Taylor & Francis
Taylor & Francis Group

Automatika

Journal for Control, Measurement, Electronics, Computing and Communications

ISSN: 0005-1144 (Print) 1848-3380 (Online) Journal homepage: http://www.tandfonline.com/loi/taut20

Nonlinear state estimation using neural-cubature Kalman filter

Zhiyong Miao, Yi Zhang, Kun Zhao & Fan Xun

To cite this article: Zhiyong Miao, Yi Zhang, Kun Zhao & Fan Xun (2017) Nonlinear state estimation using neural-cubature Kalman filter, Automatika, 58:3, 347-353, DOI: 10.1080/00051144.2018.1447272

To link to this article: https://doi.org/10.1080/00051144.2018.1447272

© 2018 The Author(s). Published by Informa UK Limited, trading as Taylor & Francis Group.

Published online: 21 Mar 2018.

Submit your article to this journal

Article views: 96

View Crossmark data

Full Terms & Conditions of access and use can be found at
http://www.tandfonline.com/action/journalInformation?journalCode=taut20

AUTOMATIKA, 2018
VOL. 58, NO. 3, 347–353
https://doi.org/10.1080/00051144.2018.1447272

Taylor & Francis
Taylor & Francis Group

SHORT PAPER

OPEN ACCESS Check for updates

Nonlinear state estimation using neural-cubature Kalman filter

Zhiyong Miao, Yi Zhang, Kun Zhao and Fan Xun

Beijing Aerospace Automatic Control Institute, Beijing, China

ABSTRACT
The cubature Kalman filter (CKF) has been widely used in solving nonlinear state estimation problems because of many advantages such as satisfactory filtering accuracy and easy implementation compared to extended Kalman filter and unscented Kalman filter. However, the performance of CKF may degrade due to the uncertainty of the nonlinear dynamic system model. To solve this problem, a neural-cubature Kalman filter (NCKF) algorithm containing a multilayer feed-forward neural network (MFNN) in CKF is proposed to further improve the estimation accuracy and enhance the robustness of CKF. In the proposed NCKF algorithm, the MFNN was used to modify the nonlinear state estimation of CKF as the measurements were processed, and the CKF was used as both a state estimator and an online training paradigm simultaneously. The experimental results show that the estimation accuracy and robustness of the proposed method are better than those of the CKF, square-root CKF and particle filter.

ARTICLE HISTORY
Received 8 July 2017
Accepted 27 February 2018

KEYWORDS
Cubature Kalman filter; nonlinear state estimation; neural-cubature Kalman filter; multilayer feed-forward neural network

1. Introduction

Over the past several decades, extended Kalman filter (EKF) has been used as the core of nonlinear state estimation problems [1–4]. However, the use of EKF results in a poor performance if the process or measurement model is highly nonlinear [5]. In addition, the computation of Jacobian matrices is also a heavy computational burden in practical situations [6]. As a better alternative to EKF, unscented Kalman filter (UKF) has been proposed to solve highly nonlinear state estimation problems [7]. UKF avoids the computation of Jacobian matrices and uses a deterministic sampling approach to determine the mean and covariances with sigma points [8–10]. In general, UKF performs better than EKF in nonlinear state estimation problems. However, the unscented transformation of UKF is potentially unstable due to the possible negative weight on the centre point in practical applications [11]. Although some other nonlinear filters such as particle filter (PF) [12] and Gaussian PF [13] are more accurate and stable than UKF, they suffer from the "curse of dimensionality" when applied to high-dimensional systems [14].

Recently, owing to satisfactory filtering accuracy, cubature Kalman filter (CKF) based on the spherical–radial cubature rule has received much attention in solving nonlinear state estimation problems [6,11,14,15]. However, for nonlinear dynamic systems with large uncertainties such as target tracking and long-term orbit uncertainty propagation [16,17], the performance of CKF may degrade due to the uncertainty of system mode. To make the CKF robust, a neural-CKF (NCKF) algorithm is proposed in this study. In the proposed NCKF, a multilayer feed-forward neural network (MFNN) was used as a regulator to approximate the difference between the prior model used in the prediction steps of the CKF and the actual model dynamics, and the CKF was used to train the neural network (NN) as well as to estimate the states and weights of NN. The simulation results show that the proposed method has a higher estimation accuracy and robustness than the CKF.

The rest of this manuscript is organized as follows. The NN points are described in Section 2. The proposed NCKF algorithm is described in Section 3. The performances of NCKF, CKF SCKF and PF are compared in Section 4. The conclusions and future work are provided in Section 5.

2. NN points

Although both radial basis function NN (RBFNN) and MFNN can approximate any continuous functional relationship to any degree of accuracy, given enough neurons, RBFNN requires more neurons/basis functions than MFNN if trained in a sequential mode. This may severely limit its application in solving a complex estimation problem. In contrast, MFNN can avoid the change in the architecture of the overall network when the NNs are trained, because its architecture can be fixed before through empirical experiments [18–23]. Therefore, MFNN is more suitable for sequential online applications.

In general, weight links, adders and activation functions are the three major components in a basic model

CONTACT Zhiyong Miao mzy407@126.com, miaozhiyong@hrbeu.edu.cn

© 2018 The Author(s). Published by Informa UK Limited, trading as Taylor & Francis Group.
This is an Open Access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

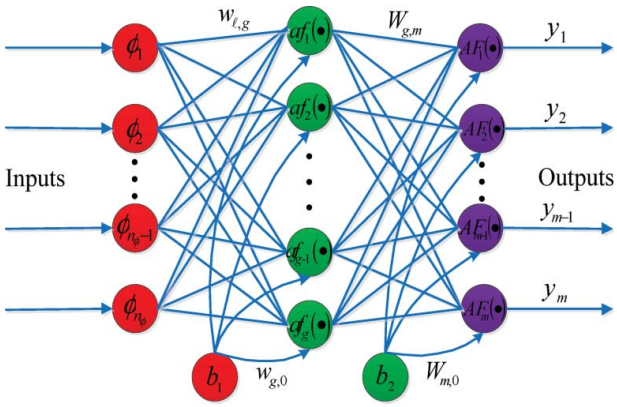

Figure 1. A simple MFNN containing input signals, a hidden layer and an output layer.

of neurons. The weight links are used to link the input layer, hidden layer and output layer. The adders are used to add the input signals that are weighted by the respective synapses of the neuron, and the activation functions are used to limit the amplitude of neuron output and the final output [20,21]. Then, the outputs of hidden and output layers of a neuron can be expressed as follows:

$$\begin{aligned} \text{out}_{\text{hidden}} &= af\left(w_{i,j}\varphi_i\right) + b_1 \\ \text{out}_{\text{output}} &= AF\left(W_{j,m}\text{out}_{\text{hidden}}\right) + b_2 \end{aligned} \tag{1}$$

where φ_i is the input signal; $w_{i,j}$ and $W_{j,m}$ are the input and output weights, respectively; b_1 and b_2 are the biases of the hidden layer and output layer, respectively; $af(\bullet)$ and $AF(\bullet)$ are the activation functions for hidden neurons and output neurons, respectively.

A simple MFNN is shown in Figure 1. If the biases (b_1, b_2) are interpreted as the weights acting on the inputs clamped to 1, the weights and biases can be represented as joint description "weights." Then, the mathematical formula of MFNN can be expressed as follows:

$$y_m(t) = AF_m\left[\sum_{j=1}^{n_h} W_{j,m} af_j\left(\sum_{i=1}^{n_\varphi} w_{i,j}\varphi_i + w_{j,0}\right) + W_{m,0}\right] \tag{2}$$

where n_φ is the number of input signals, n_h is the number of hidden neurons, $w_{j,0}$ is the bias of input weights and $W_{m,0}$ is the bias of output weights. Note that two or three hidden layers are enough, and the nonlinear hyperbolic tangent sigmoid function is used as the activation function. This enhances the approximation capabilities of NN and reduces the impact of noise [20–23].

3. Proposed NCKF algorithm

In this study, the filtering problem of nonlinear dynamic systems was considered; its state-space model can be expressed by a pair of difference equations in discrete time [15]:

$$x_k = f(x_{k-1}) + \nu_{k-1} \tag{3}$$

$$z_k = h(x_k) + \ell_k \tag{4}$$

where x_k and z_k are the states and measurements at time k; $f(\bullet)$ and $h(\bullet)$ are the nonlinear functions for the dynamic and measurement models, respectively;ν_k is the processed Gaussian noise with zero mean and covariance q_k; similarly, ℓ_k is the measured Gaussian noise with zero mean and covariance r_k.

Then, the error between the true model and priori model can be expressed as follows:

$$e_k = f_{\text{true}}(x_k) - f_{\text{priori}}(x_k) \tag{5}$$

Equation (5) shows that the true model can be obtained from an accurate e_k. However, the e_k cannot be measured directly. As stated in Section 2, the MFNN can approximate the functional relationship; therefore, the e_k can be well approximated using the MFNN as follows:

$$(\text{NN}(x_k, \eta_k, \lambda_k) - e_k) \to 0 \tag{6}$$

where η_k is the joint description "weights" of the input weight $w_{i,j}$ and bias $w_{j,0}$, and λ_k is the joint description "weights" of the output weight $W_{j,m}$ and bias $W_{m,0}$.

According to Equation (6), a new error ε_k can be expressed as follows:

$$\varepsilon_k = f_{\text{true}}(x_k) - f_{\text{priori}}(x_k) - NN\left(x_k, \eta_k, \lambda_k\right) \tag{7}$$

where $\|e_k\| \ll \|\varepsilon_k\|$.

From the above analysis, a more accurate dynamic system model can be expressed as follows:

$$x_{k+1} = f(x_k) + \text{NN}(x_k, \eta_k, \lambda_k) + \tilde{\nu}_k \tag{8}$$

where $\tilde{\nu}_k$ is the process Gaussian noise with zero mean and covariance $\tilde{Q}_k$.

Then, the generic NCKF algorithm can be summarized in the following steps:

Step 1: State augment

$$\tilde{x}_k = \begin{bmatrix} x_k \\ \eta_k \\ \lambda_k \end{bmatrix} \tag{9}$$

Suppose that the number of x_k is u_0. Therefore, the dimension of η_k is $\left(n_\varphi \times u_0\right) \times 1$, and the dimension of λ_k is $\left(n_\varphi \times n_h\right) \times 1$. Note that, the augmented state vector of NCKF contains both the state estimates and weights (input and output) of the NN. The states x_k are considered as the input to the NN; the weights of NN are the "parameters" of function approximator

that can be used to model something in the noise [21–23].

Step 2: Initialization

Set $\hat{\tilde{x}}_0 = E[\tilde{x}_0]$, $\tilde{P}_0 = \mathrm{cov}[\tilde{x}_0]$, and generate the cubature points $\chi_l, (l = 1, 2, \ldots, 2\mu)$. Note that $\mu = (n_\phi \times u_0 + n_\phi \times n_h) \times 1$ is the dimension of $\tilde{x}_k$.

Step 2: Time update

$$\tilde{P}_{k-1} = S_{k-1} S_{k-1}^T \tag{10}$$

$$\hat{\tilde{x}}_{k|k-1} = \frac{1}{2\mu} \sum_{l=1}^{\mu} \left(f\left(S_{k-1}\chi_l + \hat{\tilde{x}}_{k-1}\right) + \begin{bmatrix} NN\left(\hat{x}_{k-1}, \hat{\eta}_{k-1}, \hat{\lambda}_{k-1}\right) \\ \hline \mathbf{0} \end{bmatrix} \right) \tag{11}$$

$$\tilde{P}_{k|k-1} = \frac{1}{2\mu} f\left(S_{k-1}\chi_l + \hat{\tilde{x}}_{k-1}\right) f\left(S_{k-1}\chi_l + \hat{\tilde{x}}_{k-1}\right)^T - \hat{\tilde{x}}_{k|k-1}\hat{\tilde{x}}_{k|k-1}^T + \tilde{Q}_k \tag{12}$$

Step 3: Measurement update

$$\tilde{P}_{k|k-1} = S_{k|k-1} S_{k|k-1}^T \tag{13}$$

$$\hat{z}_{k|k-1} = \frac{1}{2\mu} \left(\sum_{l=1}^{\mu} h\left(S_{k|k-1}\chi_l + \hat{\tilde{x}}_{k|k-1}\right) \right) \tag{14}$$

$$\tilde{P}_{\tilde{x}z} = \frac{1}{2\mu} \left(S_{k|k-1}\chi_l + \hat{\tilde{x}}_{k|k-1}\right) h\left(S_{k|k-1}\chi_l + \hat{\tilde{x}}_{k|k-1}\right)^T - \hat{\tilde{x}}_{k|k-1}\hat{z}_{k|k-1} T \tag{15}$$

$$\tilde{P}_{zz} = \frac{1}{2\mu} h\left(S_{k|k-1}\chi_l + \hat{\tilde{x}}_{k|k-1}\right) h\left(S_{k|k-1}\chi_l + \hat{\tilde{x}}_{k|k-1}\right)^T - \hat{z}_{k|k-1}\hat{z}_{k|k-1} T + r_k \tag{16}$$

$$\tilde{K}_k = \tilde{P}_{\tilde{x}z} \tilde{P}_{zz}^{-1} \tag{17}$$

$$\hat{\tilde{x}}_k = \hat{\tilde{x}}_{k|k-1} + \tilde{K}_k (z_k - \hat{z}_{k|k-1}) \tag{18}$$

$$\tilde{P}_k = \tilde{P}_{k|k-1} - \tilde{K}_k \tilde{P}_{zz} \tilde{K}_k^T \tag{19}$$

Equations (9) and (18) show that the weights of the MFNN are updated by the CKF. Furthermore, Equation (11) shows that the MFNN can adaptively approximate the errors between the true model and priori model in the prediction steps of the CKF.

4. Simulations

In this section, three numerical examples were used to evaluate the estimation accuracy and robustness of the proposed NCKF.

4.1. Bearings-only tracking

In this subsection, the NCKF was compared with the CKF, square-root CKF (SCKF) and PF in bearings-only tracking example; this has been widely analysed owing to its practical application potential [24,25]. The nonlinear system model of bearings-only tracking can be expressed as follows:

$$x_k = \begin{bmatrix} 0.99 & 0 \\ 0 & 1 \end{bmatrix} x_{k-1} + w_{k-1} \tag{20}$$

$$z_k = \tan^{-1}\left(\frac{x_{2,k} - \sin(k)}{x_{1,k} - \cos(k)}\right) + v_k \tag{21}$$

where $x_k = (x_{1,k}, x_{2,k})^T = (s, t)^T$ is the position in the s–t plane. Note that the function $\tan^{-1}(\bullet)$ can be implemented using the command atan2(•) in MATLAB 2012a.

Although the actual output and estimated output along with estimated states can be used to show the estimation accuracy of the proposed method, the results of single simulation cannot fully reflect the trends. To quantitatively evaluate the filtering performance, the simulation results are based on 200 Monte Carlo runs. The process noise $q_k \sim N(0, Q_k)$ with $Q_k = \mathrm{diag}([10^{-5} \quad 10^{-5}])$, the measurement noise $r_k \sim N(0, R_k)$ with $R_k = 0.05$, the initial state $x_0 = [20 \quad 5]^T$ and x_0, P_0, Q_0 and R_0 are its associated covariance $P_0 = \mathrm{diag}([400 \quad 25])$. To implement the NCKF algorithm, the state x_k, the associated covariance $\tilde{P}_k$ and process noise variance $\tilde{Q}_k$ should be augmented as follows:

$$\tilde{x}_k = [x_k \quad \eta_k \quad \lambda_k]^T \tag{22}$$

$$\tilde{P}_k = \left[\begin{array}{c:c} P_k & \mathbf{0} \\ \hdashline \mathbf{0} & \mathbf{I}_{\mathrm{num-weights}} \end{array}\right] \tag{23}$$

$$\tilde{Q}_k = \left[\begin{array}{c:c} Q_k & \mathbf{0} \\ \hdashline \mathbf{0} & \mathbf{I}_{\mathrm{num-weights}} \end{array}\right] \tag{24}$$

where the values of η_k and λ_k are initialized between 0 and 1, respectively.

Data analysis criterion: the root-mean-square error (RMSE) was used as the benchmark to compare the performances of NCKF, CKF, SCKF and PF. The RMSEs at time k for x_1 and x_2 are defined as follows:

$$\mathrm{RMSE}_{x1}(k) = \sqrt{\frac{1}{N_{mc}} \sum_{\mu=1}^{N_{mc}} \left(x_{1,k}^{\mu} - \hat{x}_{1,k}^{u}\right)^2} \tag{25}$$

$$\mathrm{RMSE}_{x2}(k) = \sqrt{\frac{1}{N_{mc}} \sum_{\mu=1}^{N_{mc}} \left(x_{2,k}^{\mu} - \hat{x}_{2,k}^{u}\right)^2} \tag{26}$$

where $x_{1,k}^{\mu}, \hat{x}_{1,k}\mu$ are the true and estimated $x_{1,k}$ at the μth Monte Carlo run, respectively; $x_{2,k}^{\mu}, \hat{x}_{2,k}\mu$ are the true and estimated $x_{2,k}$ at the μthMonte Carlo run, respectively

The simulation results of the RMSEs for x_1 and x_2 are shown in Figures 2 and 3, respectively. Further-

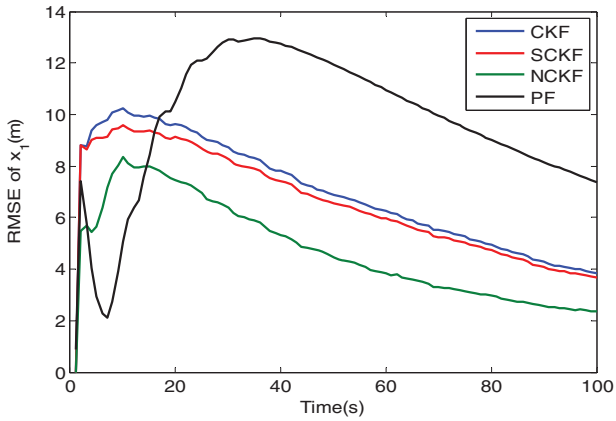

Figure 2. RMSEs of x_1 for CKF, SCKF, NCKF and PF.

more, the corresponding statistics of RMSEs for the four filters are shown in Table 1.

Figures 2 and 3 show that the NCKF performs better than the CKF, SCKF and PF. This is because the CKF was used as an estimator and a training paradigm in the NCKF. As an estimator, the CKF estimates the states and the input and output weights of NN. As a training paradigm, it is driven by the same residuals as the state estimator and ensures that the residuals are as small as possible; it approximates the difference between the prior model used in the prediction steps of the estimator and the actual model dynamics. Therefore, NCKF can improve the performance by estimating the weights of NN, which in turn is used to modify the state estimate predictions of the filter as the measurements are processed. In addition, although the sequential importance sample PF uses 500 particles, its performance is not as good as the CKF. Although the PF can be improved by increasing the number of particles or using some advanced techniques [26,27], the computational complexity will significantly increase. The PF was not studied further, because it is out of the scope of this study. A comparison of standard deviation (Std) and mean value shown in Table 1 provides the same conclusion: the NCKF is superior to the other three filters in this test.

The tracking algorithms were coded with MATLAB, and the simulations were run on an Intel Core Duo

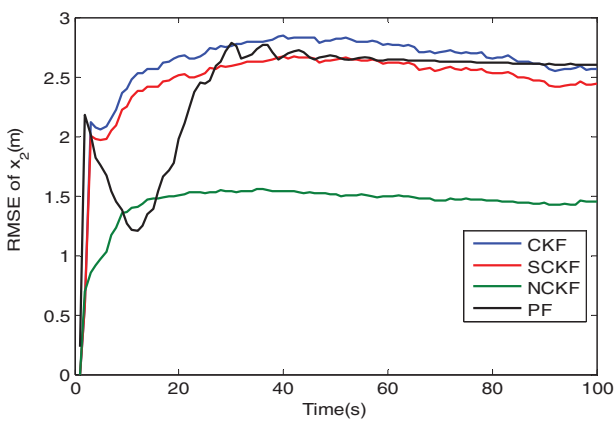

Figure 3. RMSEs of x_2 for CKF, SCKF, NCKF and PF.

Table 1. Statistics of RMSEs for CKF, SCKF, NCKF and PF.

	Items	CKF	SCKF	NCKF	PF
x_1 (m)	Mean	6.962	6.586	4.727	9.708
	Std	2.137	2.007	1.907	2.768
x_2 (m)	Mean	2.615	2.468	1.436	2.404
	Std	0.378	0.353	0.204	0.488
Computational time (s)		8.143	9.581	25.57	428.4

CPU processor at 2.00 GHz. Table 1 shows that the NCKF needs more time than the CKF and NCK, because its state vector contains both the state estimates and the weights (input and output) of the NN. Table 1 shows that the PF has a large computational burden because it needs particle resampling in each step.

4.2. Manoeuvring target tracking

In this section, the NCKF was compared with the CKF, SCKF and PF in a manoeuvring target tracking application; this has been used as a benchmark problem to evaluate the performance of CKF algorithm [11,15]. The dynamic equation of the target tracking can be modelled as follows:

$$\begin{bmatrix} \xi_k \\ \dot{\xi}_k \\ \varsigma_k \\ \dot{\varsigma}_k \\ \psi_k \end{bmatrix} = \begin{bmatrix} 1 & \frac{\sin(\psi_{k-1}T)}{\psi_{k-1}} & 0 & \frac{\cos(\psi_{k-1}T)-1}{\psi_{k-1}} & 0 \\ 0 & \cos(\psi_{k-1}T)-1 & 0 & -\sin(\psi_{k-1}T) & 0 \\ 0 & \frac{1-\cos(\psi_{k-1}T)}{\psi_{k-1}} & 1 & \frac{\sin(\psi_{k-1}T)}{\psi_{k-1}} & 0 \\ 0 & \sin(\psi_{k-1}T) & 0 & \cos(\psi_{k-1}T) & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} \xi_{k-1} \\ \dot{\xi}_{k-1} \\ \varsigma_{k-1} \\ \dot{\varsigma}_{k-1} \\ \psi_{k-1} \end{bmatrix} + \nu_{k-1} \tag{27}$$

where ξ_k and $\dot{\xi}_k$ are the position and velocity in Xdirection, respectively; ς_k and $\dot{\varsigma}_k$ are the position and velocity in Ydirection, respectively; Δtrepresents the sampling interval; ψ_k represents the unknown turn rate; ν_{k-1} is the white Gaussian noise with zero mean and covariance Q_{k-1}:

$$q_{k-1} = \begin{bmatrix} \frac{a_1\Delta t^3}{3} & \frac{a_1\Delta t^2}{2} & 0 & 0 & 0 \\ \frac{a_1\Delta t^2}{2} & a_1\Delta t & 0 & 0 & 0 \\ 0 & 0 & \frac{a_1\Delta t^3}{3} & \frac{a_1\Delta t^2}{2} & 0 \\ 0 & 0 & \frac{a_1\Delta t^2}{2} & a_1\Delta t & 0 \\ 0 & 0 & 0 & 0 & a_2\Delta t \end{bmatrix} \tag{28}$$

where $a_1 = 0.1$; $\Delta t = 1$s; and $a_2 = 1.75 \times 10^{-4}$.

The measurement equation with two sensors can be expressed as follows:

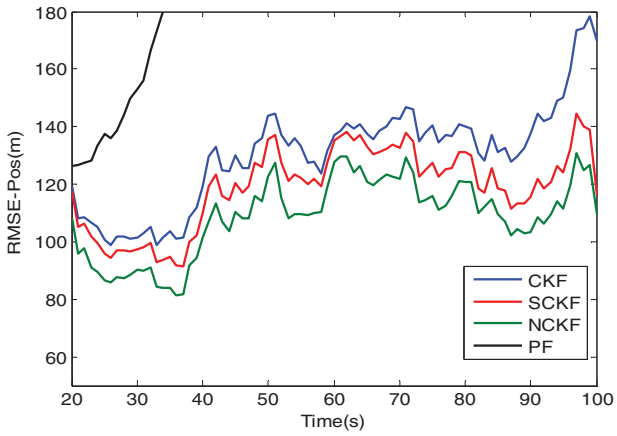

Figure 4. RMSEs of the position for CKF, SCKF, NCKF and PF.

$$\theta_{k,d} = \tan^{-1}\left(\frac{\varsigma - \varsigma_{\mathrm{ref},d}}{\xi - \xi_{\mathrm{ref},d}}\right) + \ell_{k,d} \quad d = 1, 2. \tag{29}$$

where d is the sensor index; $\ell_{k,1}$ and $\ell_{k,2}$ are the white Gaussian measurement noise with zero mean and covariance $R_{k,1} = \left(\sqrt{30\mathrm{mrad}}\right)^2$ and $R_{k,2} = \left(\sqrt{40\mathrm{mrad}}\right)^2$, respectively; the locations of the two sensors $\left(\xi_{\mathrm{ref},d}, \varsigma_{\mathrm{ref},d}\right)$ are assumed as follows:

$$\xi_{\mathrm{ref},1} = -10,000\ \mathrm{m},\ \zeta_{\mathrm{ref},1} = -10,000\ \mathrm{m} \tag{30}$$

$$\xi_{\mathrm{ref},2} = 10,000\ \mathrm{m},\ \zeta_{\mathrm{ref},2} = 10,000\ \mathrm{m} \tag{31}$$

For a fair comparison, the simulation results are based on 200 Monte Carlo runs. In each Monte Carlo run, the initial estimate value was generated randomly from $N(\hat{x}_0 \mid x_0, \mathrm{P}_0)$. The initial state x_0 and the associated covariance P_0 were assumed as follows:

$$x_0 = [\,1000\ \mathrm{m} \quad 300\mathrm{m/s} \quad 1000\ \mathrm{m} \quad 0\mathrm{m/s} \quad -3^\circ/\mathrm{s}\,]^T \tag{32}$$

$$P_0 = \mathrm{diag}([\,100\ \mathrm{m}^2 \quad 10\mathrm{m}^2/\mathrm{s}^2 \quad 100\ \mathrm{m}^2 \quad 10\mathrm{m}^2/\mathrm{s}^2 \quad 100\mathrm{mrad}^2/\mathrm{s}^2\,]) \tag{33}$$

To implement the NCKF algorithm, the state x_0, the associated covariance $\tilde{P}_0$ and process noise variance $\tilde{Q}_0$

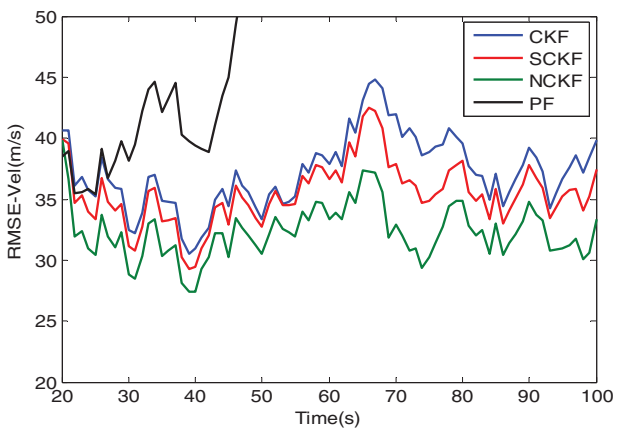

Figure 5. RMSEs of the velocity for CKF, SCKF, NCKF and PF.

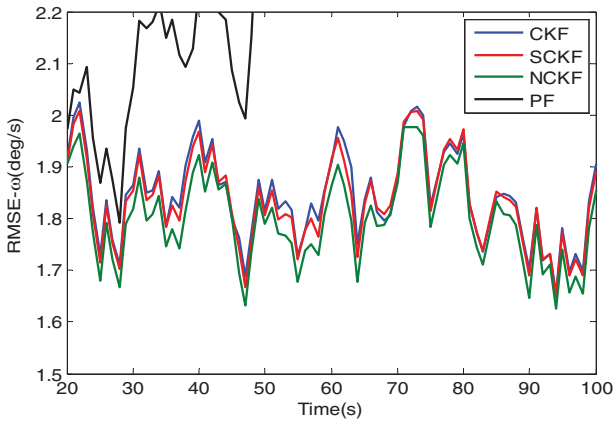

Figure 6. RMSEs of the turn rate for CKF, SCKF, NCKF and PF.

should be augmented as Equations (22)–(24). The RMSE at time k for position was defined as follows:

$$\mathrm{RMSE}_{\mathrm{pos}}(k) = \sqrt{\frac{1}{N_{mc}}\sum_{g=1}^{N_{mc}}\left(\left(\xi_k^g - \hat{\xi}_k^g\right)^2 + \left(\varsigma_k^g - \hat{\varsigma}_k^g\right)^2\right)} \tag{34}$$

where $\left(\xi_k^g, \varsigma_k^g\right)$ and $\left(\hat{\xi}_k^g, \hat{\varsigma}_k g\right)$ are the true and estimated positions at the gth Monte Carlo run, respectively. Then, the RMSE in velocity and turn rate can be formulated using Equation (34).

The simulation results of the RMSEs in position, velocity and turn rate are shown in Figures 4–6, respectively. Furthermore, the corresponding statistics of RMSEs for CKF, SCKF, NCKF and PF are shown in Table 2.

Figures 4–6 show that the CKF, SCKF and NCKF can track the position, velocity and turn rate. However, the accuracy of NCKF outperforms that of the CKF and SCKF, especially in the position tracking and velocity tracking. This is because the NCKF can improve the performance by estimating the weights of NN, which in turn is used to modify the state estimate as the measurements are processed. Furthermore, a comparison of the Std and mean value shown in Table 2 provides the same conclusion: the NCKF is better than the CKF and SCKF under the Gaussian conditions. Figures 4–6 also show that the PF with 500 particles does not perform a good estimation for this application. Note that NaN represents Not-a-Number in MATLAB.

To test the robustness of NCKF algorithm, the measurement noise $\ell_{k,d}$ was deliberately converted to a

Table 2. Statistics of RMSEs for CKF, SCKF, NCKF and PF.

	Items	CKF	SCKF	NCKF	PF
Position (m)	Mean	122.4	113.4	105.1	NaN
	Std	32.46	28.64	27.05	NaN
Velocity (m/s)	Mean	36.65	34.23	31.72	NaN
	Std	7.181	6.754	6.293	NaN
Turn rate (deg/s)	Mean	1.835	1.828	1.809	NaN
	Std	0.184	0.183	0.1822	NaN
Computational time (s)		12.06	12.38	55.32	582.68

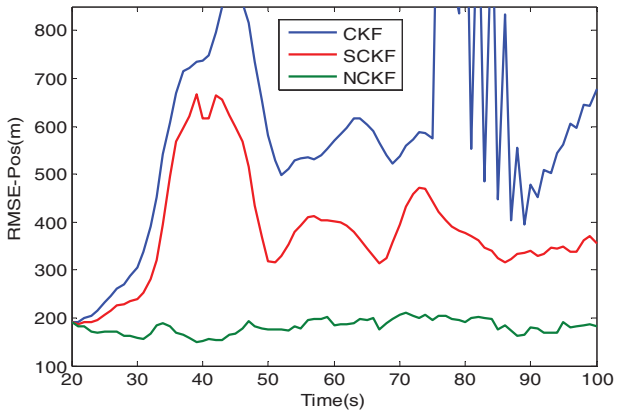

Figure 7. RMSEs of the position for CKF, SCKF and NCKF (robustness test).

non-Gaussian form as follows:

$$0.5N\left(0, 10 \times 10^{-5}\right) + 0.5N\left(0, 5 \times 10^{-5}\right) \tag{35}$$

Previous literature [28,29] shows the PF diverges in non-Gaussian measurement noise assumption. Therefore, the proposed NCKF was only compared to CKF and SCKF in this robustness test. The RMSEs of the positions, velocity and turn rate are shown in Figures 7–9, respectively.

Equation (35) shows that the equivalent measurement noise variance R_k of the above Gaussian mixture model was used in filters. Therefore, a mismatching exists between the true measurement variance and the assumed variance. Figures 7–9 show that the performances of CKF, SCKF and NCKF degrade in this situation. Moreover, the CKF diverges in position, velocity and turn rate estimation due to a mismatch between the true measurement noise and the non-Gaussian assumption. However, the NCKF can maintain a satisfactory performance in this scenario; it uses the MFNN to approximate the error caused by the mismatch mentioned above and modify the state estimation accordingly. Thus, the robustness of NCKF is better than those of the CKF and SCKF.

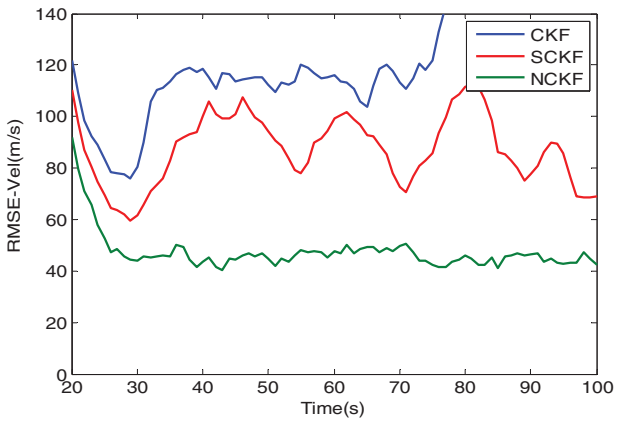

Figure 8. RMSEs of the velocity for CKF, SCKF and NCKF (robustness test).

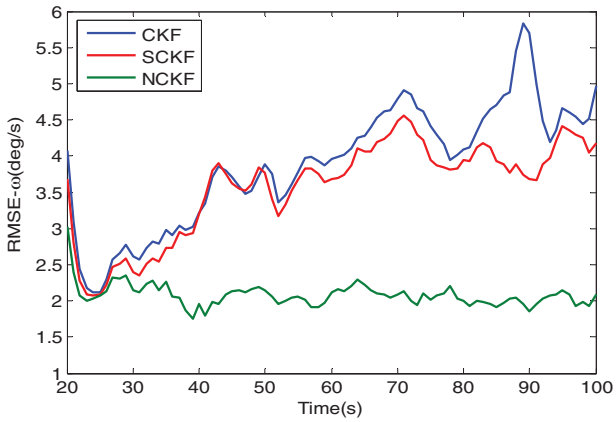

Figure 9. RMSEs of the turn rate for CKF, SCKF and NCKF (robustness test).

5. Conclusions

In this study, a NCKF algorithm was proposed to enhance the accuracy and robustness of CKF. In the proposed method, a MFNN was used to modify the state estimates and approximate the uncertainty of the system model; and the CKF was used to estimate the states and the weights of MFNN. The performance of the proposed method was compared with CKF, SCKF and PF via three numerical examples. The simulation results show that the accuracy and robustness of the NCKF is better than those of the CKF, SCKF and PF.

The success of the proposed method shows an encouraging direction for online estimation with NCKF algorithm. Although, the proposed method performs well in this study, the computational burden of NCKF is larger than CKF and SCKF. Therefore, the low computational burden of NCKF, and the performance between the NCKF and the robust filters will be studied in the future.

Acknowledgments

The author thanks Kun Zhao and Fan Xun for their support in simulation and revision.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This research is supported by the National Natural Science Foundation of China [grant number 61374208], [grant number 61603365].

References

[1] Gelb A, editor. Applied optimal estimation. Cambridge (MA): MIT Press; 1974.

[2] Nassar S, Niu X, El-Sheimy N. Land-vehicle INS/GPS accurate positioning during GPS signal blockage periods. J Surv Eng. 2007;133(3):134–143.
[3] Vasconcelos JF, Silvestre C, Oliveira P. INS/GPS aided by frequency contents of vector observations with application to autonomous surface crafts. IEEE J Oceanic Eng. 2011;36(2):347–363.
[4] Chiang KW, Lin CA, Duong TT. The performance analysis of the tactical inertial navigator aided by non-GPS derived references. Remote Sens. 2014;6(12):12511–12526.
[5] Aung H, Low KS, Goh ST. State-of-charge estimation of lithium-ion battery using square root spherical unscented Kalman filter (Sqrt-UKFST) in nanosatellite. IEEE Trans Power Electron. 2015;30(9):4774–4783.
[6] Zhang XC, Guo CJ. Cubature Kalman filters: derivation and extension. Chin Phys B. 2013;22(12):128401.
[7] Ali J, Mirza MR UB. Performance comparison among some nonlinear filters for a low cost SINS/GPS integrated solution. Nonlinear Dyn. 2010;61(3):491–502.
[8] Leven WF, Lanterman AD. Unscented Kalman filters for multiple target tracking with symmetric measurement equations. IEEE Trans Autom Control. 2009;54(2):370–375.
[9] Xiong K, Wei CL, Liu LD. Robust unscented Kalman filtering for nonlinear uncertain systems. Asian J Control. 2010;12(3):426–433.
[10] Shen H, Karlgaard CD. Sensitivity reduction of unscented Kalman filter about parameter uncertainties. IET Radar Sonar Navig. 2015;9(4):374–383.
[11] Arasaratnam I, Haykin S, Hurd TR. Cubature Kalman filtering for continuous-discrete systems: theory and simulations. IEEE Trans Signal Process. 2010;58(10):4977–4993.
[12] Zhou R, Teng J. An improved resampling algorithm for particle filtering in small target tracking. J Coast Res. 2015;73(sp1):600–605.
[13] Kotecha JH, Djuric PM. Gaussian sum particle filtering. IEEE Trans Signal Process. 2003;51(10):2602–2612.
[14] Zhang XC, Teng YL. A new derivation of the cubature Kalman filters. Asian J Control. 2014;16(5):1501–1510.
[15] Jia B, Xin M, Cheng Y. High-degree cubature Kalman filter. Automatica. 2013;49(2):510–518.
[16] Arasaratnam I, Haykin S. Cubature Kalman filters. IEEE Trans Autom Control. 2009;54(6):1254–1269.
[17] Huang Y, Zhang Y. Design of high-degree student's t-based cubature filters. Circuits Syst Signal Process. 2017;1–20.
[18] Haykin S. Neural networks: a comprehensive foundation. New York: Macmillan College Publishing Company; 1994.
[19] Demuth HB, Beale MH, De Jess O, et al. Neural network design. Boston (MA): PWS Publishing Co.; 1996.
[20] Chiang KW, Chang HW. Intelligent sensor positioning and orientation through constructive neural network-embedded INS/GPS integration algorithm. Sensors. 2010;10(10):9252–9285.
[21] Chiang KW, Huang YW, Li CY, et al. An ANN embedded RTS smoother for an INS/GPS integrated positioning and orientation system. Appl Soft Comput. 2011;11(2):2633–2644.
[22] Chiang KW, Chang HW, Li CY, et al. An artificial neural network embedded position and orientation determination algorithm for low cost MEMS INS/GPS integrated sensors. Sensors. 2009;9(4):2586–2610.
[23] Miao Z, Shi H, Zhang Y, et al. Neural network-aided variational Bayesian adaptive cubature Kalman filtering for nonlinear state estimation. Meas Sci Technol. 2017;28(10):105003.
[24] Zhang YG, Huang YL, Wu ZM, et al. Quasi-stochastic integration filter for nonlinear estimation. Math Probl Eng. 2014; 2014: Article ID 967127.
[25] Yong-Gang Z, Yu-Long H, Ning L, et al. Embedded cubature Kalman filter with adaptive setting of free parameter. Signal Process. 2015;114(9):112–116.
[26] Zhang Y, Huang Y, Li N, et al. Particle filter with one-step randomly delayed measurements and unknown latency probability. Int J Syst Sci. 2016;47(1):209–221.
[27] Yu Long H, Yong Gang Z. Particle smoother for nonlinear systems with one-step randomly delayed measurements. Asian J Control. 2017;19(2):813–819.
[28] Jia B, Xin M. Multiple sensor estimation using a new fifth-degree cubature information filter. Trans Inst Meas Control. 2015;37(1):15–24.
[29] Zhang XC. Cubature information filters using high-degree and embedded cubature rules. Circuits Syst Signal Process. 2014;33(6):1799–1818.