorithm for solving integer and mixed integer optimization problems," *Appl. Math. Comput.*, vol. 212, no. 2, pp. 505–518, 2009.

[31] B. A. Sawyerr, M. M. Ali, and A. O. Adewumi, "A comparative study of some real-coded genetic algorithms for unconstrained global optimization," *Optim. Method Softw.*, vol. 26, no. 6, pp. 945–970, 2011.

[32] T. Yalcinoz, H. Altun, and M. Uzam, "Economic dispatch solution using a genetic algorithm based on arithmetic crossover," in *Proc. IEEE Proto Power Tech. Conf.*, Proto, Portugal, Sep. 2001, p. 4.

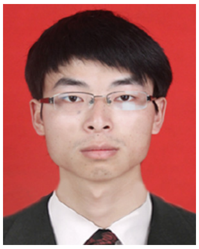
[33] S. X. Ding, *Model-Based Fault Diagnosis Techniques: Design Schemes, Algorithms, and Tools*. New York, NY, USA: Springer, 2008.

[34] R. Dormido *et al.*, "Development of a web-based control laboratory for automation technicians: The three-tank system," *IEEE Trans. Educ.*, vol. 51, no. 1, pp. 35–44, Feb. 2008.

Shen Yin (M'12–SM'15) received the B.E. degree in automation from Harbin Institute of Technology, Harbin, China, in 2004, and the M.Sc. degree in control and information systems and the Ph.D. degree in electrical engineering and information technology from the University of Duisburg-Essen, Essen, Germany, in 2007 and 2012, respectively.

Currently, he is a Full Professor in the School of Astronautics, Harbin Institute of Technology. His research interests include model-based and data-driven fault diagnosis, fault-tolerant control, and big data focused on industrial electronics applications.

Prof. Yin currently serves as the Technical Committee Chair of the IEEE Industrial Electronics Society Data Driven Control and Monitoring Committee. He is an Associate Editor for the IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, *Journal of the Franklin Institute*, *Neurocomputing*, *International Journal of Applied Mathematics and Computer Science*, *IE Technology News*, *Journal of Intelligent and Fuzzy Systems*, etc.



Xiangping Zhu (S'14) is currently working toward the Ph.D. degree in control science and engineering at Harbin Institute of Technology, Harbin, China.

His research interests include complex system modeling, data-driven fault diagnosis, nonlinear filters, and their applications in industry.

Jianbin Qiu (M'10–SM'15) received the B.Eng. and Ph.D. degrees in mechanical and electrical engineering from the University of Science and Technology of China (USTC), Hefei, China, in 2004 and 2009, respectively, and the Ph.D. degree in mechatronics engineering from the City University of Hong Kong, Kowloon, Hong Kong, in 2009.

Since 2009, he has been with the School of Astronautics, Harbin Institute of Technology, Harbin, China, where he is currently a Full Professor. His research interests include intelligent and hybrid control systems, signal processing, and robotics.

Huijun Gao (SM'09–F'14) received the Ph.D. degree in control science and engineering from Harbin Institute of Technology, Harbin, China, in 2005. From 2005 to 2007, he carried out his postdoctoral research with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB, Canada.

Since November 2004, he has been with Harbin Institute of Technology, where he is currently a Professor and Director of the Research Institute of Intelligent Control and Systems. His research interests include network-based control, robust control/filter theory, time-delay systems, and their engineering applications.

Dr. Gao is a Co-Editor-in-Chief for the IEEE Transactions on Industrial Electronics, and an Associate Editor for *Automatica*, IEEE Transactions on Cybernetics, IEEE Transactions on Fuzzy Systems, IEEE/ASME Transactions on Mechatronics, and IEEE Transactions on Control Systems Technology. He is serving on the Administrative Committee of IEEE Industrial Electronics Society. He was the recipient of the IEEE J. David Irwin Early Career Award From IEEE Industrial Electronics Society.