Jacek MICHALSKI[1], Piotr KOZIERSKI[2,1], Joanna ZIĘTKIEWICZ[1]

Poznan University of Technology, Faculty of Electrical Engineering, Institute of Control, Robotics and Information Engineering, Division of Control and Robotics (1)

Poznan University of Technology, Faculty of Computing, Institute of Automation and Robotics, Division of Electronic Systems and Signal Processing (2)

doi:10.15199/48.2018.12.19

Comparison of Auxiliary and Likelihood Particle Filters for State Estimation of Dynamical Systems

Abstract. *In this paper, algorithms of the state estimation of dynamical systems, using different types of particle filters, have been presented. Three Particle Filter methods have been used: Bootstrap Filter, Auxiliary Particle Filter and Likelihood Particle Filter. These methods have been applied to two nonlinear objects, with quadratic measurement functions. The results have been additionally compared with the outcome from Kalman filters. Based on the obtained results (5 different quality indices) the estimation methods have been evaluated.*

Streszczenie. *W niniejszej pracy zostały przedstawione algorytmy estymacji stanu układów dynamicznych za pomocą różnych rodzajów filtrów cząsteczkowych. Zaprezentowano trzy metody filtrów cząsteczkowych: algorytm Bootstrap, pomocniczy filtr cząsteczkowy i wiarygodny filtr cząsteczkowy. Metody te zastosowano dla dwóch obiektów nieliniowych o kwadratowych funkcjach pomiarowych. Z filtrami cząsteczkowymi zostały dodatkowo zestawione metody filtru Kalmana. Na podstawie uzyskanych wyników (5 różnych wskaźników jakości) metody estymacji zostały ocenione. (**Porównanie filtrów cząsteczkowych, pomocniczego i wiarygodnego, do estymacji stanu systemów dynamicznych**).*

Keywords: particle filters, state estimation, dynamical systems, Kalman filters, nonlinear plants.
Słowa kluczowe: filtry cząsteczkowe, estymacja stanu, układy dynamiczne, filtry Kalmana, obiekty nieliniowe.

Introduction

State estimation is incredibly useful in practical applications and dynamically developing branch of the science. Estimation methods have broad applications in various fields of technical sciences, e.g. in power systems to estimate hard-measurable phasors [1-3], in electric drive to obtain easy measurable quantities: current and voltage and estimate velocity and position of the motor shaft [4], in analyzing UAVs moving (estimation of UAV position on the map) [5-6] as well as in mobile robots movement, to estimate position or linear velocity [7].

In 1993, Gordon, Salmond and Smith proposed the first Particle Filter (PF) algorithm [8]. It was not widely used despite its effectiveness, due to the high computational complexity. However, in further years the computing power of processors has significantly increased, so today it is easy to apply the PF algorithm and its more computationally complex modifications, even in online estimation. Nowadays, dynamic estimation methods are still being developed, mainly modifications of existing ones, such as the Remaining Useful Life Particle Filter (RULPF) [9].

In this paper, three Particle Filters have been explained and compared. The authors of the article have focused on particle filters, because they are designed to estimate noises of any probability density function and they work well with nonlinear objects. The simulations results have been compared with two variants of the Kalman filter (Extended and Unscented), of which explanations one can found in [10, 11]. Despite the fact that the original Kalman filter did not apply Bayes' rule [12], it has been proven that Kalman's equations can be derived from Recursive Bayessian Filter (see (2) in the section *Particle filters algorithms*). Times of computing also have been compared.

In the first section, formulation of a problem has been described. The second section contains description of particle filters algorithms. Models of used objects and quality indices have been described in Sections 3 and 4, respectively. Section 6 contains the results of the simulations. Conclusions and final comments one can find in Section 7.

Formulation of the problem

The discrete system described in state space is considered by

$$\begin{cases} \boldsymbol{x}^{(k+1)} = \boldsymbol{f}\left(\boldsymbol{x}^{(k)}, \boldsymbol{u}^{(k)}; k\right) + \boldsymbol{v}^{(k)} \\ \boldsymbol{y}^{(k)} = \boldsymbol{h}\left(\boldsymbol{x}^{(k)}\right) + \boldsymbol{n}^{(k)} \end{cases} \tag{1}$$

where: $\boldsymbol{x}^{(k)}$ – state vector in k-th time step, $\boldsymbol{u}^{(k)}$ – input vector, $\boldsymbol{y}^{(k)}$ – measurement vector, $\boldsymbol{v}^{(k)}$ – process noise vector, $\boldsymbol{n}^{(k)}$ – measurement noise vector, $\boldsymbol{f}(\cdot)$ and $\boldsymbol{h}(\cdot)$ – vectors of transition and measurement functions respectively.

The task of the estimation is to reconstruct values of the state variables based on the available measurement outputs and known inputs of the system.

Particle filter algorithms

1) Generic Particle Filter

This method (and all next too) was designed for applying to nonlinear objects (1). In the literature it occurs also as Sampling Importance Resampling (SIR). The principle of the Particle Filter operation is based on the Bayes filter [13], described by equation

$$p\left(\boldsymbol{x}^{(k)} \mid \boldsymbol{Y}^{(k)}\right) = \frac{p\left(\boldsymbol{y}^{(k)} \mid \boldsymbol{x}^{(k)}\right) p\left(\boldsymbol{x}^{(k)} \mid \boldsymbol{Y}^{(k-1)}\right)}{p\left(\boldsymbol{y}^{(k)} \mid \boldsymbol{Y}^{(k-1)}\right)}, \tag{2}$$

where: $\boldsymbol{Y}^{(k)}$ – set of measurements from all k time steps, $p(\boldsymbol{x}^{(k)}|\boldsymbol{Y}^{(k)})$ – posterior Probability Density Function (PDF), $p(\boldsymbol{y}^{(k)}|\boldsymbol{x}^{(k)})$ – likelihood, $p(\boldsymbol{x}^{(k)}|\boldsymbol{Y}^{(k-1)})$ – prior PDF, $p(\boldsymbol{y}^{(k)}|\boldsymbol{Y}^{(k)})$ – evidence ratio.

In the PF algorithm PDF is represented by a set of particles. Each particle consists of a state vector and the weight value. So i-th particle can be described as a pair $\{x^i, q^i\}$. If the number of particles is high enough, posterior PDF may contain the same information as the continuous PDF function.

$$p\left(\mathbf{x}^{(k)} \mid \mathbf{Y}^{(k)}\right) \overset{N_p \to \infty}{=} \hat{p}\left(\mathbf{x}^{(k)} \mid \mathbf{Y}^{(k)}\right) = \sum_{i=1}^{N_p} q^{i,(k)} \delta\left(x^{(k)} - x^{i,(k)}\right), \tag{3}$$

where: $\delta(\cdot)$ – Dirac delta, N_p – number of particles.

Algorithm 1: SIR Particle Filter

1. Initialization. Draw initial values of particles $x^{i,(0)} \sim p(x^{(0)})$, set $k=1$.
2. Prediction. Draw N_p new particles based on transition model $\boldsymbol{x}^{i,(k)} \sim p(\boldsymbol{x}^{(k)}|\boldsymbol{x}^{i,(k-1)}, \boldsymbol{y}^{(k)})$.
3. Update. Compute particle weights based on the measurement model:

$$q^{i,(k)} = q^{i,(k-1)} \frac{p\left(\boldsymbol{y}^{(k)} \mid \boldsymbol{x}^{i,(k)}\right) p\left(\boldsymbol{x}^{i,(k)} \mid \boldsymbol{x}^{i,(k-1)}\right)}{p\left(\boldsymbol{x}^{i,(k)} \mid \boldsymbol{x}^{i,(k-1)}, \boldsymbol{y}^{(k)}\right)}. \tag{4}$$

4. Normalization. Scale the weights in such a way that their sum be equal to 1.
5. Calculate the effective sample size

$$N_{eff} = \frac{1}{\sum_{i=1}^{N_p} \left(q^{i,(k)}\right)^2}. \tag{5}$$

If $N_{eff} < N_T$, (where N_T, is a critical value, mostly $N_T = N_p/2$), go to step 6, otherwise go to step 7.

6. Resampling. Described at the end of the Section.
7. End of the iteration. Calculate the estimate for k-th time step, update $k = k + 1$, go to step 2.

If the resampling stage is executed unconditionally, in each time step, one can skip the multiplication by $q^{i,(k-1)}$ in the update model for weights calculation (because after the resampling stage the weights are equal to N_p^{-1}).

The main disadvantage of particle filters is time needed for calculations, because PFs algorithms are strongly computationally complex. The number of calculations grow exponentially with the number of state variables. However, PF algorithms are optimal for nonlinear and non-Gaussian plants (due to Bayessian based solution).

More about particle filters one can find in [14, 15].

2) Bootstrap Filter

The Bootstrap Filter (BF) was proposed in [8] by Gordon, Salmond and Smith and assumes drawing particles in prediction step from the transition model $\boldsymbol{x}^{i,(k)} \sim p(\boldsymbol{x}^{(k)}|\boldsymbol{x}^{i,(k-1)})$. It is also assumed that the resampling is executed in every iteration.

Algorithm 2: Bootstrap Filter

1. Initialization. Draw initial values of particles $x^{i,(0)} \sim p(x^{(0)})$, set $k=1$.
2. Prediction. Draw N_p new particles from transition model $\boldsymbol{x}^{i,(k)} \sim p(\boldsymbol{x}^{(k)}|\boldsymbol{x}^{i,(k-1)})$.
3. Update. Draw weights of the particles from measurement model

$$q^{i,(k)} = p\left(\boldsymbol{y}^{(k)} \mid \boldsymbol{x}^{i,(k)}\right). \tag{6}$$

4. Normalization. Scale the weights in such a way that their sum be equal to 1.
5. Resampling. Described at the end of the Section.
6. End of the iteration. Calculate the estimate of k-th time step, actualize time step $k = k + 1$, go to step 2.

In BF algorithm the particles are drawn in the prediction stage from the transition model, and in the update step the denominator $p(\boldsymbol{x}^{(k)}|\boldsymbol{x}^{i,(k-1)}, \boldsymbol{y}^{(k)})$ from (4) shortens the expression $p(\boldsymbol{x}^{(k)}|\boldsymbol{x}^{i,(k-1)})$ from the numerator. This approach is useful (regarding the SIR algorithm) especially when the transition model is given by function, from which particles drawing is not very problematic. Due to unconditionally resampling step, one do not have to calculate (5) in every time step.

BF algorithm is often used due to its easy implementation and satisfactory results of estimation.

3) Auxiliary Particle Filter

The Auxiliary Particle Filter (APF) algorithm was proposed by Pitt and Shephart in 1999 [16]. It is the modification of BF algorithm by improving the method of selecting the position of the particles in each simulation step. The algorithm assumes additionally drawing the auxiliary index, which is drawn for every particle and it has an influence on the selection of the particles during prediction and on weights calculation. Again it is assumed, that the resampling is executed in all iterations.

Algorithm 3: Auxiliary Particle Filter

1. Draw N_p particles from initial PDF $x^{i,(0)} \sim p(x^{(0)})$, set initial weights $q^{i,(0)} = N_p^{-1}$ and initial time step $k = 1$.
2. Draw auxiliary values $\boldsymbol{\mu}^{i,(k)} \sim p(\boldsymbol{x}^{(k)}|\boldsymbol{x}^{i,(k-1)})$ and calculate their weights according to

$$q_*^{i,(k)} = p\left(\boldsymbol{y}^{(k)} \mid \boldsymbol{\mu}^{i,(k)}\right). \tag{7}$$

3. Normalization of the weights $q_*^{1...Np,(k)}$.
4. Draw N_p auxiliary indices a^j, PDF for drawing is given by set of pairs: $\{x^{i,(k-1)}, q_*^{i,(k)}\}$ (probability that $a^j = b$ will be drawn is equal to $q^{b,(k)}$).
5. Draw samples from transition model using auxiliary indices: $\boldsymbol{x}^{j,(k)} \sim p(\boldsymbol{x}^{(k)}|\boldsymbol{x}^{aj,(k-1)})$.
6. Calculate weights of the particles using the expression

$$q^{j,(k)} = \frac{p\left(\mathbf{y}^{(k)} \mid \mathbf{x}^{j,(k)}\right)}{p\left(\mathbf{y}^{(k)} \mid \mu^{aj,(k)}\right)}. \tag{8}$$

7. Normalization. Scale the weights in such a way that their sum be equal to 1.
8. Resampling. Described at the end of the Section.
9. End of the iteration. Calculate the estimation of k-th time step, update $k=k+1$, go to step 2.

In the prediction step, one do not consider only the existing particle, but also the latest observation of particle. The APF algorithm introduces an auxiliary variable and it makes the PDF more reasonable and more close to the real situation.

4) Likelihood Particle Filter

The Likelihood Particle Filter (LPF) algorithm was proposed in [14]. It differs from other particle filters algorithms because here the measurement model is used for particles drawing and the transition model – for weights calculation. This algorithm was prepared specially for selected object (*Ob*1 from the next section) with quadratic measurement function and it assumes the auxiliary variable $\boldsymbol{s}^{(k)} = (\boldsymbol{x}^{(k)})^2$ addition. Again it was assumed that the resampling is executed in all iteration steps.

Algorithm 4: Likelihood Particle Filter

1. Draw N_p values of particles from initial PDF $x^{i,(0)} \sim p(x^{(0)})$, set $q^{i,(0)} = N_p^{-1}$, set time step $k = 1$.
2. Draw particles from measurement model $\boldsymbol{s}^{i,(k)} \sim p(\boldsymbol{y}^{(k)}|\boldsymbol{s}^{(k)})$. The drawing for each particle should be repeated until $\boldsymbol{s}^{i,(k)} \geq 0$.
3. Draw for each particle $u \sim U[0,1]$. If $u > 0.5$, then $\boldsymbol{x}^{i,(k)} = \sqrt{\boldsymbol{s}^{i,(k)}}$, otherwise $\boldsymbol{x}^{i,(k)} = -\sqrt{\boldsymbol{s}^{i,(k)}}$.
4. Draw weights of the particles using the transition model:

$$q^{i,(k)} = p\left(\boldsymbol{x}^{i,(k)} \mid \boldsymbol{x}^{i,(k-1)}\right)\boldsymbol{x}^{i,(k)} \tag{9}$$

5. Normalization. Scale the weights in such a way that their sum be equal to 1.
6. Resampling. Described at the end of the Section.
7. End of the iteration. Calculate the k-th time step estimate, update $k=k+1$, go to step 2.

The PDF, which particles are drawing from, is not intended to be generically applicable, because it does not

work so good in objects with not-square measurement model. One can see that the choice of importance density can influence the estimation quality of the algorithm.

5) Resampling

Resampling is the re-drawing of N_p particles, but only from the existing ones. Probability of each particle drawing is equal to its normalized weight. In the presented research systematic resampling was applied.

Algorithm 5: Resampling

1. Set the values $j = 1$, $sumQ = q^{1,(k)}$, $i = 1$.
2. Draw u value from $U[0,1]$.
3. Update $j = j + 1$, $sumQ = sumQ + q^{j,(k)}$, while $sumQ < u$.
4. Every i-th particle after resampling one can described by a set of pairs: $\{x^{j,(k)}, N_p^{-1}\}$.
5. Update $i = i + 1$, $u = u + N_p^{-1}$, go to step 3.

There are many different resampling algorithms, and the description of over twenty of them one can find in [17].

Examined objects

The system *Ob*1 is a plant without the input signal (is autonomous), so the change of the state is only due to the presence of a process noise $\boldsymbol{v}^{(k)}$. It is used very often to particle filters examination [8, 14]. According to publication [18], this object was firstly proposed in 1978 by Netto, Gimeno and Mendes. The initial value is set to $x^{(0)}=0.1$, noises parameters are $v^{(k)} \sim N(0;\ 10)$, $n^{(k)} \sim N(0;\ 1)$. System *Ob*1 is given by equations

$$(Ob1) \quad \begin{cases} x^{(k+1)} = 0.5x^{(k)} + \dfrac{25x^{(k)}}{1+x^{(k)^2}} + 8\cos(1.2k) + v^{(k)} \\ y^{(k)} = \dfrac{x^{(k)^2}}{20} + n^{(k)} \end{cases}$$

The second system *Ob*2 (MIMO 2x2) has been proposed by the authors in order to note the selected properties of estimation and quality indices. The initial value is set to $x^{(0)}=[0.1;\ 0.1]^T$, noises parameters are $v_1^{(k)} \sim N(0;\ 0.1)$, $v_2^{(k)} \sim N(0;\ 100)$, $n_1^{(k)} \sim N(0;\ 0.1)$, $n_2^{(k)} \sim N(0;\ 100)$. There are input signals in this object for which uniform noise from the interval <-1; 1> is used. System *Ob*2 is given by equations

$$(Ob2) \quad \begin{cases} x_1^{(k+1)} = 0.5\left(x_1^{(k)^2}\right)^{\frac{1}{3}} + 0.01x_2^{(k)} + 0.2u_1^{(k)} + v_1^{(k)} \\ x_2^{(k+1)} = 0.5\left(x_2^{(k)^2}\right)^{\frac{1}{3}} + 0.01x_1^{(k)} + 0.2u_2^{(k)} + v_2^{(k)} \\ y_1^{(k)} = 2\left(x_1^{(k)}\right)^2 + n_1^{(k)} \\ y_2^{(k)} = 2\left(x_2^{(k)}\right)^2 + n_2^{(k)} \end{cases}$$

Quality indices

Five different quality indices (given by equations (10-14) with MSE_i and $RMSE_i$ defined by (15) and (16), respectively) were taken into account during the studies:

$$(10) \qquad aRMSE = \frac{1}{N_x}\sum_{i=1}^{N_x} RMSE_i \ ,$$

$$(11) \qquad J_{xi} = \frac{1}{M\sigma_{v_i}^{\ 2}}\sum_{k=1}^{M}\left(\hat{x}_i^{(k)} - x_i^{+(k)}\right)^2, \ \ J_x = \frac{1}{N_x}\sum_{i=1}^{N_x} J_{xi} \ ,$$

$$(12) \qquad J_{yi} = \frac{1}{M\sigma_{n_i}^{\ 2}}\sum_{k=1}^{M}\left(\hat{y}_i^{(k)} - y_i^{+(k)}\right)^2, \ \ J_y = \frac{1}{N_y}\sum_{i=1}^{N_y} J_{yi} \ ,$$

$$(13) \qquad \varepsilon_{y_i} = \frac{\sum_{k=1}^{M}\left|\hat{y}_i^{(k)} - y_i^{+(k)}\right|}{\sum_{k=1}^{M}\left|y_i^{(k)} - y_i^{+(k)}\right|}, \ \ \varepsilon_{y1} = \frac{1}{N_y}\sum_{i=1}^{N_y}\varepsilon_{y_i} \ ,$$

$$(14) \qquad \varepsilon_{y2} = \frac{1}{N_y}\frac{\sum_{k=1}^{M}\sum_{i=1}^{N_y}\left|\hat{y}_i^{(k)} - y_i^{+(k)}\right|}{\sum_{k=1}^{M}\sum_{i=1}^{N_y}\left|y_i^{(k)} - y_i^{+(k)}\right|} \ ,$$

$$(15) \qquad MSE_i = \frac{1}{M}\sum_{k=1}^{M}\left(\hat{x}_i^{(k)} - x_i^{+(k)}\right)^2 \ ,$$

$$(16) \qquad RMSE_i = \sqrt{MSE_i} \ ,$$

where: k – time step; M – number of simulation steps; N_x – number of state variables; N_y – number of state measurements; $y_i^{(k)}$ is a measurement, hat means that these values are estimated, and sign + indicates true values.

The index giving by (10) represents absolute error, indices (11-12) – absolute errors scaled by values of noise variances, whereas (13-14) are the relative errors. First two indices (10-11) show state variables fitness and the last three show measurement compliance. Additionally, J_x and J_y are the residual functions of state variables and measurements, respectively

$$(17) \qquad r_{xi}^{(k)} = \frac{\hat{\mathbf{x}}_i^{(k)} - \mathbf{x}_i^{+(k)}}{\sigma_{vi}}, \ \ r_{yi}^{(k)} = \frac{\hat{\mathbf{y}}_i^{(k)} - \mathbf{y}_i^{+(k)}}{\sigma_{ni}}.$$

One more quality index given by

$$(18) \qquad \beta_i = \sum_{k=1}^{M}\left|\frac{\hat{x}_i^{(k)} - x_i^{+(k)}}{x_i^{+(k)}}\right|, \ \ \beta = \frac{1}{N_x}\sum_{i=1}^{N_x}\beta_i$$

one can find in literature [19] – relative error of state variables. However, in previous research there has been shown that this index is not good for objects which state value is nearly or equal to zero, because the appearance of zero in denominator causes rapid growth of the index value [20].

Proposed indices are from [20-23] or have been suggested by the authors by modifying them.

Simulation results

For each method and object, simulations with $M = 1000$ time steps have been performed. In order to decrement standard deviations of the quality indices, each simulation has been repeated 1000 times (but with new signals every time). All simulations with particle filters were made with 500 particles. Standard deviations of the means were calculated based on the theory from [24], which says that for Gaussian PDF the variance from mean value is m times smaller than variance from m-elemented sample.

Values of quality indices for each method and object have been presented in Fig. 1-2. Standard deviations with 99.7% probability (based on 68-95-99.7 rule from [24]) are presented on the graphs.

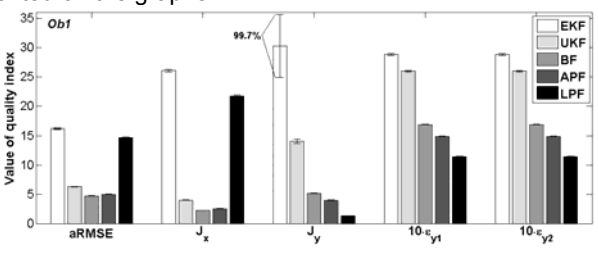

Fig.1. Comparison of quality indices for *Ob*1

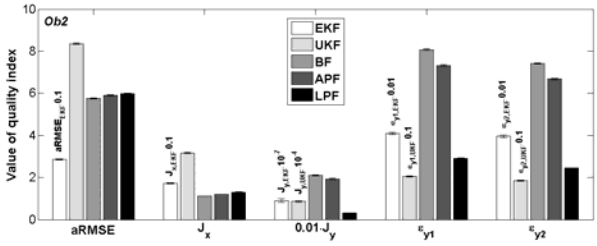

Fig.2. Comparison of quality indices for *Ob*2

Conclusions

Standard deviations on bars in Fig. 1 and Fig. 2 are relatively low, for most bars invisible. Therefore, the evaluation of quality (the order from the best to the worst filter) is justified. In Fig. 2, most of the quality indices for KFs are scaled because these methods had far worse results than the particle filters (by the height of these bars, the remaining charts could became invisible). All average values of quality indices are listed in Table 1

Based on the simulation results, one can easily see that for examined objects particle filters give definitely better results than Kalman filters. The best action, regarding the state variables tracking (according to (10-11) indices), is provided by BF and APF algorithms for both *Ob*1 and *Ob*2. In all cases one can see minimal advantage of Bootstrap algorithm. The worst is the EKF algorithm for both systems and all quality indices. High standard deviation of J_y index is caused by mismatch of the real and estimated state variables, what affects the divergence between real and estimated output values too. The best adjustment of measurement values (indices (12-14)) is provided by LPF algorithm, both BF and APF algorithms turned out to be somewhat worse. The bad results of KF algorithms are caused by high nonlinear transition and measurement models of used plants and also by relatively high number of particles in PF algorithms.

LPF algorithm can be used only for the selected objects type, so in the article only objects with square measurement functions were used. It does not produce the best results in the estimation of state variables ($aRMSE$ and J_x indices), because the square of the state variable value is drawn, and then the sign of that variable is selected randomly. However, in the measurement match (the choice of such state variables to estimate the object's output as close as possible to the real values), this algorithm provides the best estimation (state variables are also best suited here, but only for absolute value). Therefore, in the case where only output signals need to be estimated and this outputs are square functions of state variables, it is best to use the LPF algorithm.

In the case of *Ob*2, one can see a major defect of $aRMSE$. As has already been said, this index represents an absolute error and is not rescaled in any way. Therefore, the $RMSE_2$, derived from the state variable x_2 which has a higher noise variance than x_1, is significantly higher than $RMSE_1$. This results in a much greater impact of the second component on the total $aRMSE$ value. In other indices, this problem is resolved by scaling them by the value of the noise variances (11-12) or by entering a relative error (13-14). In order to presented results for 1000 simulations of Extended Kalman Filter on object *Ob*2, the components of values of selected quality indices were collected. The components of the $aRMSE$ index were: $RMSE$ = [0.4393; 56.8948], while J_x = [1.9354; 32.6845]. It is therefore apparent that in the case of error scaling by the value of noise variance (11) the differences between the components, with large differences in noise variances, are less pronounced.

The characteristic quality index is ε_{y2}. It has no division into components (sums up in one step after all measurements) and its value is similar to value of ε_{y1}, both of the indices show the ratio of estimation errors to the measurements errors. For one-dimensional objects these indices have exactly the same values. For a multidimensional system *Ob*2, index ε_{y2} is lower (from 3.2% to 28.1%) than ε_{y1}. It is caused by the division by one high value, instead of few lower. Large difference between ε_{y1} and ε_{y2} for UKF show that one of the outputs is much worse estimated than the second one. However, both indices generally show the same information, hence in the future, authors will probably limit the number of used quality indices.

Simulations times are shown in the Table 2. It is easy to see that the shortest computation time is provided by the Extended Kalman Filter method, since there are only matrix calculations. A little longer time is needed for the UKF method, due to the additional two stages of the algorithm (unscented transformation and selection of sigma points). The most computationally complex are particle filter algorithms, since particle drawings and weight calculations have to be repeated at each time step N_p times. Reducing N_p may shorten the calculation time, but at the cost of the worse estimation quality. The longest computation time takes APF algorithm, because the indices a^i and the auxiliary variables $\mu^{i,(k)}$ must be additionally calculated N_p times. The LPF algorithm for the multidimensional object is also time-consuming, since at every time step, the auxiliary variables $s^{i,(k)}$ must be drawn to each particle until the values of those variables corresponding to each state variable are greater than zero (in the authors opinion it could be improved and it will be one of the future research directions).

In the future, the authors plan to develop several other modifications of the particle filter, general Likelihood PF among others (to work not only with objects with square measurement functions).

Table 1. The average values of quality indices

Object	Indices	EKF	UKF	BF	APF	LPF
*Ob*1	$aRMSE$	16.1	6.29	4.70	5.00	14.9
	J_x	26.0	3.97	2.22	2.51	21.7
	J_y	36.2	14.1	5.14	4.00	1.28
	ε_{y1}	2.88	2.59	1.68	1.49	1.14
	ε_{y2}	2.88	2.59	1.68	1.49	1.14
*Ob*2	$aRMSE$	28.7	8.35	5.74	5.89	5.98
	J_x	17.3	3.18	1.13	1.20	1.31
	J_y	$8.99 \cdot 10^6$	$8.59 \cdot 10^3$	210	194	30.4
	ε_{y1}	410	26.0	8.08	7.32	2.94
	ε_{y2}	397	18.7	7.40	6.65	2.46

Table 2. The calculation times [s] for all methods and objects (time for 1000 repetitions)

Obiekt	EKF	UKF	BF	APF	LPF
*Ob*1	0.3	48.9	1511.8	3043.7	733.7
*Ob*2	100	222.3	2019.9	6541.2	5497.7

***Authors**: Eng. Jacek Michalski, Poznan University of Technology, Faculty of Electrical Engineering, Institute of Control, Robotics and Information Engineering, Division of Control and Robotics, Piotrowo 3a Street, 60-965 Poznań, E-mail: jacek.michalski95@wp.pl; M.Eng. Piotr Kozierski, Poznan University of Technology, Faculty of Computing, Institute of Automation and Robotics, Division of Electronics Systems and Signal Processing and also Faculty of Electrical Engineering, Institute of Control, Robotics and Information Engineering, Division of Control and Robotics, Piotrowo 3a Street, 60-965 Poznań, E-mail: piotr.kozierski@gmail.com; EngD Joanna Ziętkiewicz, Poznan University of Technology, Faculty of Electrical Engineering, Institute of Control, Robotics and Information Engineering, Division of Control and Robotics, Piotrowo 3a Street, 60-965 Poznań, E-mail: joanna.zietkiewicz@put.poznan.pl*

REFERENCES

[1] Abur A., Exposito A. G., Power System State Estimation: Theory and Implementation, Marcel Dekker, Inc., (2004), 17-49. DOI: 10.1201/9780203913673.ch2

[2] Okon T. Weighted-least-squares Power System State Estimation in Different Coordinate Systems, Przegląd Elektrotechniczny 86 (2010), No. 11, 54-58.

[3] Udupa H. N., Minal M., Mishra M. T., Node Level ANN Technique for Real Time Power System State Estimation, International Journal of Scientific & Engineering Research, 5 (2004), No. 1, 1500-1505.

[4] Zawirski K., Deskur J., Kaczmarek T., Automation of Electric Drive (in Polish), Publishing House of Poznań University of Technology, Poznań 2012.

[5] Hajiyev C., Soken H. E., Robust adaptive Kalman filter for estimation of UAV dynamics in the presence of sensor/actuator faults, Aerospace Science and Technology, 28 (2013), No. 1, 376-383.

[6] Marantos P., Koveos Y., Kyriakopoulos K. J., UAV State Estimation using Adaptive Complementary Filters, IEEE Transactions on Control Systems Technology, 24 (2016), No. 4, 1214-1226.

[7] Schulz D., Burgard W., Fox D., Cremers A. B., Tracking multiple moving targets with a mobile robot using particle filters and statistical data association, Robotics and Automation, (2001). Proceedings 2001 ICRA. IEEE International Conference on. Vol. 2. IEEE, 2001., 1665-1670.

[8] Gordon N. J., Salmond D. J., Smith A. F. M., Novel Approach to Nonlinear/non-Gaussian Bayesian State Estimation, IEE Proceedings-F, 140 (1993), No. 2, 107-113. DOI: 10.1049/ip-f-2.1993.0015

[9] Ke L., Jingjing W., Lei S., Peng Ch., New Particle Filter Based on GA for Equipment Remaining Useful Life Prediction. Sensors 17 (2017), No. 4, 696.

[10] Ribeiro M., Riberio I., Kalman and Extended Kalman Filters: Concept, Derivation and Properties, Institute for Systems and Robotics, 43 (2004)

[11] Wan, Eric A., and Rudolph Van Der Merwe, The unscented Kalman filter for nonlinear estimation, Adaptive Systems for Signal Processing, Communications, and Control Symposium, (2000), AS-SPCC, The IEEE 2000. Ieee, 2000.

[12] Kalman R E., A New Approach to Linear Filtering and Prediction Problems, Journal of basic Engineering, 82 (1960), No. 1, 35-45

[13] Candy J. V., Bayesian Signal Processing, WILEY, New Jersey 2009, 36-44, 237-298. DOI: 10.1002/9780470430583

[14] Arulampalam S., Maskell S., Gordon N., Clapp T., A Tutorial on Particle Filters for On-line Non-linear/Non-Gaussian Bayesian Tracking, IEEE Transactions on Signal Processing, 50 (2002), No. 2, 174-188. DOI: 10.1109/78.978374

[15] Doucet A., Johansen A. M., A Tutorial on Particle Filtering and Smoothing: Fifteen years later, handbook of Nonlinear Filtering 2009/12, 656-704.

[16] Pitt, Michael K., Shephard N., Filtering via simulation: Auxiliary particle filters, Journal of the American statistical association, 94 (1999), No. 446, 590-599.

[17] Kozierski P., Lis M., Ziętkiewicz J., Resampling in Particle Filtering – Comparison, Studia z Automatyki i Informatyki, 38 (2013), 35-64.

[18] Kitagawa G., Monte Carlo Filter and Smoother for Non-Gaussian Nonlinear State Space Models, Journal of computational and graphical statistics, 5 (1996), No. 1, 1-25.

[19] Chen H., Liu X., She C., Yao C., Power System Dynamic State Estimation Based on a New Particle Filter, Procedia Environmental Sciences, 11 (2011), Part B, 655-661. DOI: 10.1016/j.proenv.2011.12.102

[20] Michalski J., Kozierski P., Ziętkiewicz J., Comparison of State Estimation Methods for Dynamical Systems (in Polish), Pomiary Automatyka Robotyka, 21 (2017), No. 4, 41-47.

[21] Valverde G., Terzija V., Unscented Kalman Filter for Power System Dynamic State Estimation, IET Generation, Transmission & Distribution, 5 (2011), Iss. 1, 29-37.

[22] Singh R., Pal B. C., Jabr R. A., Choice of estimator for distribution system state estimation, IET Generation, Transmission & Distribution, 3 (2009), Iss. 7, 666-678.

[23] Kozierski P., Lis M., Horla D., Wrong Transition and Measurement Models in Power System State Estimation, Archives of Electrical Engineering, 65 (2016), No. 3, 559-574. DOI: 10.1515/aee-2016-0040

[24] Florek A., Mazurkiewicz P., Dynamic Signals and Systems (in Polish), 2nd ED., Poznań 2015. ISBN: 978-83-7775360-6