Scientific Journals
Maritime University of Szczecin

Zeszyty Naukowe
Akademia Morska w Szczecinie

2012, 32(104) z. 2 pp. 81–87

2012, 32(104) z. 2 s. 81–87

The analysis of methods of interaction between elementary filters in multiple model tracking filter in marine radars

Witold Kazimierski

Maritime University of Szczecin, Faculty of Navigation, Chair of Geoinformatics
70-500 Szczecin, ul. Wały Chrobrego 1–2, e-mail: w.kazimierski@am.szczecin.pl

Key words: radar target tracking, multiple model filters, manoeuvre detection

Abstract

Radar target tracking on a sea-going ship is a basic source of information about movement of other vessels, influencing directly safety of navigation. The research on improving of tracking methods has led to a new concept of multiple model neural filtration, which is a combination of multiple model approach with the use of artificial neural networks for the needs of estimation of movement vector. One of the key issue during designing of filter id to establish rules of interaction between elementary filters. The paper presents the most popular methods of manoeuvre detection and interaction of elementary filters in the numerical filtration. The modifications of them for the needs of neural tracking are proposed. Additionally, a concept of use of probabilistic neural network for this purpose is described. The idea was checked in the experimental research with the use of simulation. The result of the research confirmed usefulness of using PNN in multiple model filtration, showing however simultaneously the directions of future research in this. The research was financed by Polish National Centre of Science under the research project "Development of radar target tracking methods of floating targets with the use of multiple model neural filtering".

Introduction

Radar target tracking is one of the basic functions used by the officers in charge of navigational watch for determining of movement parameters of other ships, as well as for making the decisions about anti-collision manoeuvres. Thus, accuracy and topicality of estimated movement vector has a direct impact on the safety of navigation and water transport. The specifics of movement of floating targets cause significant decrease of accuracy of target tracking during non-linear movement of target, e.g. during manoeuvres. In the theory and practice, different methods of implementing non-linearity to Kalman filtration (usually used for radar target tracking) have been tried. One of the possible approaches is to use multiple model filtration. In the research project carried out in Maritime University of Szczecin a new multiple model neural filter has been developed. The structure of elementary filters has been established as a result of earlier stages of the research with the use of simulation and real data. The next stage is to determine the method of interaction between these elementary filters. The paper presents a concept for implementation in the filter together with the research on its reliability. The research have been carried out with the use of Monte Carlo simulation and the implementation platform was a self-made PC application of radar target simulator. For testing probabilistic neural networks, Statistica Neural Network application was used prior to implementation in the simulator.

The paper consists of three major parts. First the concept of multiple model filtration and possible approaches are presented. Then, the problem of manoeuvre detection and mode matching is shown and finally the results of a numerical experiment are presented. It's goal was to test the reliability of proposed solution.

Multiple model filters for target tracking

Basic concepts of multiple model filtration has been presented already in [1] and since then it has been developed by various authors and numerous applications and implementations has occurred. A fine discussion on it can be found for example in

[2] or [3]. The idea of such a filter is that a few elementary filters are used simultaneously. Each of them is adjusted to different dynamics of targets movement and each of them estimates its own elementary movement vector. Final vector can be achieved be selecting one of the elementary vectors (decision-based filters) or by mixing them by calculating weighted average of elementary vectors. The elementary filters can be seen as a representatives of different behaviors of the ship. In [4] five classes of movement dynamics has been proposed, based on simulation analysis of rate of turn, which has been next adjusted based on real data research [5]:

- stable movement or very slow manoeuvre – up to 15°/min;
- slow manoeuvre – 25–30°/min;
- manoeuvre – 30–40°/min;
- fast manoeuvre – 40–55°/min;
- very quick turn – more than 55°/min.

Each mode has its own elementary filter and the decision has to be made which of them shall be used at this particular moment. This is so called decision-based approach. More sophisticated algorithms however assume that instead of hard decision, soft decision shall be made and the final estimated shall be a weighted average of elementary filters outputs.

Numerical filters

There are several numerical multiple model filters. All of them make use of a Bayesian framework – starting with prior probabilities that the system is in a particular mode, the corresponding posterior probabilities are obtained [2]. The basis for these concepts is the Kalman filter. Two basic approaches can be observed. First – static multiple model estimator, in which no switching models is allowed and the dynamic multiple model estimator in which mode jumping is considered. In first case one model is assumed to be in effect throughout the entire process. The prior probability that M_j is correct (the system is in mode j) can be calculated from [2]:

$$P\left\{M_j \middle| Z^0\right\} = \mu_j(0), \quad j = 1,...,r \tag{1}$$

where Z^0 is the prior information and $\sum_{j=1}^{r} \mu_j(0) = 1$, since the correct model is among assumed r possible models. The model is selected based on the likelihood function of mode j at time k [2]:

$$\begin{aligned} \Lambda_j(k) &= p\left[z(k) \middle| Z^{k-1}, M_j\right] = p\left[v_j(k)\right] = \\ &= N\left[v_j(k); 0; S_j(k)\right] \end{aligned} \tag{2}$$

where v_j and S_j are the innovation vector and its covariance corresponding to mode j.

In case of dynamic multiple model filters the process of mode switching is considered and it is assumed to be a Markov process. The mode transition probabilities are known a priori. Based on them and on the innovation process the probability of model switching sequence is calculated. In the next steps it is used in reinitialization of the filter.

There are two most popular dynamic multiple model filters – GPB (generalized pseudo-Bayesian) and IMM (interacting multiple model). The major difference between then is the reinitialization process. GPB uses previous overall estimate as an input for each filter and in IMM each filter uses its own estimate, which is calculated as a weighted sum of estimates of all elementary filters with the weightings of suitable mode jumping. The mode in the estimation steps are selected based on likelihood function with the assumptions of mode jumping. The detailed algorithms can be found in [2, 3, 6].

Neural filter

The concept of using neural network for target tracking arose with the growth of popularity of artificial intelligence in 80's of XX century. Various network structures can be used, however the research carried out in maritime University of Szczecin focused on General Regression Neural Network (GRNN), due to very promising results. The concept of such a filter was shown in [7, 8, 9]. It consists of two parallel GRNNs working together for better smoothing of movement vector. Observed (measured) values of movement vectors are used as input and teaching values while estimated movement vector is the output.

From the mathematical point of view GRNN is basically neural implementation of kernel regression algorithms from the sixties, resulting in computing weighted average of teaching vectors. The weights are the values of Gaussian kernel function for the distances of input vector to teaching vector.

Multiple model approach can also be used for GRNN filter. The research has shown, that different dynamics of targets require different network parameters. There are two basic parameters of the network to be adjusted: the length of teaching sequence and the smoothing factor. The first one says how many previous vectors shall be sued for estimation. Unlike Kalman filter, where estimation (prediction) is based only on the last previous vector, in GRNN many past observations can be included. This allows better smoothing of signal and the vector on radar screen is more stable. The teaching sequence shall be shortened if the target is manoeuvring – the errors of detection are smaller if the measurements for the stable movement are not

included. The other parameter – smoothing factor – says directly how wide the Gaussian kernel function shall be. The larger it is, the more important are the measurements far from the observed vector. It should be small for the dynamic manoeuvres, so that only nearest vectors are included in estimation-then an accuracy of estimated vector is increasing, however it is less smoothed.

Multiple model neural filter consists of at least two GRNN elementary filters as proposed in [4]. Each of them is adjusted for specific movement dynamics model by selecting proper parameters. These can be established in the research based on Monte Carlo simulation. The concept of integrating elementary filters into one complex structure can assume manoeuvre detection and choosing one of the models or interaction between model by calculating weighted average of elementary filters outputs. Both of them will be presented in next chapters.

Manoeuvre detection algorithms for neural filter

The philosophy of detection-based methods requires introduction, so called switching module, into the structure of filter. In case of GRNN filter it is a logical algorithm with the task of selecting the network to take the output signal from and transmitting it to the filter output. For this purpose it is necessary to determine whether the tracked target is in the phase of manoeuvre or of uniform motion by detecting the manoeuvre start or completion. Another essential task of the module is such switching from one network to the other that the vector stability should not be lost too much. The differences between the outputs of both networks may be large enough to cause a sudden vector jump on the radar screen by direct switching over. This function may be joined with detection depending on the detection method applied. The detection-based method in its classical concept shall be used for only two-model filter. If this is the case, two alternative hypotheses are formulated for the needs of manoeuvre detection:

- H_0: the target is not manoeuvring;
- H_1: the target is manoeuvring.

The validity of hypotheses is verified at each estimation step, after the vector's initial stabilization and based on it, suitable model (stable or manoeuvre) is selected.

Manoeuvre detection methods for numerical filters are widely described for example in [10]. They are mostly based on a statistical analysis of the innovation process and its co-variance. Innovation process seems to be the most appropriate for this reason and it is computed in numerical filter algorithm anyhow. The other way sometimes used for manoeuvre detection is to analyze unknown input in so called input detection and estimation approach. Four basic classes of manoeuvre detectors can be indicated [10]:

- chi-squared methods (χ^2);
- Generalized Likeliehood Ratio test;
- modified GLR methods – MLR, CUSUM, SPRT;
- Gaussian significance test.

In the case of a GRNN filter, unlike in numerical methods, the statistical analysis is out of necessity based solely on the observation of estimation's final effects. Methods of manoeuvre detection in a GRNN filter can be divided into two groups:

- based on the statistical analysis of elements estimated by means of only one filter (stable or manoeuvres);
- based on comparing signals coming from both filters (stable and manoeuvre).

Based on analysis of manoeuvre detection methods for numerical filter, three of them have been proposed for GRNN in the research:

- fixed threshold method;
- chi-squared method (χ^2);
- fading average memory method.

The most intuitive and simple m is the fixed threshold method. The validity of hypotheses is verified based on the increments of estimated course (*KRe*) and speeds (*Ve*) at each estimation step (*k)*. If any increase is larger than the established threshold (μ), the detector states in accordance with equations (1), that hypothesis H_1 is true, and so the target is in the state of manoeuvring.

$$\begin{aligned} &\left| KRe\left(k\right) - KRe\left(k-1\right)\right| > \mu_{KR} \\ &\left| Ve\left(k\right) - Ve\left(k-1\right)\right| > \mu_{V} \end{aligned} \tag{3}$$

The proper selection of thresholds μ_{KR} and μ_V is the key problem for ensuring an effective functioning of the method; it takes place in the way of empirical research, for example by means of the Monte Carlo method. The threshold values are a compromise between fast manoeuvres detection and the frequency of false detections.

Chi-square based methods are probably the most popular in numerical filters. For the needs of GRNN they can be also used, however a small modification has to be made, as the innovation vector and its covariance matrix are not directly calculated in GRNN algorithm. Variable q may be defined as in (4):

$$q = (x - \bar{x})^T P (x - \bar{x}) \tag{4}$$

where:

$x = [Vx, Vy]^T$ – estimated movement vector;

P – covariance matrix of x:

$$P = \begin{bmatrix} \sigma_{Vx}^2 & \sigma_{Vx,Vy}^2 \\ \sigma_{Vy,Vx}^2 & \sigma_{Vy}^2 \end{bmatrix} \tag{5}$$

σ_{Vx}^2, σ_{Vy}^2 – variance of Vx, Vy;

$\sigma_{Vx,Vy}^2$, $\sigma_{Vy,Vx}^2$ – covariance (Vx, Vy) and covariance (Vy, Vx), valued 0, as Vx and Vy are independent.

The statistical test is formulated as in (6):

$$q \sim \chi_2^2 > \mu \tag{6}$$

Threshold μ is established based on χ_2^2 test for the assumed probability of false alarm ($P_{FA} = 1 - \alpha$).

Fading memory average method is a kind of mutation of χ^2 method. The basic idea is the same, but the statistics used in the method is different (7).

$$q_k^\alpha = \alpha q_{k-1}^\alpha + q_k \tag{7}$$

The variable q is χ^2-distribute. The number of degrees of freedom is established based on (8):

$$n_\alpha = 2\frac{1+\alpha}{1-\alpha} \tag{8}$$

The research has show that χ^2 and fixe threshold methods shall be used for further implementation of the algorithms.

Model matching concepts for neural filter

If more than one model is to be used in the filter manoeuvre detector is not sufficient. Suitable model shall be selected. This can be done similar to the numerical method, namely based on likelihood function (2). Once again the function makes use of innovation process and its covariance, which are computed directly in numerical methods based on Kalman filter, but not in neural filter. It can be however calculated according to following equations.

First, the innovation vector can be defined by (9) – (11):

$$v(k) = z(k) - x(k) \tag{9}$$

state vector:

$$x(k) = \begin{bmatrix} Vxe \\ Vye \end{bmatrix} \tag{10}$$

Vxe, Vye – estimated values of Vx and Vy;

measurement vector:

$$\begin{aligned} z(k) &= \begin{bmatrix} Vxo \\ Vyo \end{bmatrix} = \begin{bmatrix} (xo(k) - xe(k-1))/T \\ (yo(k) - ye(k-1))/T \end{bmatrix} = \\ &= \begin{bmatrix} (D(k)\cos(BE(k)) - xe(k-1))/T \\ (D(k)\sin(BE(k)) - ye(k-1))/T \end{bmatrix} \end{aligned} \tag{11}$$

where:

Vxo, Vxo – observed values of Vx and Vy;

$D(k)$ – distance measurement at time k;

$BE(k)$ – bearing measurement at time k.

Covariance matrix S of innovation can be calculated from (12) – (15):

$$\begin{aligned} S(k) &= \mathrm{cov}[v(k)] = Q(k) + P(k) = \\ &= \mathrm{cov}[z(k)] + \mathrm{cov}[x(k)] \end{aligned} \tag{12}$$

$P(k)$ can be calculateed as above in (5), while $R(k)$:

$$Q(k) = \mathrm{cov}[z(k)] = R(k) + P(k-1) \tag{13}$$

$$R(k) = \begin{bmatrix} \sigma_x^2 / T^2 & k_{xy} \\ k_{xy} & \sigma_y^2 / T^2 \end{bmatrix} \tag{14}$$

And the values of σ_x^2 and σ_y^2 are calculated as (15) and (16):

$$\sigma_x^2 = (\sigma_D \cos(BE))^2 + (\sigma_{BE} D \sin(BE))^2 \tag{15}$$

$$\sigma_x^2 = (\sigma_D \sin(BE))^2 + (\sigma_{BE} D \cos(BE))^2 \tag{16}$$

where σ_D and σ_{BE} are the errors of distance and bearing measurements.

Another approaches to calculating interaction wages between elementary filters in neural multiple model are also possible. One of them is to use probabilistic neural network. It allows to avoid calculating of innovation vector and its covariance, while computing the probability of mode matching.

Probabilistic Neural Network

Probabilistic Neural Network (PNN) seems to be an interesting algorithm to use in case of GRNN neural multiple model filter. They belong to the same family of networks using the same concept. Their implementation into the system should be thus quite easy as they are using the same classes and methods. GRNN networks are quite often called probabilistic networks for repression estimation.

Probabilistic Neural Networks has been proposed in 1990 by D.F. Specht in [11] as an implementation of Bayes strategy for pattern classification based on estimating of probability density function [11]. A PNN uses a statistical algorithm called kernel discriminant analysis in which the

operations are organized into a multilayered feed forward network. The pdf is calculated according to so called Parzen estimator:

$$f_n(x) = \frac{1}{n\sigma}\sum_{k=1}^{n} W\left(\frac{x - x_k}{\sigma}\right) \qquad (17)$$

where:

- W – weighting function – commonly Gaussian function is used;
- n – number of samples (length of teaching sequence).

The network consists of four layers (input, pattern, summation and output) and it employs radial network in the pattern layer. The structure of a network is strictly determined with the problem to solve and the teaching data. The number of pattern neurons usually is equal to the number of teaching cases (they can be however grouped). The number of neurons in the outer layer is determined with the number of classes to be recognized. A classical structure of network is presented in the figure 1.

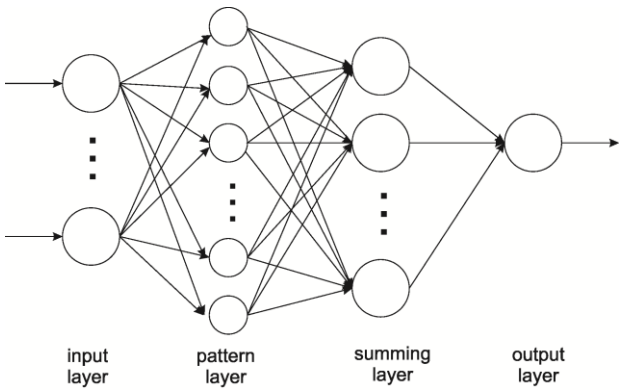

Fig. 1. Structure of Probabilistic Neural Network

Input layer performs the pre-processing of the input values. In the pattern layer the Gaussian function values are calculated based on Euclidean distance of input value and the pattern value. Then the Gaussian values are summed on classes in summing layer. Thus, the weights for each class are calculated, showing the probability of input vector belonging to each class. In the output layer the class with the biggest probability is selected.

Implementation of PNN for multiple model filter requires definition of test value used for classification. This might be innovation vector like in previous mode matching concepts, but this also might be variance of measurements or state vector. This might be also rate of turn and speed rate of target, like in the numerical example below. Then, a required number of sample shall be collected for each – usually during simulation tests. Teaching of the network means in this case copying of all teaching values into the network and proper selection of weights between pattern layer and summing layer (based on the model of teaching pattern). The output layer in fact is not needed in the implementation for multiple model filter. The values computed in the summing layer are the weights for each model in multiple model filter.

Numerical experiment

The numerical experiment presented in the paper aimed at checking the possibility of using PNN in multiple model neural filter for interaction between elementary filters. The outputs of the summing layer are here treated as the weights for the final estimation. Basing on the concept of PNN, the output of the summing layer is in fact the probability of the statement, that input vector belongs to one of the classes (models). The research has been carried out in Statistica Neural Network and the recording of sample data came from PC based tracking simulator programmed by author.

Concept and scenarios

The first step in the research was to determine the value for which the probability shall be calculated, as this would become pattern, input and output values. The most obvious criterion is the rate of turn. However, rate of turn for unfiltered data, calculated from two consecutive measurement, is in case of vessels usually very big, so the average of temporary rate of turn and rate of turn from last minute was taken into account. As the second coordinate of input vector, the standard deviation of averaged rate of turn was proposed. Another proposal was to use the same values, but for the data filtered with GRNN filter adjusted for manoeuvring.

The third approach based on the idea of variance analysis like in [12]. The input vector had a form of (18), in which variance of Vx and Vy for last 20 steps (one minute of estimation) is calculated.

$$v(k) = \begin{bmatrix} \mathrm{var}\left[Vxe\left(k/k-20\right)\right] \\ \mathrm{var}\left[Vye\left(k/k-20\right)\right] \end{bmatrix} \qquad (18)$$

The data for the experiment came from 100 Monte Carlo simulation for each model. Five models has been proposed (based on [5]). The simulated manoeuvres dynamics were established according to earlier presented classes. Figure 2 presents a cluster diagram for input vectors based on variance calculations.

For each model 60 values have been recorded, what gives the total number of 300 training patterns. This set has been divided into training set (200 samples), validating set (50 samples) and

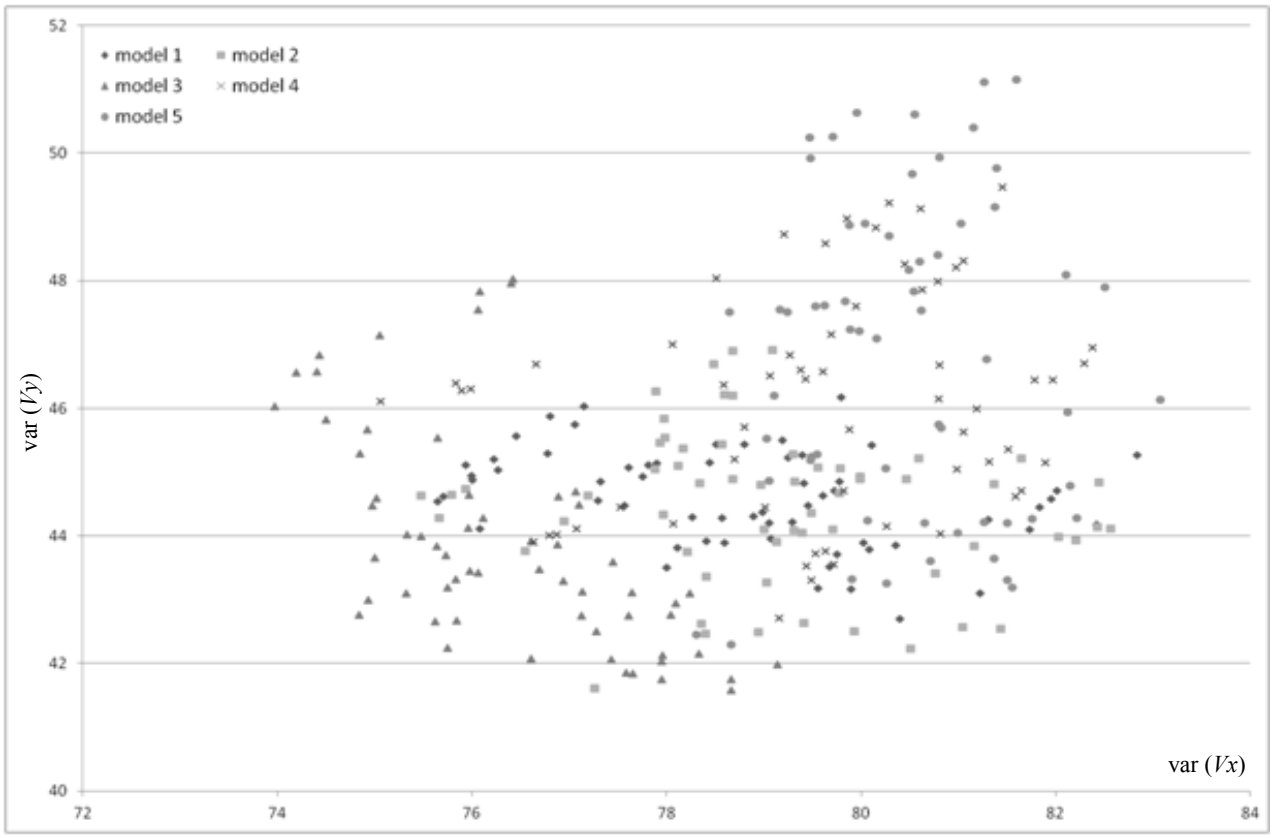

Fig. 2. Cluster diagram for patterns in PNN (variances of *Vx* and *Vy*)

training set (50 samples). Thus, the structure of a network included 200 hidden neurons and 5 neurons in summing layer. The network was trained with the use of Network Wizard and adjusted manually to obtain the best results.

Results

The results of the network for classification problems can be observed in Statistica Neural Network in the form of so called Classification Statistics. They show how many cases from each set has been correctly classified and how many incorrectly. The information is also given about number of classification to each of the classes.

Classification statistics for vector with *Vx* and *Vy* variances for unfiltered data are presented in table 1.

The data in table 1 are presented jointly for training, validating and testing sets. It can be noticed that majority of cases are classified correctly. Most of incorrectly classified cases belongs to neighbor classes, which might have happened as some of the manoeuvres dynamics were just at the border of classes.

Very important advantage of PNN is that not only most probable class is identified, but also that probability for all the classes are calculated. This is presented in figure 3. This is an example of calculated weights for all models for a few cases. It can be seen that the cases belongs to model 4, however PNN also suggests that they are close to model 5.

Table 1. Classification statistics for vector with *Vx* and *Vy* variances for unfiltered data

	Model 1	Model 2	Model 3	Model 4	Model 5
all	60	60	60	60	60
correct	46	47	57	51	52
incorect	14	13	3	9	8
unclassified	0	0	0	0	0
model 1	46	5	0	4	1
model 2	9	47	3	2	2
model 3	0	2	57	0	0
model 4	1	2	0	51	5
model 5	4	4	0	3	52

1.733e-06	0	0	0.9999983	1.835e-16
0	0	0	0.7030838	0.2969162
0	0	0	0.9999999	5.932e-08
0	0	0	1	1.445e-10
0	0	0	0.6960306	0.3039694
0	0	0	1	9.835e-22
0	0	0	0.999947	5.297e-05
0	0	0	1	3.792e-35
0	0	0	1	5.396e-13
0	0	0	1	0

Fig. 3. Probabilities for all the models for a few cases (example)

Jointly results of the research are presented in table 2.

Table 2. Results of classification for different input vectors

	Unfiltered Variance		Unfiltered ROT		Filtered ROT	
	number	%	number	%	number	%
all	300	100	300	100	300	100
correct	253	84.33	210	70.00	213	71.00
incorect	47	15.67	90	30.00	87	29.00
incorect (neighbour allowed)	20	6.67	57	19.00	42	14.00
model 1	56	18.67	58	19.33	70	23.33
model 2	63	21.00	66	22.00	66	22.00
model 3	59	19.67	57	19.00	50	16.67
model 4	59	19.67	59	19.67	69	23.00
model 5	63	21.00	60	20.00	45	15.00

It can be noticed, that the results for rate of turn vector are quite similar for unfiltered and for filtered data, but the unfiltered values seems to be more reliable. In case of filtered ROT major mismatching appeared for models 1, 3 and 5. The best results however was achieved with the vector of variances. About 15% of cases were classified incorrectly, however if neighbor miss-matching are allowed the number of incorrect classification is reduced to less than 7%.

Conclusions

The paper presents different possible approaches to interaction between elementary filters in multiple model neural filters. Different methods for making a decision about manoeuvre and numerical methods for interaction were theoretically presented. They are different for neural filer, comparing to numerical filters due to lack of Kalman matrixes. The innovation and its covariance has to be estimated based on output values from GRNN filter.

Also the neural method has been proposed for interaction between filters – the use of probabilistic neural networks is suggested. PNN gives the probability for each model matching to analyzed input vector. These can be directly used as weights in multiple model filter.

The results show that PNN might be an interesting alternative for numerical methods. In future another values for mode matching shall be considered (like proposed in [12]) and the direct comparison to the methods used in numerical multiple model filtering shall be performed.

References

1. MAGILL D.T.: Optimal Adaptive Estimation of Sampled Stochastic Processes. IEEE Trans.
2. Automatic Control, AC-10:434–439, 1965.
3. BAR SHALOM Y., LI X.R., KIRUBARJAN T.: Estimation with Applications to Tracking and Navigation: Theory Algorithms and Software. John Wiley & Sons, Inc., NY USA, 2001.
4. LI X.R, JILKOV V.P.: A Survey of Manoeuvring Target Tracking. Part V: Multiple-Model Methods. IEEE Transactions on Aerospace and Electronic Eystems, Vol. 41, 2005.
5. KAZIMIERSKI W.: Determining of marine radar target movement models for the needs of multiple model neural tracking filetr. Deutsche Gesellschaft für Ortung und Navigation, e.V. Inaternational Radar Symposium IRS 2011, Leipzig, Niemcy 07–09.09.2011, 611–616.
6. WOŁEJSZA P.: Statistical analysis of real radar target course and speed changes for the needs of multiple model tracking filter. Scientific Journals Maritime University of Szczecin 30(102), 2012, 166–169.
7. LI X.R., JILKOV V.P.: A Survey of Manoeuvring Target Tracking. Part V: Multiple-Model Methods. IEEE Transactions on Aerospace and Electronic Eystems, Vol. 41, 2005.
8. JUSZKIEWICZ W., STATECZNY A.: GRNN Cascade Neural Filter for Tracked Target Manoeuvre Estimation. Neural Networks and Soft Computing, Zakopane 2000.
9. KAZIMIERSKI W.: Two-stage General Regression Neural network for radar target tracking. Polish Journal of Environmental Studies, Vol. 17, No. 3B, 2008.
10. STATECZNY A. (ed.): Radar navigation. GTN, Gdańsk 2011 (in Polish).
11. LI X.R, JILKOV V.P.: A Survey of Manoeuvring Target Tracking. Part IV: Decision-Based Methods, Proc. of SPIE Conference on Signal and Data Processing of Small Targets, Orlando USA, 2002.
12. SPECHT D.F.: Probabilistic Neural Networks. Neural Networks, Vol. 3, 1990, 109–118.