Discrete measurement and transition models in Particle Filter

Piotr Kozierski, *Poznan University of Technology*

Abstract

In article there is proposed a new approach to particle filter modeling – creation discrete transition and measurement models. This new approach assumes knowledge of transition model (2) and measurement model (3), but written by certain values set (non-analytical form). It opens new perspectives for particle filter prosper, in problems which analytical form cannot be used or it is hindered.

1. Introduction

Particle filter (PF), is a state observer similar to Kalman filter, although PF can be used in either cases with non-linear/non-Gaussian processes, in which Kalman filter and its modifications (EKF and UKF) are insufficient [2].

First research regarding PF starts in the mid-twentieth century [5, 6, 12], but their popularity still rises – it is connected with computational power increase over last years. Parallel calculations also contributed to frequent use PF methods, because of their principle of operation [10, 13].

Particle filter principle is described in chapter 2 – there is shown basic variants of PF. In chapter 3 particle filter with discrete transition and measurement models are proposed (DMPF). Simulation results and PF comparison are presented in chapter 4.

List of abbreviations:

BF	Bootstrap filter
DMBF	Discrete models in BF
DMPF	Discrete models in particle filter
DMSIS	Discrete models in SIS
EKF	Extended Kalman filter
ESS	Effective sample size
HMM	Hidden Markov Model
IS	Importance sampling
PDF	Probability density function
PF	Particle filter
RBE	Recursive Bayesian estimation
SIR	Sequential importance resampling
SIS	Sequential importance sampling
SLLN	Strong law of large numbers
UKF	Unscented Kalman filter

2. Particle filter

Consider the state space model

$$\begin{aligned} x_k &= f_k\left(x_{k-1}, v_{k-1}\right) \\ y_k &= g_k\left(x_k, n_k\right) \end{aligned} \tag{1}$$

where v_{k-1} represents process noise and n_k measurement noise [1]. System (1) is hidden Markov model (HMM), therefore state variables are unobserved (filtration is needed) and their value depends only on the previous step state.

PF is one method of implementing a recursive Bayesian estimation (RBE), where are assumed that both hidden state values and measured object output are random variables [3]. Hence model (1) one can represent in the stochastic form [11]

$$x_k \sim p\left(x_k \middle| x_{k-1}\right) \tag{2}$$

$$y_k \sim p\left(y_k \middle| x_k\right) \tag{3}$$

where (2) is a transition model whereas (3) is a measurement model.

RBE goal is to find the posterior PDF

$$p\left(x_k \middle| Y_k\right) = \frac{p\left(y_k \middle| x_k\right) p\left(x_k \middle| Y_{k-1}\right)}{p\left(y_k \middle| Y_{k-1}\right)} \tag{4}$$

where $p\left(y_k \middle| x_k\right)$ is the likelihood, $p\left(x_k \middle| Y_{k-1}\right)$ is the prior PDF and $p\left(y_k \middle| Y_{k-1}\right)$ is the evidence or normalizing factor. The notation assumes

$$Y_k = \left\{y_1, y_2, ..., y_k\right\} \tag{5}$$

Equation (4) can be also written as

$$p\left(x_k \middle| Y_k\right) = \frac{p\left(y_k \middle| x_k\right) p\left(x_k \middle| x_{k-1}\right) p\left(x_{k-1} \middle| Y_{k-1}\right)}{p\left(y_k \middle| Y_{k-1}\right)} \tag{6}$$

where $p\left(x_{k-1} \middle| Y_{k-1}\right)$ is the posterior from the previous step. Thus one can see that it is recursive form. $p\left(y_k \middle| Y_{k-1}\right)$ can be describe as

$$p(y_k|Y_{k-1}) = \int p(y_k|x_k)p(x_k|Y_{k-1})dx_k \quad (7)$$

It follows that denominator in (6) is used only to normalization posterior and this equation can be written as

$$p(x_k|Y_k) \propto p(y_k|x_k)p(x_k|x_{k-1})p(x_{k-1}|Y_{k-1}) \quad (8)$$

where $\propto$ means „directly proportional".

To be able to use the RBE, both models (2) and (3) must be known. There is also required knowledge of the initial PDF $p(x_1)$. Knowledge about the model (2) is equivalent to knowledge about distribution $v_k \sim V$ (because model (1) is known).

Particle filter implementation assumes representation of posterior PDF by a set of particles $\{x_k, \quad k = 1, \ldots, N\}$ with associated weight w_k [1]

$$p(x_k|Y_k) \approx \sum_{i=1}^{N} w_k^i \delta(x_k - x_k^i) \quad (9)$$

where N means number of particles, and $\delta(\cdot)$ is the Dirac delta. Using strong law of large numbers (SLLN) one can write

$$p(x_k|Y_k) \overset{N\to\infty}{=} \sum_{i=1}^{N} w_k^i \delta(x_k - x_k^i) \quad (10)$$

therefore with increase number of particles N, posterior accuracy improved.

But there are also more sophisticated PDF notation methods, for example

$$p(x) = \sum_{i=1}^{N} \beta_i p_i(x) \quad (11)$$

where each $p_i(x)$ is a PDF and the sum of all β_i is equal to 1 [4].

Finally to calculate state variable estimate is need to compute the expected value of the discrete distribution

$$\hat{x}_k = E[x_k|y_k] = \sum_{i=1}^{N} x_k^i w_k^i \quad (12)$$

2.1 Sequential Importance Sampling (SIS)

In (9) weights are selected to Importance Sampling (IS) principle [7]. In general draw from distribution $p(x_k|Y_k)$ can be difficult, so the particles are drawn from proposal distribution $q(\cdot)$ which can easily draw – it is so-called importance density. With this approach particle weights drawn from the importance density are defined by

$$w_k^i \propto \frac{p(x_k^i|Y_k)}{q(x_k^i|y_k)} \quad (13)$$

Substituting $q(x_k^i|y_k) = q(x_k^i|x_{k-1}^i, y_k)q(x_{k-1}^i|y_{k-1})$ and (8) to (13) received

$$w_k^i \propto \frac{p(y_k|x_k^i)p(x_k^i|x_{k-1}^i)}{q(x_k^i|x_{k-1}^i, y_k)} \cdot w_{k-1}^i \quad (14)$$

In Algorithm 1 SIS principle of operation is written.

Algorithm 1

1. Draw particles from initial distribution $x_0^i \sim p(x_0);\ k = 1$.
2. Draw particles from importance density $x_k^i \sim q(x_k|x_{k-1}^i, y_k)$.
3. Compute particle weights w_k^i according to (14)
4. Weights normalization

$$w_k^i = \frac{w_k^i}{\sum_{j=1}^{N} w_k^j} \quad (15)$$

5. Step update $k = k+1$; return to step 2.

Importance density $q(\cdot)$ can be any function (for example in [9] Gaussian Particle Filter is presented, importance density is normal distribution where expected value and variance are dependent on state variable), moreover it can depend on one of these two arguments (for example in the Bootstrap Filter importance density is depended on state value from the previous step only), or be completely independent of state values and system output (but this choice may affect the PF principle).

The biggest drawback of the SIS method is degeneration problem – after few (dozen) of steps all particles except one have weights close to or equal to zero. The easiest way to prevent the degeneration problem is resampling.

2.2 Sequential Importance Resampling (SIR)

SIR SIS algorithms are different from those that have resampling, therefore SIR principle of operation is as follows:

Algorithm 2

1. Draw particles from initial distribution $x_0^i \sim p(x_0);\ k=1$.
2. Draw particles from importance density $x_k^i \sim q(x_k|x_{k-1}^i, y_k)$.
3. Compute particle weights w_k^i according to (14).
4. Weights normalization according to (15).
5. Check the condition for resampling.
6. If the condition is met, do resampling.
7. Step update $k=k+1$; return to step 2.

SIR versions will differ in importance densities only, but also in resampling conditions and in resampling at all.

In general is checked so called effective sample size (ESS), which can be estimated by

$$\hat{N}_{ESS} = \frac{1}{\sum_{i=1}^{N}\left(w_k^i\right)^2} \tag{16}$$

If ESS is smaller then the set threshold N_T, then resampling is needed. Typically threshold is set on $N_T = N/2$ [2, 7].

The most commonly used resampling method is drawing N new samples from the current approximation (9) – probability of selecting a value x_k^i is equal to w_k^i. After resampling all weights are equal to $w_k^i = 1/N$.

With the resampling particles with small weights are removed, and particles with greater weights are copied. One can say that resampling is a probabilistic implementation of the Darwinian idea of *survival of the fittest* [14].

2.3 Bootstrap Filter (BF)

Algorithm proposed in 1993 by Gordon, Salmond and Smith [8] is considered to be classic SIR algorithms, and PF at all. It assumed that resampling is in every step, and the transition model is importance density

$$q(x_k|x_{k-1}^i, y_k) = p(x_k|x_{k-1}^i) \tag{17}$$

Hence (14) one can write as

$$w_k^i \propto w_{k-1}^i \cdot p(y_k|x_k^i) \tag{18}$$

Resampling is in every step, so all weights are equal $w_{k-1}^i = 1/N$, and finally

$$w_k^i \propto p(y_k|x_k^i) \tag{19}$$

BF principle of operation present Algorithm 3.

Algorithm 3

1. Draw particles from initial distribution $x_0^i \sim p(x_0);\ k=1$.
2. Draw particles from transition model $x_k^i \sim p(x_k|x_{k-1}^i)$.
3. Compute weights $w_k^i = p(y_k|x_k^i)$.
4. Weights normalization according to (15).
5. Resampling.
6. Step update $k=k+1$; return to step 2.

Transition model choice as importance density results that particles are drawing without taking into account the system output – algorithm is not resisting for outliers values [2], but its benefits (ease of implementation, no requirement to save weights values from previous step, relatively high speed) make this algorithm is willingly used.

3. Transition model and measurement model

Knowledge of both models is one of the PF assumptions and usually follows directly from the model (1). For example one can consider state-space model

$$\begin{aligned} x_k &= 0.9x_{k-1} + v_{k-1} \\ y_k &= \frac{1}{x_k^2+1} + n_k \end{aligned} \tag{20}$$

Transition model needed to computing in step 3 of Algorithm 1 one can receive by appointment noise value from system model

$$v_{k-1} = x_k - 0.9x_{k-1} \tag{21}$$

and then it need to find PDF value this random variable

$$S_v(v_{k-1}) = S_v(x_k - 0.9x_{k-1}) \tag{22}$$

Just proceed to the measurement model to find $p(y_k|x_k^i)$ value – then is obtained

$$S_n(n_k) = S_n\left(y_k - \frac{1}{x_k^2+1}\right) \tag{23}$$

To draw particle in 2nd step of Algorithm 3 one only need to draw system noise sample v_{k-1} from distribution S_v and to substitute into the first equation in (20).

It should be noted that in each of these cases, knowledge about PDF S_v and S_n is required. The problem also exist when the object is a white box (model structure is known, but the unknown is each

constant), or when knowledge is even less about the model (for example, limited to the number of state variables). There are also much more complex systems than (20), in which for states affect more different random variables. There may not be able to provide noise in the equation form. Another problem is need to modificate the code at every change the system – in this case, better to write the universal program.

In each of these cases one can use proposed in this article transition and measurement models write form – it involves saving the both models in the form of discrete PDF.

An example is illustrated in Fig.1 – first there is drawing some samples. Then histogram is created, which becomes PDF after normalization.

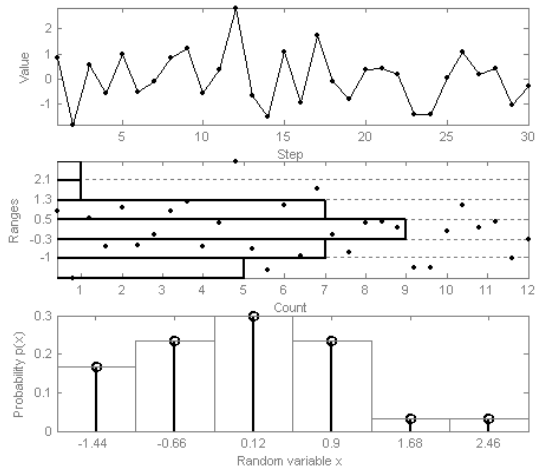

Fig.1. Way to create discrete PDF.

Thus obtained transition model and measurement model written as well as posterior – by a set of particles with associated weights. But these models are fixed and the distance between all particles is constant.

Unfortunately, to create both model it must be possible to measure not only the system output, but also the state variables. Only after obtaining information needed to create transition and measurement models, object can be considered as HMM.

It should also be noted that the greater simulation experiment steps number M_{sim}, the more accurate discrete models are created. Also increase in the number of intervals (particle counts) L_f has a positive effect on the accuracy of representation of these two models (but M_{sim} has to be big enough).

Therefore by the SLLN, using discrete transition and measurement models, assuming $M_{sim} \to \infty$ and $L_f \to \infty$ one can write

$$p(x_k|x_{k-1}) = \\ = \sum_{i=1}^{L_f}\sum_{j=1}^{L_f} \widehat{w}^{ij} \cdot \delta(x_k - \widehat{x}_0^i) \cdot \delta(x_{k-1} - \widehat{x}_{-1}^j) \quad (24)$$

$$p(y_k|x_k) = \\ = \sum_{i=1}^{L_f}\sum_{j=1}^{L_f} \breve{w}^{ij} \cdot \delta(y_k - \breve{y}_0^i) \cdot \delta(x_k - \breve{x}_0^j) \quad (25)$$

It follows that there are a particles number N, beyond which further increase has no effect on improvement of PF, therefore (10) ceases to apply for DMPF.

4. Simulation results

To simulation uses system

$$x_k = 0.8 \cdot x_{k-1} + \frac{\exp(0.1 \cdot x_{k-1})}{0.1 + x_{k-1}^2} \cdot v_{k-1} \quad (26)$$
$$y_k = x_k + n_k$$

In all cases there have been used the same sequence of state variable and system output (simulation length is $M = 10000$), so it possible to compare results of different methods.

For $DMBF_m$ it is assumed that always $L_f = 20$. Besides there is subscript next to the method name – it informs about the length of the simulation experiment

$$M_{sim} = 10^m \quad (27)$$

4.1 SIS degeneration

Fig.2 presents the problem of SIS degeneration. One can see that after a few steps number of significant particles is reduced to one.

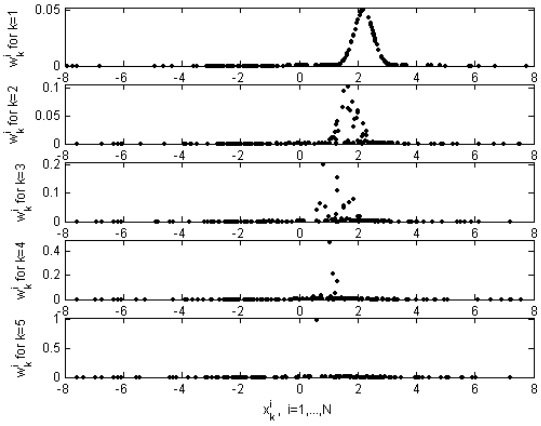

Fig.2. Particle weights (posterior PDF) in SIS method.

By increasing the number of particles N one can delay degeneration, but it is inevitable.

4.2 BF and DMBF comparison

Fig.3 presents fragments of plots with true value of state variable and two estimates – from algorithm BF and $DMBF_4$.

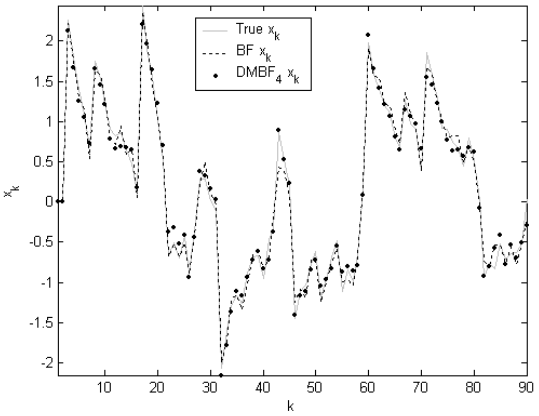

Fig.3. Estimates of state variable for methods BF and $DMBF_4$.

One can notice that in both cases state variable is well tracked. However, it should be noted that in BF the perfect knowledge about distributions S_v and S_n is considered. In the next plot – Fig.4 – presents estimates of state variable for $DMBF_7$ and case of BF in which both noise distributions are poorly recognized – the average value is shifted by the value of variance.

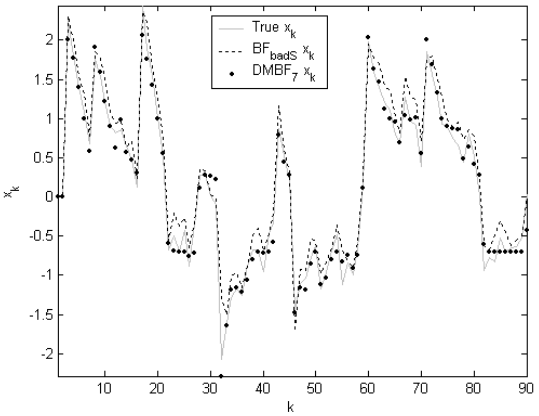

Fig.4. Estimates of state variable for methods BF_{bads} and $DMBF_7$.

Compared Fig.3 and Fig.4 one can notice that valid knowledge about S_v and S_v is required for proper functioning of BF. On the other hand, looking at the state values for $DMBF_7$, one can get the impression that it is the worst algorithm.

There are results for all simulations in Tab.1 – mean square error

$$MSE = \frac{1}{M} \cdot \sum_{k=1}^{M} \left(x_k - \hat{x}_k \right)^2 \qquad (28)$$

and computation time t.

Tab.1.

Compare the effects of different PF algorithms

	N=50	N=100	N=150	N=200
$DMBF_4$	MSE=0.114 t=1.80s	MSE=0.102 t=3.26s	MSE=0.094 t=5.40s	MSE=0.095 t=7.84s
$DMBF_5$	MSE=0.076 t=1.73s	MSE=0.056 t=3.16s	MSE=0.055 t=5.22s	MSE=0.037 t=7.84s
$DMBF_6$	MSE=0.063 t=1.55s	MSE=0.037 t=3.37s	MSE=0.044 t=5.28s	MSE=0.039 t=7.82s
$DMBF_7$	MSE=0.067 t=1.77s	MSE=0.047 t=3.17s	MSE=0.037 t=5.19s	MSE=0.033 t=7.75s
BF	MSE=0.401 t=1.39s	MSE=0.051 t=2.64s	MSE=0.025 t=4.16s	MSE=0.014 t=6.34s
BF_{badS}	MSE=1.543 t=1.62s	MSE=0.398 t=2.61s	MSE=0.124 t=4.34s	MSE=0.050 t=6.43s

This means that the simulation of the presented graphically (all were made for $N = 200$), the $DMBF_7$ given quite good result.

The best results were achieved by the standard method BF. DMBF is also worse in terms of computation time, but it is a quite small difference. It should be noted as a huge impact on the BF effects has the number of particles – for $N = 50$ algorithms with discrete transition and measure models are much better than the standard version.

It can be concluded that DMBF algorithm will be useful where, because of the computation time, very small number of particles is required.

4.3 Influence of particle number N

Previously, it was found that the increase in particles number N does not always improve the x_k tracking. Fig.5 shows the change in mean square error, with values of particles number for BF and $DMBF_7$

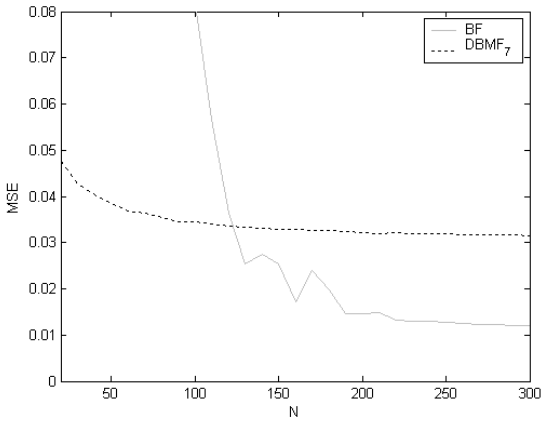

Fig.5. Dependence of particles number N for MSE average value.

The graph confirms earlier observations – BF algorithm is very dependent on the number of particles. It has greater MSE value than $DMBF_7$ for small N, and smaller MSE value than $DMBF_7$ for large N.

From the graph can be read number of particles – about $N = 125$ – for which both methods give the same results.

5. Summary

The paper presents new approach to particle filters – discretization of transition and measurement models. Given a number of cases in which their use is appropriate. Based on the simulations it was found that for a small number of particles DMBF method competes with standard versions of algorithms.

Further research can help to improve the algorithm and achieve better results while maintaining the requirements of a small number of particles.

Bibliography

[1] Arulampalam S., Maskell S., Gordon N., Clapp T.: *A Tutorial on Particle Filters for On-line Non-linear/Non-Gaussian Bayesian Tracking*, IEEE Proceedings on Signal Processing, Vol.50, No.2, 2002, pp.174-188.

[2] Brzozowska-Rup K., Dawidowicz A.L.: *Metoda filtru cząsteczkowego*, Matematyka Stosowana: matematyka dla społeczeństwa 2009, T. 10/51, pp.69-107.

[3] Candy J.V.: *Bayesian signal processing*, WILEY, New Jersey 2009, pp.19-44, 237-298.

[4] Carpenter J., Clifford P., Fearnhead P.: *Improved particle filter for nonlinear problems*, IEE Proceedings – Radar, Sonar and Navigation, Vol.146, No.1, 2/1999, pp.2-7.

[5] Doucet A., Freitas N., Gordon N.: *Sequential Monte Carlo Methods in Practice*, Springer-Verlag, New York 2001, pp.225-246.

[6] Doucet A., Godsill S., Andrieu C.: *On sequential Monte Carlo sampling methods for Bayesian filtering*, Statistics and Computing, 10, 2000, pp.197-208.

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

[8] Gordon N.J., Salmond N.J., Smith A.F.M.: *Novel approach to nonlinear/non-Gaussian Bayesian state estimation*, IEE Proceedings-F, Vol.140, No.2, 1993, pp.107-113.

[9] Kotecha J.H., Djurić P.M.: *Gaussian Particle Filtering*, IEEE Trans Signal Processing, Vol.51, No.10, 10/2003, pp.2592-2601.

[10] Mountney J., Obeid I., Silage D.: *Modular Particle Filtering FPGA Hardware Architecture for Brain Machine Interfaces*, Conf Proc IEEE Eng Med Biol Soc. 2011, pp.4617-4620.

[11] Schön T.B., Wills A., Ninness B.: *System identification of nonlinear state-space models*, Automatica 47 (2011), pp.39-49.

[12] Simon D.: *Optimal State Estimation*, WILEY--INTERSCIENCE, New Jersey 2006, pp.461-484.

[13] Sutharsan S., Kirubarajan T., Lang T., McDonald M.: *An Optimization-Based Parallel Particle Filter for Multitarget Tracking*, IEEE Transactions on Aerospace and Electronic Systems, Vol.48, No.2, 4/2012, pp.1601-1618.

[14] Thrun S., Burgard W., Fox D.: *Probabilistic robotics*, MIT Press, Cambridge, MA, 2005, pp.67-90.

Author:

MSc. Piotr Kozierski
Poznan University of Technology,
Electrical Department,
Room 109
ul. Piotrowo 3A
60-965 Poznań

email:
piotr.kozierski@gmail.com