MultiPDF particle filtering in state estimation of nonlinear objects

Jacek Michalski · Piotr Kozierski · Wojciech Giernacki · Joanna Zietkiewicz · Marek Retinger

Abstract This paper presents a new particle filter algorithm (MultiPDF) for state estimation of nonlinear systems. The proposed method is a modification of the standard particle filter approach. Due to the strong need for the acceleration of calculations and an improvement in the estimation quality of state estimation, the authors propose a method which enables one to divide the main particle filter into smaller sub-filters with an accordingly smaller number of particles for each one of them. The algorithm has been implemented for various numbers of particles and subordinate parallel filters. Estimation quality has been checked for nine nonlinear objects (both one- and multidimensional) and evaluated through the quality index, average root-mean-squared error. The computation time of the particle filter algorithm for several hardware configurations has been compared. Based on the obtained results, it can be concluded that, besides the computation acceleration, the parallelization of the particle filter’s operation also improves the estimation quality.

Keywords State estimation · Dynamical systems · Particle filter · MultiPDF particle filter · Multivariable systems · Nonlinear objects

1 Introduction

Scientific and technical problems with noisy measurements require state estimation to filter out the noise. State estimation is widely used in power systems [1,2], electronics and electric drives [3,4], unmanned aerial vehicles (UAVs) [5–7] and in image processing [8,9]. An efficient tool for measurement noise filtering is the particle filter (PF), firstly proposed by Gordon, Salmond and Smith in 1993 [10]. The PF algorithm builds on the Bayesian Filter, which is the optimal solution [11]. It enables estimation in the presence of noise with any probability density function (PDF). Furthermore, it provides proper estimates even for strongly nonlinear objects, in contrast to Kalman filter algorithms [12].

Over the years, many modifications of the particle filter have been made, for example, the mixed Kalman particle filter (MKPF) [13], i.e., a combination of two Kalman filters, extended and unscented, with the PF method. This approach provides the advantage of combining the features of all three methods. Another example is the Remaining Useful Life Particle Filter
The paper is organized as follows. In Sect. 2, the algorithm of the particle filter is presented. Section 3 contains a description of MultiPDF Particle Filter—the parallelization of PF calculations were also highlighted in the surveys [29,30], in which several proposals were also presented, like non-interacting particle filters (not using communication between sub-filters) or particle islands (using the sequential Monte Carlo SMC method). Additionally, the total particles’ population is divided here into sub-populations (named “islands”). As presented above, sometimes, the PFs usage can be problematic because of their computational complexity, especially with a large number of particles and with multidimensional systems [31]. A modification of the particle filter proposed here by the authors can solve this problem. Additionally, the main aim of this research is to improve the quality of the estimation and find the group of objects for which this method works best. It is obvious that it is impossible to obtain a better quality of the estimation than for the Bayesian filter; however, the particle filter is only an approximation, so one can search for methods and modifications thanks to which, despite the use of a relatively small number of particles, the quality of the estimation will approach the optimal solution (for the case of using continuous PDF). The proposed method, named MultiPDF Particle Filter (MultiPDF PF), divides the total number of particles into smaller parts, which allows one to parallelize the computation and reduces the overall time cost. This approach also makes it possible to improve the quality of results compared to the traditional PF. As will be shown, several smaller sub-filters can mutually compensate for incorrect estimates.

In the vein of the MultiPDF approach, a total probability density function, which should be estimated by PF, is divided into parts. Each partial PDF is calculated using its own sub-filter with, respectively, smaller number of particles. (The total particle number is divided by the number of sub-filters, which can work parallelly.) As is shown in this paper, the total PDF can be treated as the set of particles from all sub-filters; however, it is not estimated in itself, and only the expected value of the PDF is calculated as the mean of expected values from all sub-filters (in each time step). Factors improving the quality of estimation using this method (and special types of plants) were indicated. The paper also shows the possibilities of accelerating the calculations thanks to the use of the MultiPDF PF.
Table 1 Description of important symbols used in the paper

| Symbol | Explanation |
|--------|-------------|
| \(x^{(k)}\) | State vector at \(k\)-th time step |
| \(\hat{x}_j^{(k)}\) | Estimated value of \(j\)-th state variable at \(k\)-th time step |
| \(x_j^{(k)}\) | Real value of \(j\)-th state variable at \(k\)-th time step |
| \(x_i^{(k)}\) | \(i\)-th particle vector at \(k\)-th time step |
| \(x_i^{(k)}\) | \(i\)-th particle vector from \(l\)-th independent filter at \(k\)-th time step |
| \(q_i^{(k)}\) | Weight of \(i\)-th particle at \(k\)-th time step |
| \(u^{(k)}\) | Input (control) vector at \(k\)-th time step |
| \(y^{(k)}\) | Output (measurement) vector at \(k\)-th time step |
| \(Y^{(k)}\) | Set of all measurement vectors, from the first to \(k\)-th time step |
| \(v^{(k)}\) | Vector of internal noise at \(k\)-th time step |
| \(n^{(k)}\) | Vector of measurement noise at \(k\)-th time step |
| \(f(\cdot)\) | Vector of transition functions |
| \(h(\cdot)\) | Vector of measurement functions |
| \(p(\cdot)\) | Probability density function |
| \(N_x\) | Number of state variables (state-space size) |
| \(N_p\) | Number of particles |
| \(N_f\) | Number of parallel filters |
| \(M\) | Number of time steps (length of the simulation) |
| \(\mathcal{N}(\mu, \sigma^2)\) | Normal distribution with mean value \(\mu\) and variance \(\sigma^2\) |
| \(U[a, b]\) | Uniform distribution with range from \(a\) to \(b\) |

state-space representation given by
\[
\begin{aligned}
x^{(k+1)} &= f(x^{(k)}, u^{(k)}; k) + v^{(k)} \\
y^{(k)} &= h(x^{(k)}) + n^{(k)}
\end{aligned}
\] (1)

where \(x^{(k)}\) denotes the state vector at the \(k\)-th time step, \(u^{(k)}\) is the input vector, \(y^{(k)}\) is the output vector, \(v^{(k)}\) and \(n^{(k)}\) are the vectors of process and measurement noises vectors, respectively, and \(f(\cdot)\) and \(h(\cdot)\) denote vectors of nonlinear functions. The main task of PF is to estimate the state vector based on the available measured outputs, known PDFs, and inflicted inputs to the system.

The PF operation principle is based on the recursive Bayesian filter [11] and is given by
\[
p \left( x^{(k)} | Y^{(k)} \right) = \frac{p \left( Y^{(k)} | x^{(k)} \right) p \left( x^{(k)} | Y^{(k-1)} \right)}{p \left( Y^{(k)} | Y^{(k-1)} \right)},
\] (2)

where \(Y^{(k)}\) is the set of output measurements from \(y^{(1)}\) to \(y^{(k)}\), \(p \left( x^{(k)} | Y^{(k)} \right)\) is the posterior PDF, \(p \left( y^{(k)} | x^{(k)} \right)\) is the likelihood, \(p \left( x^{(k)} | Y^{(k-1)} \right)\) is the prior PDF, and \(p \left( Y^{(k)} | Y^{(k-1)} \right)\) is the evidence given by equation
\[
p \left( y^{(k)} | Y^{(k-1)} \right) = \int p \left( y^{(k)} | x^{(k)} \right) p \left( x^{(k)} | Y^{(k-1)} \right) dx^{(k)}.
\] (3)

It is worth noting that the Bayesian filter, apart from the problem of online filtration presented in the paper, may also concern two other issues. These are smoothing, where the past state vector \(p \left( x^{(k)} | Y^{(k-1)} \right)\) is estimated, and prediction, where the goal is to estimate future state values \(p \left( x^{(k+1)} | Y^{(k)} \right)\), which are given by

2 Particle filter

This method can be used for discrete-time nonlinear objects (also for plants with non-Gaussian transition or measurement noises). The models can be written in modification proposed by the authors. The examined objects are presented in Sect. 4. In Sect. 5, one can see the obtained simulation results. Section 6 contains considerations about computational complexity of the proposed algorithm and performance comparisons for different devices. Section 7 constitutes the discussion, and the main conclusions are given in Sect. 8.

The most important symbols and shortcuts used in the article are described in Tables 1 and 2.
the formulas
\[
p\left( x^{(k)}| y^{(k+1)} \right) = \int \frac{p(x^{(k+1)}| y^{(k+1)})}{p(x^{(k+1)}| x^{(k)})} p\left( x^{(k+1)}| x^{(k)} \right) dx^{(k+1)},
\]
and
\[
p\left( x^{(k+1)}| y^{(k)} \right) = \int p\left( x^{(k+1)}| x^{(k)} \right) p\left( x^{(k)}| y^{(k)} \right) dx^{(k)}.
\]
Contrary to the filtering process, the above-mentioned issues are not used in real-time applications, but in post-
or pre-processing noise-smoothing applications [32–34].

In the PF algorithm, the posterior PDF is represented
by a set of particles. Each \( i \)-th particle consists of the
value \( x_i \) and the weight \( q^i \) for \( i = 1, 2, ..., N_p \), where \( N_p \) is the number of particles. Hence, the \( i \)-th particle
is represented by the pair \( \{ x_i, q^i \} \). When the parti-
cles number \( N_p \) is large enough, posterior PDF can be
approximated by
\[
p\left( x^{(k)}| y^{(k)} \right) \overset{N_p \to \infty}{\approx} \hat{p}\left( x^{(k)}| y^{(k)} \right) = \sum_{i=1}^{N_p} q_i^{i,(k)} \delta \left( x^{(k)} - x_i^{(k)} \right),
\]
where \( \delta (\cdot) \) is a Dirac delta function. The simplest and
first proposed particle filter method was the bootstrap
filter (BF) [11]. The algorithm of bootstrap filter is pre-
sented below.

**Algorithm 1. Bootstrap Filter**

1. Initialization. Draw \( N_p \) initial values \( x_i^{(0)} \sim p(x^{(0)}) \), set time step \( k := 1 \).
2. Prediction. Draw \( N_p \) new particles from the transition
model \( x_i^{(k)} \sim p(x^{(k)}| x_i^{(k-1)}) \).
3. Update. Calculate particles’ weights based on the
measurement model
\[
q_i^{i,(k)} = p( y^{(k)}| x_i^{(k)} ).
\]
4. Normalization. Scale values of the weights in such
a way that their sum equals 1.
5. Resampling. Draw \( N_p \) particles, but only from
existing ones. PDF for drawing is defined by the
set of normalized weights (in the current paper the
systematic resampling is used [35]).
6. End of the iteration. Calculate the estimate of the
state vector at \( k \)-th time step, update the time step
\( k := k + 1 \), go to Step 2.

The presented algorithm, named bootstrap filter, is a special case of sequential importance resampling (SIR)
[36], in which the resampling step is executed conditionally, when the effective sample size value, given by
\[
N_{eff} = \frac{1}{\sum_{i=1}^{N_p} \left( q_i^{i,(k)} \right)^2 },
\]
is not high enough. In the SIR update step, particles are
drawn from \( x_i^{(k)} \sim g(x^{(k)}| x_i^{(k-1)}, y^{(k)}) \), where \( g(\cdot) \)
is the importance function. In the PF algorithm pro-
posed here, the resampling is executed at every time
tep, and the importance function (from which particles
are drawn) is given by \( g(x^{(k)}| x_i^{(k-1)}, y^{(k)}) = p(x^{(k)}| x_i^{(k-1)}) \), so calculations in prediction and update
steps are simplified in relation to the SIR method.

More information about particle filters can be found in [36,37].

### 3 MultiPDF particle filter

In this paper, we propose a modification of the particle
filter—MultiPDF PF, which divides the main filter
into \( N_f \) parts—independent filters with \( \frac{N_p}{N_f} \) particles
each. These parallel filters are independent, and they
can communicate with each other (in general). At each
time step, the posterior PDF is calculated in all \( N_f \)
sub-filters independently, as well as estimated value.
The estimated value of the whole MultiPDF PF is cal-
culated as the mean of sub-filter results. After calcu-
lations, the \( N_f \) independent state variable results are
averaged (Fig. 1).

It is assumed that PDFs of each parallel sub-filter are
completely independent. This way, a set of estimated
state vectors is obtained. Based on equation (6), one can write
\[
\hat{p}\left( x^{(k)}| Y^{(k)} \right) = \frac{1}{N_f} \sum_{l=1}^{N_f} \hat{p}_l\left( x_l^{(k)}| Y^{(k)} \right),
\]
where \( \hat{p}_l(\cdot) \) is the estimated posterior PDF from the
\( l \)-th sub-filter. The estimated state vector \( x_l^{(k)} \) is calcu-
lated in every time step based on \( \hat{p}_l(\cdot) \).

For example, for a total particle number \( N_p = 500 \)
and dividing PF into \( N_f = 3 \) parallel parts, every \( l \)-th
sub-filter will contain \( N_{pl} = \text{round} \left( \frac{500}{3} \right) = 167 \)
particles.
Algorithm 2. MultiPDF Particle Filter

1. Initialization. For each parallel filter \( l = 1, \ldots, N_f \)
   draw \( N_{p[l]} = \text{round} \left( \frac{N_p}{N_f} \right) \) initial values \( x_{i[0]}^{[l]} \) from
   the initial PDF \( p(x^{(0)}) \), set time step \( k = 1 \).

2. Calculate \( N_f \) state vector values. For each parallel
   filter \( l = 1, \ldots, N_f \) perform the following steps:
   
   a. Prediction. Draw \( N_{p[l]} \) particles from the transition model
      \( x_{i[k]}^{[l]} \sim p(x^{(k)}|x_{i[k-1]}^{[l]}) \) (for every
      \( i = 1, \ldots, N_{p[l]} \)). The set of particle values is
      given for each parallel filter separately, from
      the previous time step.
   
   b. Update. Calculate the weights of particles based
      on the measurement model
      \( q_{i[k]}^{[l]} = p(y^{(k)}|x_{i[k]}^{[l]}) \).
      \( \text{(10)} \)
   
   c. Normalization. Normalize the weights so that
      their sum equals 1.
   
   d. Resampling. Draw \( N_{p[l]} \) particles from PDF
      obtained by the set of normalized weights (systematic
      resampling [35] has been used).
e. Calculate the estimated state vector $\hat{x}_{[l]}^{(k)}$ for the $l$-th independent sub-filter.

3. End of iteration. Calculate the estimate $\hat{\mathbf{x}}^{(k)}$ at $k$-th time step as the average of the $N_f$ estimated vectors obtained in Step 2, set $k = k + 1$, go to Step 2.

The operation of the method can be easily compared to tourists climbing to the peak, but without precise information on where the peak is located. (The position of the peak is the estimated value.) The group looks for the top, not on the basis of the ground slope (this is how the gradient methods would work), but randomly placing the participants of the expedition and finding which of them are higher than the others. On the other hand, if this group is divided into smaller parts, each sub-group will be worse at estimating the vertex individually, but if each group comes from a different slope, the average of their positions may give a better estimate than if they all worked together, within one group. In this comparison, it should be taken into account that this “climb” is one step, and this “peak” changes its position in each subsequent iteration (the state vector and thus the function $p(y^{(k)}|x^{(k)})$ whose vertex we are looking for).

Sub-filters realize the same algorithm, but within different particle’s sets. The only communication is averaging the estimate results from sub-filters at the end of each step, which does not require a lot of computation. It is worth mentioning that the communication used is one way—no sub-filter receives any information, and they only send information about the estimate to the master (averaging unit). This approach has a positive impact on the computational time over the PF, allowing for a linear acceleration of computations, which has been expanded on in Introduction. However, the authors do not reject the possibility of broader communication between sub-filters in the MultiPDF algorithm, but this has not been currently studied—it is a topic for future research.

It results from the research described in detail in Discussion section that for a two-dimensional system (with two state variables) in 85.46% of cases, the distance between the real and the estimated point $(\hat{x}_1^{(k)}, \hat{x}_2^{(k)})$ (on the trajectory for MultiPDF PF) is lower than the minimum obtained by the independently operating components (see Tables 6 and 7 for more details).

It can be also mentioned that in this work, in order to improve the estimation quality using parallelization of the particle filter, the object has to be multidimensional (the number of state variables must be greater than one). Also, the stronger the nonlinearity (especially in the measurement function), the better the estimation quality. The proposed algorithm allows one to minimize the degeneracy risk of all particles [36] (thanks to the division of the main filter), which often occurs in the mentioned plant types. The elements and threats mentioned above have been analyzed in Discussion section. Besides the results presented in the simulation part of this paper, additional studies were made on the influence of object factors on the estimation quality. See Appendix for the results.

The proposed solution is universal and allows one to speed up calculations in situations where there is limited computing power and time to perform the studies. This parallelization allows for an acceleration of calculations, for example, on GPU or FPGA [38–42], which is a current and much needed task. The special case of MultiPDF PF for $N_f = 1$ is the simple bootstrap particle filter.

### 4 Examined objects

Nine objects were considered, eight of which were multidimensional and all nonlinear. They were selected in such a way as to show the influence of various plant features on the operation of the proposed algorithm. For example, some of them have linear transition functions and some have linear measurement functions.

Most of the examined systems are plants without an input signal (autonomous systems), so the change of their state is only due to the presence of process noise. $Ob1$ is used very often in particle filters studies [10,36]. According to [43], this object was firstly proposed in 1978 by Netto, Gimeno and Mendes.

$$Ob1: \begin{cases}
    x^{(k+1)} = 0.5x^{(k)} + \frac{25x^{(k)}}{1+e^{x^{(k)}}} + 8\cos(1.2k) + v^{(k)} \\
    y^{(k)} = \frac{20x^{(k)}}{20} + n^{(k)} \\
    v^{(k)} \sim \mathcal{N}(0; 10), \quad n^{(k)} \sim \mathcal{N}(0; 1), \\
    x^{(0)} = 0.1.
\end{cases}$$

$Ob2$ is a multidimensional model proposed by the authors, with 5 state variables and 15 outputs. This plant shows the benefits of the parallelization of the particle filter.
Subsequent versions of the system

\begin{align*}
   x_i^{(k+1)} &= 0.5 x_i^{(k)} + v_i^{(k)}, \quad i = 1, \ldots, 5 \\
   y_i^{(k)} &= 4 x_i^{(k)} + \frac{x_i^{(k)}}{1+x_i^{(k)}+n_i^{(k)}} + 0.2 \sin \left( x_i^{(k)} + x_{i+2}^{(k)} \right) + n_i^{(k)}, \quad i = 1, \ldots, 5 \\
   y_{i+5}^{(k)} &= 4 x_i^{(k)} + \frac{x_i^{(k)}}{1+x_i^{(k)}+n_i^{(k)}}, \quad i = 1, \ldots, 5 \\
   y_{i+10}^{(k)} &= 2 x_i^{(k)} + 0.5 \left( x_{i+1}^{(k)} + x_{i+2}^{(k)} \right) + n_{i+10}^{(k)}, \quad i = 1, \ldots, 5
\end{align*}

*Ob2:*

\begin{align*}
   v_{i:5}^{(k)} &\sim \mathcal{N}(0; 1), \quad n_{i:5}^{(k)} \sim \mathcal{N}(0; 6), \\
   n_{i:10}^{(k)} &\sim \mathcal{N}(0; 4), \quad n_{i:15}^{(k)} \sim \mathcal{N}(0; 1), \\
   x^{(0)} &= [0 \ 0 \ 0 \ 0 \ 0]^T.
\end{align*}

*Ob3* has been proposed (in each of the following 5 versions) [44]. This object has been created in a network structure; the precursor for it is a power system grid. However, in power systems, every additional node (bus) is associated with two additional state variables. In the proposed network-type theoretical object, every node is related to only one additional state variable, which enables research on the system structure in a relatively short time (the number of particles \( N_p \) should grow exponentially with the number of state variables \( N_x \)).

\begin{align*}
   x_i^{(k+1)} &= 0.9 x_i^{(k)} + v_i, \quad i = 1, \ldots, 4 \\
   y_i^{(k)} &= h_i(x^{(k)}, \mu_{ij}) + n_i^{(k)}, \quad i = 1, \ldots, 4 \\
   y_{ij}^{(k)} &= h_{ij}(x^{(k)}, \mu_{ij}) + n_{ij}^{(k)}, \quad i = 1, \ldots, 4, \quad j = 1, \ldots, 4
\end{align*}

*Ob3:* (with different measurements) are as follows:

- (v1)
  \begin{align*}
   h_i(x^{(k)}, \mu_{ij}) &= x_i^{(k)}, \\
   h_{ij}(x^{(k)}, \mu_{ij}) &= x_i^{(k)} - x_j^{(k)}.
\end{align*}

- (v2)
  \begin{align*}
   h_i(x^{(k)}, \mu_{ij}) &= \prod_{j=1, n_{ij}^{(k)} \neq 0}^N x_j^{(k)}, \\
   h_{ij}(x^{(k)}, \mu_{ij}) &= x_i^{(k)} x_j^{(k)}.
\end{align*}

- (v3)
  \begin{align*}
   h_i(x^{(k)}, \mu_{ij}) &= x_i^{(k)} + \sum_{j=1, \mu_{ij} \neq 0}^{N_x} x_j^{(k)}, \\
   h_{ij}(x^{(k)}, \mu_{ij}) &= x_i^{(k)} + x_j^{(k)} \sin \left( x_i^{(k)} + x_j^{(k)} \right) x_j^{(k)},
\end{align*}

- (v4)
  \begin{align*}
   h_i(x^{(k)}, \mu_{ij}) &= \sum_{j=1, \mu_{ij} \neq 0}^{N_x} x_i^{(k)} x_j^{(k)} \mu_{ij}, \\
   h_{ij}(x^{(k)}, \mu_{ij}) &= x_i^{(k)} - x_j^{(k)} - \mu_{ij}, \\
   \cdot \cos \left( x_i^{(k)} - x_j^{(k)} - \mu_{ij} \right),
\end{align*}

- (v5)
  \begin{align*}
   h_i(x^{(k)}, \mu_{ij}) &= x_i^{(k)} \sum_{j=1, \mu_{ij} \neq 0}^{N_x} x_j^{(k)} \mu_{ij}, \\
   h_{ij}(x^{(k)}, \mu_{ij}) &= \max \left( x_i^{(k)} + x_j^{(k)} x_i^{(k)} x_j^{(k)} \right), \\
   \cos \left( \frac{x_i^{(k)} - x_j^{(k)}}{\ln \left( 1.01 + \left| x_j^{(k)} \mu_{ij} \right| \right)} \right), \\
   h_{ij}(x^{(k)}, \mu_{ij}) &= \frac{\max \left( x_i^{(k)} + x_j^{(k)} x_i^{(k)} x_j^{(k)} \right)}{0.01 + \min \{ A; B \}},
\end{align*}

\begin{align*}
   A &= \log_2 \left( 0.01 + \left| x_i^{(k)} x_j^{(k)} \right| \right), \\
   B &= -0.1 \left( x_i^{(k)} x_j^{(k)} \right)^2 + 2,
\end{align*}

\begin{align*}
   v_i^{(k)} &\sim \mathcal{N}(0; 1), \quad n_{i}^{(k)} \sim \mathcal{N}(0; 1), \quad n_{ij}^{(k)} \sim \mathcal{N}(0; 1), \\
   x_i^{(0)} &\sim \mathcal{U}[-1, 1] \text{ for each } i, j, \\
   \mu_{14} = \mu_{41} &= 0, \text{ for others } i, j \mu_{ij} = 1.
\end{align*}

All the transition and measurements functions for this plant can be found in [45] (where this object is referred to as “Ob403”).

*Ob4* is a two-dimensional system which was also proposed by the authors. This plant also enables one to show the improvement of the algorithm quality for
a higher number of parallel sub-filters.

\[
\begin{align*}
\dot{x}_1^{(k+1)} &= 0.5 \left( x_1^{(k)} \right)^3 + 0.01 x_2^{(k)} \\
&\quad + 0.2 u^{(k)} + v^{(k)} \\
\dot{x}_2^{(k+1)} &= 0.5 \left( x_2^{(k)} \right)^3 + 0.01 x_1^{(k)} \\
&\quad + 0.2 u^{(k)} + v^{(k)} \\
y_1^{(k)} &= 2x_1^{(k)}^2 + 0.004x_2^{(k)} + n^{(k)} \\
y_2^{(k)} &= 2x_2^{(k)}^2 + 0.004x_1^{(k)} + n^{(k)} \\
v^{(k)} &\sim \mathcal{N}(0; 1), \quad n^{(k)} \sim \mathcal{N}(0; 1), \\
u^{(0)} &= [0.1 \ 0.1]^T.
\end{align*}
\]

Figure 2 Ob5 is a magnetic levitation system (MLS) [46] produced by Inteco. The electromagnetic forces (caused by the voltages applied to the system inputs) change the position of the ball (sphere), which is one of the outputs of the system. The second measurement is the electromagnetic force from the upper magnet, which is given by a nonlinear function of the state variables.

The system is given by the continuous and strongly nonlinear state equations below.

\[
\begin{align*}
\dot{x}_1 &= x_2, \\
\dot{x}_2 &= -F_{em1} \frac{F_{em1}}{m} + g + \frac{F_{em2}}{m} \\
\dot{x}_3 &= \frac{F_{em2}}{f_1} e^{\frac{x_2}{f_1}} (k_1 u_1 + c_1 - x_3) \\
\dot{x}_4 &= \frac{F_{em2}}{f_2} e^{\frac{x_2}{f_2}} (k_2 u_2 + c_2 - x_4) \\
y_1 &= x_1 \\
y_2 &= x_2^2 \frac{F_{em1} \, e^{-x_1}}{F_{em2} \, e^{-x_2}} \\
F_1 &= x_3^2 \frac{F_{em1}}{F_{em2} \, e^{-x_2}} \quad \text{and} \quad F_2 = x_4^2 \frac{F_{em1}}{F_{em2} \, e^{-x_2}} \\
v_1^{(k)} &\sim \mathcal{N}(0; 10^{-6}) \quad \text{for each } i = 1, \ldots, 4, \\
n_2^{(k)} &\sim \mathcal{N}(0; 6 \cdot 10^{-5}), \quad n_2^{(k)} \sim \mathcal{N}(0; 1.6 \cdot 10^{-4}), \\
u_1^{(0)} &\sim \mathcal{N}(0; 10^{-3}), \\
x^{(0)} &= [0.015 \ 0 \ 0 \ 0]^T.
\end{align*}
\]

Object parameters are described in Table 3. For the simulation test, the model was discretized by the Euler extrapolation method with the sampling period \( T_s = 0.1 \text{ ms} \). Constraints on the system, also included in the simulation model, are as follows.

\[
0 \leq x_1(t) \leq x_{dm}
\]

Table 3 Parameters of the magnetic levitation system

| Parameter  | Value | Unit | Description |
|------------|-------|------|-------------|
| \( m \)   | 0.0571 | kg   | Ball mass   |
| \( g \)   | 9.81  | m/s² | Gravitation |
| \( F_{em1} \) | 1.7521 \times 10^{-2} | H | Inductance |
| \( F_{em2} \) | 5.8231 \times 10^{-3} | H | Inductance |
| \( f_1 \) | 1.4142 \times 10^{-4} | ms | Construction constant |
| \( f_2 \) | 4.5626 \times 10^{-3} | m | Construction constant |
| \( k_1 \) | 2.5165 | A | Construction constant |
| \( k_2 \) | 0.0243 | A | Construction constant |
| \( d \) | 0.075 | m | Distance between the magnets |
| \( b_d \) | 0.06 | m | Ball diameter |
| \( x_d \) | 0.015 | m | \( d - b_d \) |

\[
0.03884A \leq x_3(t), \quad x_4(t) \leq 2.38A \\
0.00498 \leq u_1(t), \quad u_2(t) \leq 1
\]

5 Simulation results

The aim of this part of the paper is to compare the performance quality of the proposed MultiPDF PF with several known particle filter algorithms (bootstrap and marginalized). Simulations and calculations were performed for nine objects and repeated many times to obtain accurate results. Each time the ground truth was generated separately, and the adjustment of the estimation results was checked. For Ob1, simulations were repeated 1400 times, for Ob2—at least 100 times, and for the remaining systems, the tests were repeated several thousand times. The different number of simulations is a consequence of the specific construction of each object and was chosen to obtain the appropriate accuracy of the simulation results. Simulations were performed using various numbers of particles \( N_p \) and parallel filters \( N_f \)—the results were compared for those parameters.

In this paper, the average root-mean-square error (aRMSE) quality index has been used, which is given by

\[
aRMSE = \frac{1}{N_x} \sum_{j=1}^{N_x} \text{RMSE}_j,
\]

\( \sum_{j=1}^{N_x} \text{RMSE}_j \)
RMSE_j = \sqrt{MSE_j}, \quad (12)

MSE_j = \frac{1}{M} \sum_{k=1}^{M} \left( \hat{x}_j^{(k)} - x_j^{+(k)} \right)^2, \quad (13)

where M is the number of time steps, N_x is the number of state variables, and \( \hat{x}_j^{(k)} \) and \( x_j^{+(k)} \) are the estimated and the real value of the j-th state variable, respectively, obtained at the k-th time step.

Additionally, for objects containing linear state variables in the transition function (Ob2, Ob3 and Ob5), the operation of the above-mentioned algorithms was compared with marginalized PF [19, 21]. The analysis of the computational complexity of the algorithm is presented in [47]. MPF enables one to use fewer particles than PF for the same, multidimensional system.

The simulation results (quality indices) are presented in Figs. 3, 4, 5, 6, 7, 8, 9, 10, 11. Standard deviations with 95% probability in both directions from the mean, according to the 68-95-99.7 rule, are marked on the graphs. Standard deviations were estimated based
on the well-known square root law which states that for a Gaussian PDF, the standard deviation of the mean value is $\sqrt{m}$ times smaller than the standard deviation from $m$ observations. The results of an MPF operation for $Ob2$ and $Ob3$ are presented in Table 4 (the results have not been presented in the figures, because the values exceeded the chart’s ranges). Since all state variables in the transition function are linear, the performance of MPF is independent of the number of particles [21], and hence, only one value is presented for these plants. In the case of $Ob5$, the performance quality was also examined under the influence of particle number, and the results are shown in Fig. 11.
6 Performance analysis

The algorithms 1 and 2 were thoroughly analyzed for Ob1. Firstly, the theoretical complexity of both was calculated. Then, after implementation of the algorithms in the C# programming language, the performance on four independent hardware platforms was tested. The goal was to compare computational speed of algorithms and determine how the number of threads affects the speed.

The complexity was calculated using the most common method called big O notation [48]. The method allows one to describe the asymptotic behavior of a function or a set of functions. The conducted calculations have shown that both algorithms have the same theoretical complexity, i.e., $O(n^2)$. In this case, dissimilarities are not noticeable, due to the feature of
Fig. 9 Values of aRMSE for Ob3, version 5 with a variable number of particles $N_p$ and for a different number of parallel sub-filters $N_f$.

Fig. 10 Values of aRMSE for Ob4 with a variable number of particles $N_p$ and for a different number of parallel sub-filters $N_f$.

dedicated for multithreaded processors, tests were performed using 1, 2, 3, 5, 7 and 10 threads. The platform details are shown in Table 5. For each experiment, Microsoft Windows operating systems based on x64 architecture were used.

The results of the performed tests are shown in Figs. 12 and 13. Each value was calculated as an average of 10 tests. 12 tests per single scenario were conducted, and the best and the worst time was excluded due to
Fig. 11 Values of aRMSE for \( \text{Ob5} \) with a variable number of particles \( N_p \) and for a different number of parallel sub-filters \( N_f \); additionally, the results of the MPF’s operation are shown.

possible anomalies. As Platform D was much slower than others, it was decided to use a secondary axis (on the right) where values are correlated with the gray bars. It makes charts much easier to read.

Figure 12 presents results for \( N_p = 1200 \) particles, and Fig. 13—for \( N_p = 1500 \) particles. The simulations were performed for the same number of threads and the number of \( N_f \), marked in the graphs as “T”. As one can see, the computational time for PF and MultiPDF (1T) PF for the same platforms is similar, and both of them are using only one thread. With the increasing number of threads, one can observe a reduction in computational time. While working with more than 7 threads, computational time is not reduced, and even extended in a few cases. The situation is directly related to processors, especially with the numbers of cores and threads. Platforms B, C and D offer only 4 threads (or cores). Platform A offers up to 16 threads. If one is trying to use more threads than physically or logically available, the system wastes a lot of time changing active tasks. In the literature, it is called context switching [49].

Based on the obtained results, the authors can confirm that their algorithm is well optimized, it is scalable and it could reduce computational time up to 7 times, when used with multithreaded systems. In that sense, a calculation which was done within 7 days can now be done in just 1 day. What’s more, the benefits of the parallelization are noticeable even using older hardware.

Results are not so impressive as in the case of modern hardware, but the algorithm is still up to 2.5 times faster than the PF version.

7 Discussion

Based on the simulation results, one can see that division of the PF model into several parallel filters can improve the quality of estimation, but not for all cases. For \( \text{Ob1} \), increasing \( N_f \) causes a deterioration of the estimation quality. For multivariable plant \( \text{Ob2} \), division into a few parallel parts improves estimation quality. \( \text{Ob3} \) is a specific plant, with a simple linear transition function and nonlinear measurement function (five different types). For versions (v1)-(v3), the best estimation quality was provided by the division into a few sub-filters (see Figs. 5, 6, 7). The most interesting results were obtained for versions (v4) and (v5), because for these versions of MultiPDF PF, a lot of parallel filters still improve the estimation quality (see Figs. 8, 9). For version (v4), \( N_f = 10 \) and higher values provide the superior effect (similar for each \( N_f \) value). For version (v5), the best quality is obtained for \( N_f = 10 \), and an increase or decrease in this parameter causes deterioration of the estimation quality. One can infer that for more complicated measurement functions (\( \text{Ob3} \), and especially (v4), with the state variable multiplication
and trigonometric functions) the PF algorithm dividing into a higher number of parallel filters works better. For (almost) every case, the increase of the number of particles for an unchanged $N_f$ value causes an improvement in the estimation quality.

The results obtained for the last systems, named $Ob4$ and $Ob5$, show benefits from the proposed PF modification. As one can see in Figs. 10 and 11, the higher the $N_p$ parameter, the better the estimation quality (i.e., the lower value of the $aRMSE$ index). This may be caused by the strong nonlinear character of these plants and their multidimensionality. For a higher (than one)-dimensional system, there are a lot of time steps when the estimation quality deteriorates, because of a long distance between the estimated and real values. Using more than one sub-filter, there is always a chance for error compensation by sub-filters resulting in the estimated values being closer to the real values.

Additional research has been carried out for $Ob4$, for simulations with $N_p = 300$, $N_f = 3$ (each parallel filter with 100 particles) and $M = 100000$. In every time step ($k = 1, ..., M$), the authors examined mean values and variances of the distances between the estimated and real values given by

$$d^{(k)} = \sqrt{\left(x_1^{(k)} - x_1^{+(k)}\right)^2 + \left(x_2^{(k)} - x_2^{+(k)}\right)^2}. \quad (14)$$

Distances were calculated for three parallel filters ($PF[1]$, $PF[2]$ and $PF[3]$), for whole MultiPDF PF and for the traditional bootstrap filter (specifically MultiPDF PF with $N_f = 1$). As one can see in Table 6, the results from parallel filters provided higher mean values and standard deviations than from single ones, but statistics for estimate from MultiPDF PF are noticeably smaller. This is caused by the compensation of the larger distances by the results from other sub-filters with a better fit. The bootstrap filter provided somewhat worse results, because there is no possibility of the wrong estimated values being compensated by the other components, as in MultiPDF PF with a parameter $N_f$ higher than 1.

Table 7 presents the percent of cases, in which the distance of the MultiPDF PF estimated value given by (14) is smaller than for the special cases (named “events”) during one simulation with $M = 100000$. A comparison of distances for MultiPDF and BF slightly favors BF, but as one can see in Table 6, the mean distance for MultiPDF PF is generally smaller than for BF. These results show the potential benefits with the possibility of the compensation of wrong values. As one can see, almost always the result from one of the parallel filters is farther from the true value than in the case of the MultiPDF PF result.

Most of Figs. 3, 4, 5, 6, 7, 8, 9, 10, 11 show that for a high number of particles, standard deviations for specific number of particles overlap. This is due to the very small distance between these results on the graph. For versions (v4) and (v5) of $Ob3$, the distance between adjacent charts is relatively large for many particles. This is probably caused by the strong nonlinearity of these objects. To make results similar to each other for a different $N_f$ parameter, one must make calculations for a much larger number of particles. In Fig. 8 and for the highest $N_p$, an interesting phenomenon can be observed: as the number of particles increases, the quality of estimation worsens. This may be caused by the distance between the subsequent noised measurements. The more particles, the greater is the chance that noisy measurements correspond and will be recognized by the algorithm as correct, and hence, they will provide a higher quality index.

Regarding the marginalized PF simulation results (Table 4), it can be shown that the estimation quality was improved only for $Ob3$ in version (v1), because of linear measurement function of this plant. Only for this plant, the quality index in Table 4 is smaller than most

### Table 4 Results of marginalized particle filter simulations ($aRMSE$ mean values from 1000 simulations)

| $Ob2$ | $Ob3, v1$ | $Ob3, v2$ | $Ob3, v3$ | $Ob3, v4$ | $Ob3, v5$ |
|-------|-----------|-----------|-----------|-----------|-----------|
| 0.8156 | 0.5053    | 1.6871    | 1.0198    | 1.7121    | 2.5061    |

### Table 5 Specification of the hardware platforms

| Platform | Processor | Memory | Windows version |
|----------|-----------|--------|-----------------|
| A        | AMD Ryzen 7 2700 | DDR4/64 GB | 10              |
| B        | Intel Core i5 4210M | DDR3/8 GB | 8.1             |
| C        | Intel Core i7 Q720 | DDR3/8 GB | 10              |
| D        | Intel Atom D525 | DDR3/4 GB | 7               |
of the values presented in the corresponding Fig. 5. Due to the Kalman filtration, which was used instead of particles draw, this method does not provide good results for highly nonlinear objects (the rest of the considered systems have a strongly nonlinear measurement function). For other objects, the MPF result is worse than the worst results from MultiPDF PF (higher aRMSE values than all values visible in the graphs). Therefore, in some cases, speeding up the calculations using MPF may result in a significant deterioration of the estimation quality.

Ob5 is the system with one linear and three nonlinear state variables. The MPF results in this case show a slight improvement over BPF (the Kalman filter works better for linear systems), but with paralleliz-
tion the MultiPDF PF still works best. The average simulation time measured for 1500 particles was 6.29 s for PF and 5.72 s for MPF, so the time gain was small and could be compensated for by MultiPDF PF with multithreaded implementation.

8 Conclusions

The main conclusion of this article is that the proposed particle filter can improve the quality of estimation and its multithreaded implementation can significantly speed up calculations. The main condition for improving estimation quality is the presence of strong nonlinearities in the measurement function of the object (detailed conditions are presented in section MultiPDF Particle Filter and Appendix, additionally). For these objects types, there is a bigger chance for a wrong estimation, so large errors from one sub-filter can be compensated by the other units. For special object types, even a lot of parts make the quality index smaller. In general, a trade-off must be found between the speed of computation and the quality of the estimation, because all real-world applied systems are specific and different.

In most cases, an improvement in the estimation can be observed for smaller sums of particles, while for large sums there is no difference (or it is negligible) in the quality of the estimation. It is possible to further improve the quality of the estimation (as can be seen on the right-hand side of the graphs), but the additional computational effort has a relatively small impact on the result. At the same time, it can be clearly observed that improving the quality of estimation by using MultiPDF PF is possible especially for a relatively small number of particles, while according to the theory, when $N_p \to \infty$, the specific PF method does not matter.

In the vein of this method, it is also possible to further parallelize the PF algorithm (in every sub-filter), because the bottlenecks are normalization and resampling. Normally, they must be carried out using information about all particles. (In the first case, the sum of all weights is needed; in the second case, even using the fastest resampling, it is necessary to act in sequence, not parallel, while parallel resampling algorithms deteriorate the estimation quality [35].)

Data Availability  Data with saved simulation results are available on the website https://drive.google.com/drive/folders/1KdVh3vgxElqmAPH6P4AUvY0PyTzd9T?usp=

Declarations

Conflicts of interest  The authors declare that they have no conflict of interest

Open Access  This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article’s Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article’s Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

Appendix

Additional research on the effect of object properties on the estimation quality has been carried out. For the Ob4 object, which enables one to show improvement of estimation quality by the MultiPDF PF, a number of modifications (named Ob4X) have been proposed. The operation of the algorithm checked for, among others, separated state variables and various types of nonlinearities in transition or measurement functions, or in both of them. The test results with the exact specification of the objects have been included in the additional file 1.

Based on the results, it can be concluded that the condition for improving the quality of the estimation is, first of all, the presence of nonlinearities in the measurement function. Moreover, too strong an influence of the linear part may deteriorate the quality of the estimation, which can be seen in the cases of object Ob44 and Ob46. The type of the transition function (linear or nonlinear) has no influence on the obtained results, and neither does the separation of the state variables.

1  https://drive.google.com/file/d/1C7GhqmfibIx8SIw7lkcFxUKLS-LLY09d/view
References

1. Abur, A., Exposito, A.G.: Power System State Estimation: Theory and Implementation, pp. 17–49. Marcel Dekker Inc., New York (2004). https://doi.org/10.1201/9780203913673.ch2
2. Udupa, H.N., Minal, M., Mishra, M.T.: Node level ANN technique for real time power system state estimation. Int. J. Sci. Eng. Res. 5(1), 1500–1505 (2004)
3. Zawirski, K., Deskur, J., Kaczmarek, T.: Automatic of electric drive (in Polish). Poznan University Publisher, Poznan (2012)
4. Zhang M., Wang K., Zhou Y. T.: Online state of charge estimation of lithium-ion cells using particle filter-based hybrid filtering approach. Complexity (2020)
5. Weiss, S., Achtelik, M.W., Lyen, S., Achtelik, M.C., Kneip, L., Chli, M., Siegwart, R.: Monocular vision for long-term micro aerial vehicle state estimation: a compendium. J. Field Robot. 30(5), 803–831 (2013)
6. Marantos, P., Koveos, Y., Kyriakopoulos, K.J.: UAV state estimation using adaptive complementary filters. IEEE Trans. Control Syst. Technol. 24(4), 1214–1226 (2016)
7. Mukhina M., Pymak A.: Resampling errors of particle filtering in correlation-extreme navigation of UAV by belief field. In: Proceedings of the 2019 IEEE 5th International Conference Actual Problems of Unmanned Aerial Vehicles Developments (APUAVD), pp 61–66, IEEE (2019)
8. Askar, W.A., Elmowafy, O., Raiescu, A., Youssif, A.A., Elmasar, G.A.: Occlusion detection and processing using optical flow and particle filter. Int. J. Adv. Intell. Parad. 15(1), 63–76 (2020)
9. Moshtagh N., Romberg P. M., Chan M. W.: Projective particle filter for multi-sensor fusion, US Patent No. 10,535,144. Washington, DC: US Patent and Trademark Office (2020)
10. Gordon, N.J., Salmond, D.J., Smith, A.F.M.: Novel approach to nonlinear/non-Gaussian Bayesian state estimation. IEE Proc. F 140(2), 107–113 (1993). https://doi.org/10.1049/ip-f-2.1993.0015
11. Candy, J.V.: Bayesian Signal Processing, pp. 36–44. Wiley, Hoboken (2009). https://doi.org/10.1007/9780470340583
12. Michalski, J., Kozierski, P., Zietkiewicz, J.: Comparison of particle filter and extended Kalman particle filter. Stud. Autom. Inform. Technol. 42(4), 43–51 (2017)
13. Chen, H., Liu, X., She, C., Yao, C.: Power system dynamic state estimation based on a new particle filter. Proc. Environ. Sci. 11, 655–661 (2011). https://doi.org/10.1016/j.proenv.2011.12.102
14. Ke, L., Jingjing, W., Lei, S., Peng, C.: New particle filter based on GA for equipment remaining useful life prediction. Sensors 17(4), 696 (2017)
15. Yin, S., Zhu, X.: Intelligent particle filter and its application to fault detection of nonlinear system. IEEE Trans. Ind. Electron. 62(6), 3852–3861 (2015)
16. Zhou, H., Deng, Z., Xia, Y., Fu, M.: A new sampling method in particle filter based on Pearson correlation coefficient. Neurocomputing 216, 208–215 (2016)
17. Su, X., Wang, S., Pech, M., Zhao, L., Ye, Z.: Interacting multiple model particle filter for prognostics of lithium-ion batteries. Microelectron. Reliab. 70, 59–69 (2017)
18. Kumar, H., Mishra, A.: Feedback particle filter based image denoiser. In: Proceedings of the 2017 International Conference on Recent Innovations in Signal processing and Embedded Systems (RISE), pp 320–325. IEEE (2017)
19. Saha, S., Ökan, E., Gustafsson, F., Smiddl, V.: Marginalized particle filters for Bayesian estimation of Gaussian noise parameters. In: Proceedings of the 2010 13th International Conference on Information Fusion. IEEE, pp 1–8 (2010)
20. Ribeiro, M.I.: Kalman and extended Kalman filters: concept, derivation and properties. Inst. Syst. Robot. 43, 46 (2004)
21. Ebingre, B., Bouaynaya, N., Georgieva, P., Mihaylova, L.: EEE dynamic source localization using marginalized particle filtering. In: Proceedings of the 2015 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), IEEE, pp. 454–457 (2015)
22. Yang, X., Wang, M., Qiao, Y., Zhang, B., Yang, Y.: Efficient marginalized particle smoother for indoor CSS-TOF localization with non-gaussian errors. Remote Sens. 12(22), 3838 (2020)
23. Brun, O., Teuliere, V., Garcia, J.M.: Parallel particle filtering. J. Paral. Distrib. Comput. 62(7), 1186–1202 (2002)
24. Martinov, L., Read, J., Elvira, V., Louzada, F.: Cooperative parallel particle filters for online model selection and applications to urban mobility. Digital Signal Process. 60, 172–185 (2017)
25. Murray, L.M., Lee, A., Jacob, P.E.: Parallel resampling in the particle filter. J. Comput. Graph. Stat. 25(3), 789–805 (2016)
26. Vergé, C., Dubarry, C., Del Moral, P., Moulines, E.: On parallel implementation of sequential Monte Carlo methods: the island particle model. Stat. Comput. 25(2), 243–260 (2015)
27. Liu, S., Mingas, G., Bouganis, C. S.: Parallel resampling for particle filters on FPGAs. In: Proceedings of the 2014 International Conference on Field-Programmable Technology (FPT), IEEE, pp 191–198 (2014)
28. Nicely, M.A., Wells, B.E.: Improved parallel resampling methods for particle filters. IEEE Access 7, 47593–47604 (2019)
29. Crisan, D., Miguez, J., Rios-Muñoz, G.: On the performance of parallelisation schemes for particle filtering. EURASIP J. Adv. Signal Process. 2018(1), 1–18 (2018)
30. Wang, X., Li, T., Sun, S., Corchado, J.M.: A survey of recent advances in particle filters and remaining challenges for multitarget tracking. Sensors 17(12), 2707 (2017)
31. Michalski, J., Kozierski, P., Zietkiewicz, J.: Comparison of state estimation methods for dynamical systems (in Polish). Meas. Autom. Robot. 21(4), 41–47 (2017)
32. Qian, N., Chang, G., Gao, J.: Smoothing for continuous dynamical state space models with sampled system coefficients based on sparse kernel learning. Nonlinear Dyn. 100(4), 3597–3610 (2020)
33. Dunik, J., Straka, O.: Design of Rao–Blackwellized pointmass smoother for conditionally linear and Gaussian models. IEEE Trans. Signal Process. 67(23), 6053–6066 (2019)
34. Jacob, P.E., Lindsten, F., Schön, T.B.: Smoothing with couplings of conditional particle filters. J. Am. Stat. Assoc. 115(530), 721–729 (2020)
35. Kozierski, P., Lis, M., Zietkiewicz, J.: Resampling in particle filtering—comparison. Stud. Autom. Inform. Technol. 38, 35–64 (2013)
36. Arulampalam, S., Maskell, S., Gordon, N., Clapp, T.: A tutorial on particle filters for on-line non-linear/non-Gaussian Bayesian tracking. IEEE Trans. Signal Process. 50(2), 174–188 (2002). https://doi.org/10.1109/78.978374
37. Doucet, A., Johansen, A. M.: A tutorial on particle filtering and smoothing: fifteen years later. Handbook of Nonlinear Filtering 2009/12, pp 656–704 (2009)
38. Mountney, J., Obeid, I., Silage, D.: Modular particle filtering FPGA hardware architecture for brain machine interfaces. In: Conference Proceedings of the IEEE Engineering Medicine Biology Society, pp 4617–4620, Boston, Massachusetts USA (2011)
39. Sutharsan, S., Kirubarajan, T., Lang, T., McDonald, M.: An optimization-based parallel particle filter for multitarget tracking. IEEE Trans. Aerosp. Electron. Syst. 48(2), 1601–1618 (2012)
40. Caulk, R.A., Catalano, E., Chareyre, B.: Accelerating yades poromechanical coupling with matrix factorization reuse, parallel task management, and GPU computing. Comput. Phys. Commun. 248, 106991 (2020)
41. Inam O., Basit A., Qureshi M., Omer, H.: FPGA-based hardware accelerator for SENSE (a parallel MR image reconstruction method). Comput. Biol. Med. 103598 (2020)
42. Psarakis, M., Dounis, A., Almabrok, A., Stavrinidis, S., Gkekas, G.: An FPGA-based accelerated optimization algorithm for real-time applications. J. Signal Process. Syst. 1, 22 (2020)
43. Kitagawa, G.: Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. J. Comput. Graph. Stat. 5(1), 1–25 (1996)
44. Kozierski, P., Michalski, J., Sadalla, T., Giernacki, W., Zietkiewicz, J., Drgas, S.: New grid for particle filtering of multivariable nonlinear objects. In: Proceedings of the 2018 federated conference on computer science and information systems (FedCSIS), 15, 1073–1077 (2018)
45. Kozierski, P., Michalski, J., Zietkiewicz, J., Retinger, M., Giernacki, W.: A new network for particle filtering of multivariable nonlinear objects. Energies 13(6), 1355 (2020). https://doi.org/10.3390/en13061355
46. Magnetic Levitation System User’s Manual, Inteco Ltd
47. Karlsson, R., Schon, T., Gustafsson, F.: Complexity analysis of the marginalized particle filter. IEEE Trans. Signal Process. 53(11), 4408–4411 (2005)
48. Rubinstein-Salzedo, S.: Big O notation and algorithm efficiency. In: Cryptography. Springer, Cham, pp 75–83 (2018)
49. Iwasaki, S., Amer, A., Taura, K., Balaji, P.: Analyzing the performance trade-off in implementing user-level threads. IEEE Trans. Parallel Distrib. Syst. 31(8), 1859–1877 (2020)

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.