

Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering

<http://pii.sagepub.com/>

Application of artificial neural networks to weighted interval Kalman filtering

Amit Motwani, Sanjay K Sharma, Robert Sutton and Phil Culverhouse

Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering 2014 228: 267

originally published online 3 February 2014

DOI: 10.1177/0959651813520148

The online version of this article can be found at:

<http://pii.sagepub.com/content/228/5/267>

Published by:



<http://www.sagepublications.com>

On behalf of:



[Institution of Mechanical Engineers](http://www.imeche.org)

Additional services and information for *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering* can be found at:

Email Alerts: <http://pii.sagepub.com/cgi/alerts>

Subscriptions: <http://pii.sagepub.com/subscriptions>

Reprints: <http://www.sagepub.com/journalsReprints.nav>

Permissions: <http://www.sagepub.com/journalsPermissions.nav>


Citations: <http://pii.sagepub.com/content/228/5/267.refs.html>

>> [Version of Record](#) - Apr 24, 2014

[OnlineFirst Version of Record](#) - Feb 3, 2014

[What is This?](#)

Application of artificial neural networks to weighted interval Kalman filtering

Proc IMechE Part I:
J Systems and Control Engineering
2014, Vol. 228(5) 267–277
© IMechE 2014
Reprints and permissions:
sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/0959651813520148
pii.sagepub.com


Amit Motwani¹, Sanjay K Sharma¹, Robert Sutton¹ and Phil Culverhouse²

Abstract

The interval Kalman filter is a variant of the traditional Kalman filter for systems with bounded parametric uncertainty. For such systems, modelled in terms of intervals, the interval Kalman filter provides estimates of the system state also in the form of intervals, guaranteed to contain the Kalman filter estimates of all point-valued systems contained in the interval model. However, for practical purposes, a single, point-valued estimate of the system state is often required. This point value can be seen as a weighted average of the interval bounds provided by the interval Kalman filter. This article proposes a methodology based on the application of artificial neural networks by which an adequate weight can be computed at each time step, whereby the weighted average of the interval bounds approximates the optimal estimate or estimate which would be obtained using a Kalman filter if no parametric uncertainty was present in the system model, even when this is not the case. The practical applicability and robustness of the method are demonstrated through its application to the navigation of an uninhabited surface vehicle.

Keywords

Interval Kalman filter, artificial neural network, robust estimation

Date received: 21 August 2013; accepted: 29 November 2013

Introduction

In a recent study by Annamalai et al.,¹ a system for determining the heading of an uninhabited surface vehicle (USV) based on an interval Kalman filter (IKF) was explored. The IKF, designed upon imprecise knowledge of the vehicle's yaw dynamics, provided upper and lower bounds to the statistically optimal estimate of its heading angle at each time instant. However, the guidance and control of the vehicle require a point-valued estimate of its heading. Since the optimal estimate must lie within the IKF interval estimate,² it must correspond to some weighted average of the interval boundaries. The problem remains on how to infer the appropriate weighting value and is the objective of this study. This article proposes a method that provides a good approximation to the optimal weight by using appropriately simulated data and artificial neural networks (ANNs).

Problem formulation

Consider the following stochastic–deterministic state-space model of a system's dynamics

$$x(k+1) = \mathbf{A}_m x(k) + \mathbf{B}_m u(k) + \boldsymbol{\omega}(k) \quad (1)$$

$$y(k) = \mathbf{C}_m x(k) + \nu(k) \quad (2)$$

with

$$\begin{aligned} \mathbf{A}_m &= \begin{bmatrix} 1.002 & 0 \\ 0 & 0.9945 \end{bmatrix}, \mathbf{B}_m = \begin{bmatrix} 6.354 \\ -4.699 \end{bmatrix} \times 10^{-6}, \\ \mathbf{C}_m &= \frac{180}{\pi} [34.13 \quad 15.11], \boldsymbol{\omega}(k) \sim N(\mathbf{0}, \mathbf{Q}_m), \nu(k) \sim N(0, R_m), \\ \mathbf{Q}_m &= \text{cov}(\boldsymbol{\omega}) = \text{diag}\{1, 1\} \times 10^{-10}, \\ R_m &= \text{var}(\nu) = 4, T_s = 1 \end{aligned} \quad (3)$$

where $u(k)$ is the known input to the system, $\boldsymbol{\omega}(k)$ represents a random input disturbance, $y(k)$ is the measured output and $\nu(k)$ represents a random measurement noise. Assuming that the random processes follow

¹Marine and Industrial Dynamic Analysis Group, School of Marine Science and Engineering, Plymouth University, Plymouth, UK

²School of Computing and Mathematics, Plymouth University, Plymouth, UK

Corresponding author:

Amit Motwani, Marine and Industrial Dynamic Analysis Group, School of Marine Science and Engineering, Plymouth University, Drake Circus, Plymouth, Devon PL4 8AA, UK.

Email: amit.motwani@plymouth.ac.uk

zero-mean Gaussian distributions, then the classical Kalman filter (KF) approach based on combining model predictions with actual measurements may be used to obtain statistically optimal estimates of the system output. Let such a filter, based upon the above model, be denoted as KF-1 (the KF equations are detailed in Appendix 1).

Now, let it be supposed that the modeller is uncertain of the precise values of the vector \mathbf{B}_m and in fact declares that

the values of \mathbf{B} could not be ascertained with complete exactitude, however, it is possible to warrant that they are not further departed from these than by an amount equating to twenty five per cent of the same.

Upon this revelation, it is apparent that the filter KF-1 will no longer supply an optimal estimate if the actual values of the state equation's input vector, \mathbf{B} , depart from those of \mathbf{B}_m , and in particular, the difference between the measurements $y(k)$ and the model's predicted output $\mathbf{C}\mathbf{x}(k)$ will no longer have a zero-mean value.² In order to try and account for imprecisely modelled values while maintaining a KF-like structure, Chen et al.³ proposed describing the system dynamics using intervals. Consider the following interval model

$$\mathbf{x}^I(k+1) = \mathbf{A}^I \mathbf{x}^I(k) + \mathbf{B}^I u(k) + \boldsymbol{\omega}(k) \quad (4)$$

$$y^I(k) = \mathbf{C}^I \mathbf{x}^I(k) + \nu(k) \quad (5)$$

with

$$\begin{aligned} \mathbf{A}^I &= \begin{bmatrix} [1.002, 1.002] & [0, 0] \\ [0, 0] & [0.9945, 0.9945] \end{bmatrix}, \\ \mathbf{B}^I &= [\mathbf{B}_m - 0.25 \times \text{abs}(\mathbf{B}_m), \mathbf{B}_m + 0.25 \times \text{abs}(\mathbf{B}_m)] \\ &= \begin{bmatrix} [0.75 \times 6.354, 1.25 \times 6.354] \\ [1.25 \times (-4.699), 0.75 \times (-4.699)] \end{bmatrix} \times 10^{-6}, \\ \mathbf{C}^I &= \frac{180}{\pi} [[34.13, 34.13] \quad [15.11, 15.11]], \\ \boldsymbol{\omega}(k) &\sim N(\mathbf{0}, \mathbf{Q}), \quad \nu(k) \sim N(0, R), \\ \mathbf{Q} &= \text{cov}(\boldsymbol{\omega}) = \text{diag}\{1, 1\} \times 10^{-10}, \quad R = \text{var}(\nu) = 4, \\ T_s &= 1 \end{aligned} \quad (6)$$

in which the components of \mathbf{A}^I , \mathbf{B}^I and \mathbf{C}^I , as well as those of the system state vector \mathbf{x}^I and output y , are now given by interval values rather than ordinary point values. For simplicity but without loss of generality, the example illustrated here only contains interval model coefficients with non-zero widths in the vector \mathbf{B}^I , as it is assumed that all other coefficients are modelled precisely. Based on this interval model, an IKF, which provides interval-valued estimates, can be designed.³

Let it also be supposed that the true dynamics of the system, while not corresponding exactly to the values given in equation (3), are contained within the interval model (equations (4)–(6)) and are given by

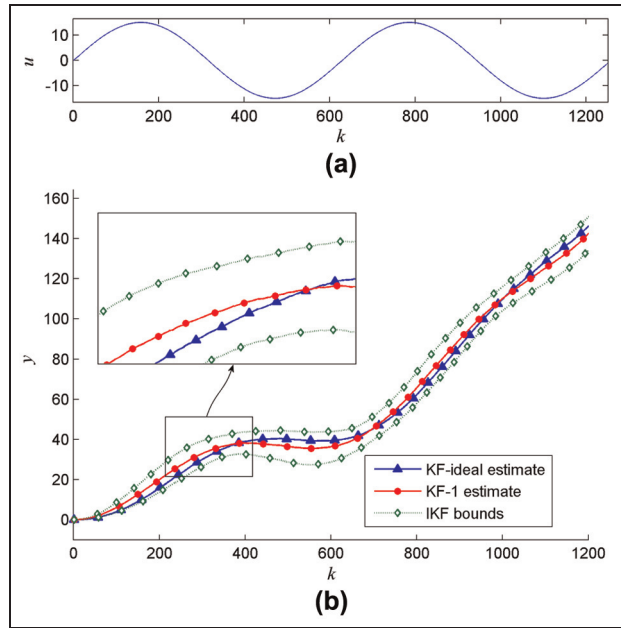


Figure 1. KF-ideal, IKF and KF-1 estimates of the output of the system to the sinusoidal input $u(k) = 15 \sin(0.01k)$ and (b) system output estimates. KF: Kalman filter; IKF: interval Kalman filter.

$$\mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}u(k) + \boldsymbol{\omega}(k) \quad (7)$$

$$y(k) = \mathbf{C}\mathbf{x}(k) + \nu(k) \quad (8)$$

with

$$\begin{aligned} \mathbf{A} &= \mathbf{A}_m = \begin{bmatrix} 1.002 & 0 \\ 0 & 0.9945 \end{bmatrix}, \\ \mathbf{B} &= \mathbf{B}_m - 0.25 \times \mathbf{B}_m = \begin{bmatrix} 0.75 \times 6.354 \\ 1.25 \times (-4.699) \end{bmatrix} \times 10^{-6}, \\ \mathbf{C} &= \mathbf{C}_m = \frac{180}{\pi} [34.13 \quad 15.11], \\ \boldsymbol{\omega}(k) &\sim N(\mathbf{0}, \mathbf{Q}), \quad \nu(k) \sim N(0, R), \\ \mathbf{Q} &= \mathbf{Q}_m = \text{cov}(\boldsymbol{\omega}) = \text{diag}\{1, 1\} \times 10^{-10}, \\ R &= R_m = \text{var}(\nu) = 4, \quad T_s = 1 \end{aligned} \quad (9)$$

Then, a KF based upon it would provide a statistically optimal estimate of the state vector and system output. Let such a KF be denoted by KF-ideal.

Consider the following arbitrarily chosen sinusoidal signal as input to the system (Figure 1(a))

$$u(k) = 15 \sin(0.01k) \quad (10)$$

where the notation $u(k)$ implies $u(kT_s)$ and is used for convenience. Then, based on simulated values of $\boldsymbol{\omega}(k)$ and $\nu(k)$ and the respective filter models, the estimates of the output by the three filters can be calculated and are shown in Figure 1(b) (the KF and IKF equations are detailed in Appendix 1). It is to be noted that in all cases, the measurements are simulated using the true system's dynamics (equations (7)–(9)) and not the

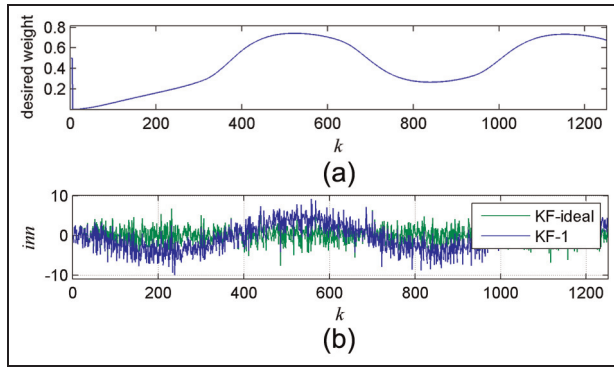


Figure 2. (a) Sequence of weights calculated so that the weighted average of the IKF boundaries coincides with the KF-ideal estimate and (b) innovation sequence of KF-ideal and KF-1. IKF: Interval Kalman filter .

respective models since they represent the actual measurements.

Several observations ensue. First, it can be seen that the KF-1 estimate deviates from the KF-ideal estimate due to the incorrect model assumed by the former. However, both of these lie within the bounds of the IKF interval estimate, as the latter must in principle contain every single KF estimate arising from a model contained within the interval model.³ Finally, it can also be verified that the arithmetic average of the IKF bounds approximately coincides with the KF-1 estimate.

Let $y^{IKF}(k)$ be the IKF estimate of the system output. If a weight $w \in [0, 1]$ is chosen at each time step, then the weighted average of its bounds, henceforth the weighted interval Kalman filter (wIKF) estimate, is given by

$$y^{wIKF}(k) = y^{IKF-}(k) + w(k)[y^{IKF+}(k) - y^{IKF-}(k)]$$

with

$$y^{IKF+}(k) = \max\{y^{IKF}(k)\} \text{ and } y^{IKF-}(k) = \min\{y^{IKF}(k)\} \tag{11}$$

and lies within the boundaries of the IKF interval estimate. In addition, based upon the previous observations, there exists a particular value of w for which the wIKF estimate matches the KF-ideal estimate. (Note also that the KF-1 estimate can be computed from equation (11) with $w(k) = 0.5$.)

Figure 2(a) depicts, at each time step, these desired weights that produce a wIKF estimate coincident with that of KF-ideal, easily calculated from equation (11) when the KF-ideal estimate is known. The key question is *can these weights be calculated in practice without the knowledge of the true system dynamics, and hence, without the availability of the KF-ideal estimate?* The answer, fortunately, is yes, as is explained in what follows.

It is well established that under optimal conditions, the innovations of the KF, or difference between a

priori prediction and measured output, should comprise a white noise sequence.⁴ However, under erroneous modelling assumptions, the optimality of the KF estimate is lost,⁵ resulting in an innovation sequence that ceases to correspond to white noise. The innovation sequences of both the KF-ideal and KF-1 estimates of Figure 1(b) are shown in Figure 2(b).

It seems likely that there should exist a deterministic relationship between the innovation sequence and the desired weighting sequence, and as such, it should be possible to model such a relationship. It is also well established that ANNs are capable of replicating complex cause-effect relationships, enabling one to predict the output of such processes for new inputs.⁶

An ANN as the missing link

The recurrent multi-layer perceptron (RMLP)-type ANN shown in Figure 3 was trained using as input the innovation sequence of KF-1 and as target the desired weights (Figure 2). Due to the fact that the relationship between innovations and desired weights in most likelihood depends not just on the instantaneous values but on the *trends* of the innovations as well, these trends were incorporated into the ANN model by considering six consecutive values of the innovations for each desired output, consisting of the present value as well as the previous five values: $inn(k), inn(k-1), \dots, inn(k-5)$. Although not apparently necessary, another feature was added to the input of the network: the width of the IKF interval, $\Delta ikf(k) = y^{IKF+}(k) - y^{IKF-}(k)$. The addition of this extra input was seen to enhance the performance of the network, the reason for which will be discussed in a later section.

It was also observed that the use of feedback from the output also helped increase the network’s accuracy, and so, five time-delayed values from the output were fed back as inputs to the network. Thus, the combined input to the network at time step k (not counting the bias unit) can be described as

$$x(k) = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_6 \\ x_7 \\ x_8 \\ \vdots \\ x_{12} \end{bmatrix} = \begin{bmatrix} inn(k) \\ inn(k-1) \\ \vdots \\ inn(k-5) \\ \Delta ikf(k) \\ \hat{w}(k-1) \\ \vdots \\ \hat{w}(k-5) \end{bmatrix} \tag{12}$$

where \hat{w} is the output of the network.

Details of the training process used are given in Appendix 2. The training results are shown in Figure 4(b). The virtue of the fit is evaluated by calculating the mean square error (MSE) between the predicted output \hat{w} and the desired one w_t and comparing it to the MSE

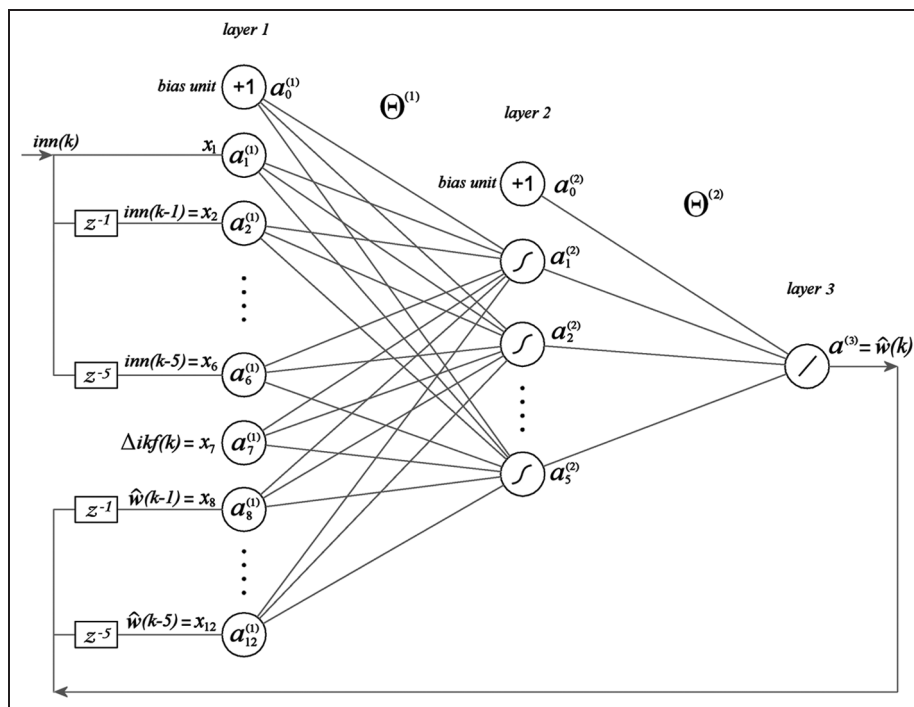


Figure 3. RMLP used, consisting of 12 input units, 5 hidden units and a single output unit. RMLP: recurrent multi-layer perceptron.

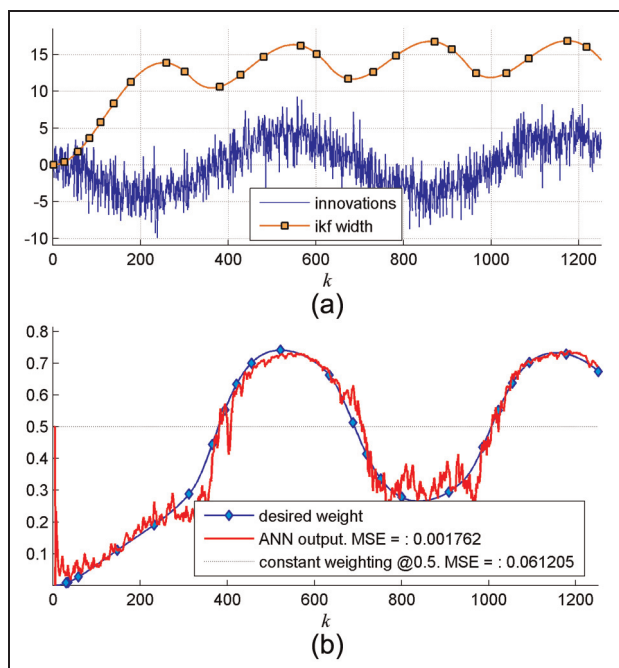


Figure 4. Comparison of desired weighting sequence and output of trained ANN. The MSE between the desired weight and ANN output is 0.001762, compared to 0.061205 between the desired weight and a constant value of 0.5. (a) KF-1 innovation sequence and Δikf and (b) comparison of desired weighting sequence and ANN output.

IKF: interval Kalman filter; ANN: artificial neural network; MSE: mean square error.

between a constant weighting sequence of 0.5 (the default weighting that would be used to select a nominal value from the IKF estimate in the absence of any specific criterion) and w_t . In this case, the MSE decreases from 0.061205 for the latter to 0.001762 for the ANN prediction, a decrease of over 1.5 orders of magnitude.

Figure 4 clearly shows that the trained ANN establishes a mapping between the inputs (innovation sequence of KF-1 and IKF interval width) and the desired weighting. However, it is crucial to investigate whether this model generalises well to new data.

In order to test the trained ANN on new data, two new data sets were generated from new input signals applied to the dynamic system, from which the KF and IKF estimates, desired IKF weighting and KF innovation sequences were generated. These are summarised in Figures 5 and 6. Figure 5(a) depicts a superposition of sinusoidal waveforms of various frequencies and amplitudes used as input to the dynamic system, while Figure 5(b) and (c) show the data set generated from it. Also in Figure 5(c) is the predicted output of the previously trained ANN to the signals shown in Figure 5(b). Similar graphs are shown in Figure 6 for the case in which an input consisting of a square waveform was applied to the dynamic system (equations (7)–(9)). Table 1 summarises the test performances of the trained ANN on these new test sets.

Test case 1.

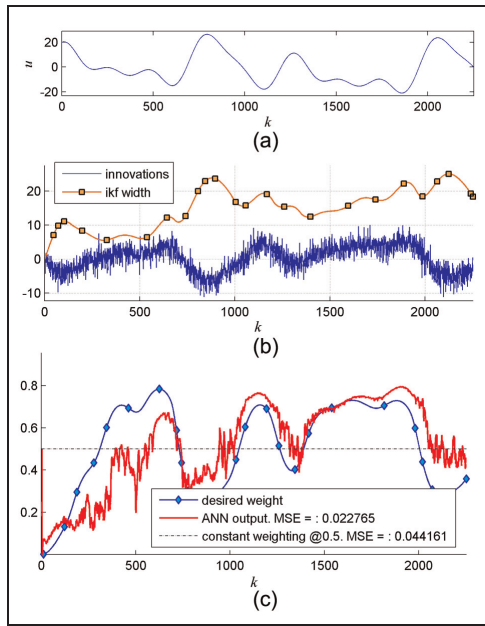


Figure 5. Performance of trained ANN on test set generated from an input of superimposed sinusoids. (a) System input: $u(k) = 5 \sin(0.25k) + 10 \cos(0.15k) + 10 \sin(0.08k) - 8 \sin(0.04k) + 5 \cos(0.05k)$, (b) sequences used to generate ANN input: KF-1 innovation sequence and Δikf and (c) prediction performance: comparison of desired weighting sequence and ANN output. IKF: interval Kalman filter; ANN: artificial neural network; MSE: mean square error.

Test case 2.

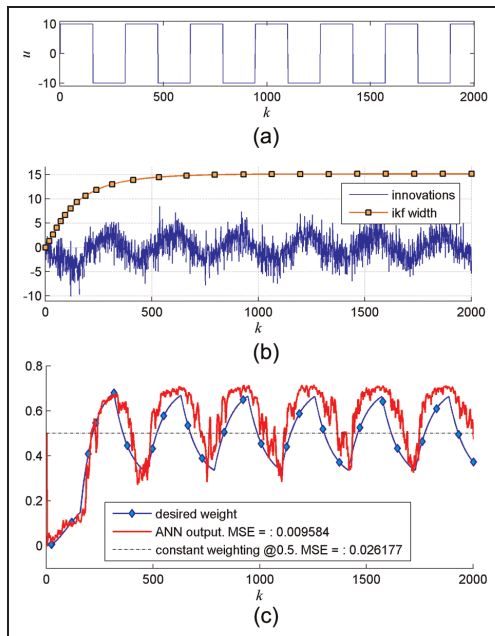


Figure 6. Performance of trained ANN on test set generated from a square wave input. (a) System input: $u(k) = 10\text{sign}(\sin(0.02k))$, (b) sequences used to generate ANN input: KF-1 innovation sequence and Δikf and (c) prediction performance: comparison of desired weighting sequence and ANN output. IKF: interval Kalman filter; ANN: artificial neural networks; MSE: mean square error.

As can be observed, in both test cases, the MSE of the trained ANN output is considerably lower than the mean square difference between the target weighting and the constant weight that represents the arithmetic mean of the IKF boundaries.

Thus far so good; however, the attentive reader may pose the following conundrum: in practice, the real system dynamics may differ from that which was used to generate the data on which the ANN was trained. In fact, this is most likely to be the case, and one would not know the precise values representative of the real system dynamics, for if that were so, then there would be no need for using an IKF in the first place.

Hence, let it now be supposed that the true dynamics of the system are given by

$$x(k + 1) = Ax(k) + Bu(k) + \omega(k) \tag{13}$$

$$y(k) = Cx(k) + \nu(k) \tag{14}$$

with

$$A = A_m = \begin{bmatrix} 1.002 & 0 \\ 0 & 0.9945 \end{bmatrix},$$

$$B = \begin{bmatrix} 0.75 \times 6.354 \\ 0.75 \times (-4.699) \end{bmatrix} \times 10^{-6},$$

$$C = C_m = \frac{180}{\pi} [34.13 \quad 15.11], \quad \omega(k) \sim N(\mathbf{0}, Q), \quad \nu(k) \sim N(0, R),$$

$$Q = Q_m = \text{cov}(\omega) = \text{diag}\{1, 1\} \times 10^{-10},$$

$$R = R_m = \text{var}(\nu) = 4, \quad T_s = 1 \tag{15}$$

rather than the values given in equation (9). One should wonder whether the ANN trained under the previous (initial) assumptions would still be able to correlate innovations with desired weightings in this new context. Let an input sequence given by equation (10) (Data 1) now be applied to the system described by equations (13)–(15) and the corresponding KF and IKF estimates be calculated. Figures 7(a)–(c) show the input, the KF-1 innovation sequence together with the IKF interval widths, and a comparison of the desired weighting sequence with the output of the trained ANN, respectively. The MSE of the ANN prediction with respect to the desired weighting is 0.007434, a 72.17% reduction compared to the value of 0.026711 that results if a constant sequence of 0.5 is used.

This ascertains that the ANN trained from data generated through simulation by using some assumed system dynamics can still be applied successfully to the prediction of the desired IKF weighting sequence even when the true system dynamics differ from those assumed for training, as long as both lie within the intervals constitutive of the interval model.

A case study presenting how these concepts may be used in practice is detailed in the following section.

An ANN-guided wIKF for the navigation of a USV

In this section, the ideas described previously are applied to the problem of estimating the heading angle

Table 1. Test performances for the ANN trained on data set shown in Figure 4.

Test case 1			Test case 2		
MSE_0.5	MSE_ANN	Reduction	MSE_0.5	MSE_ANN	Reduction
0.044161	0.022765	48.45%	0.026177	0.009584	63.39%

MSE: mean square error; ANN: artificial neural network.

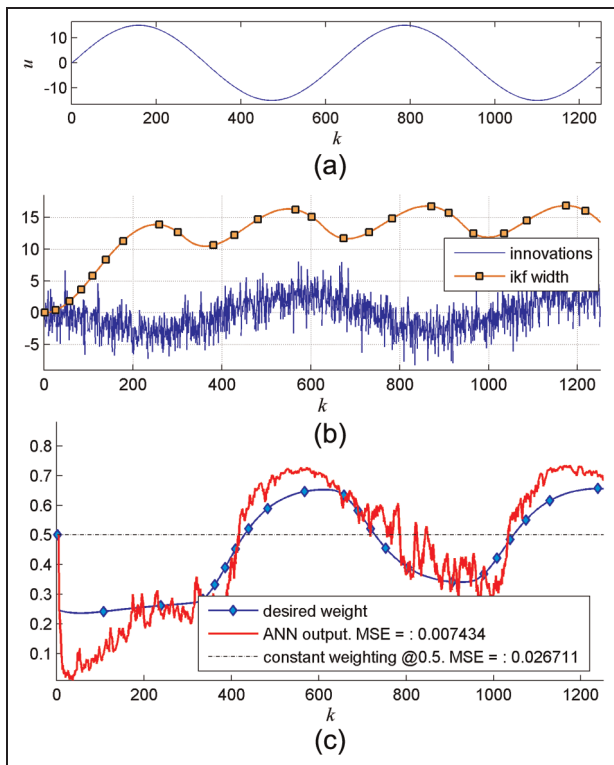


Figure 7. Performance of trained ANN on test set generated from modified system (equations (13)–(15)). (a) System input: $u(k) = 15 \sin(0.01k)$, (b) KF-I innovation sequence and Δikf and (c) prediction performance: comparison of desired weighting sequence and ANN output.

IKF: interval Kalman filter; ANN: artificial neural network; MSE: mean square error.

of an Uninhabited Surface Vehicle (USV). The vehicle in question consists of a twin-hull catamaran driven by two propellers, the difference in speed of which enables the vehicle to steer, and is controlled purposefully to this end. Let n_d represent the difference in revolutions per minute (rpm) of the two propellers, such that the vehicle steers to the left when n_d is positive and to the right when it is negative. In effect, from a control point of view, the vehicle's yaw dynamics has been modelled using system identification (SI) techniques.⁷ The model obtained is precisely that described by equations (1)–(3), where u is the aforesaid differential speed of the propellers in rpm, n_d , and ω models random input disturbances to the system to take into account randomly varying surface effects. Additionally, the vehicle is equipped with a magnetic compass unit that provides the instantaneous heading with a random unbiased root

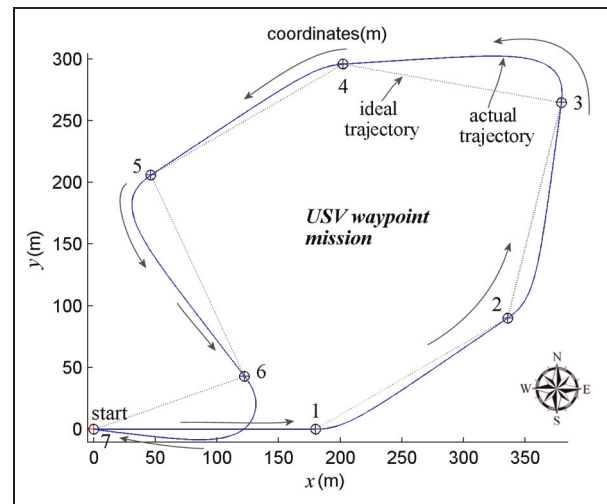


Figure 8. Way-point following mission showing the trajectory followed by the USV.

USV: uninhabited surface vehicle.

mean square (RMS) error of 2° and is given by y in equation (2), where ν represents the compass error.

In a simulation study, the vehicle is given the task of completing a way-point following mission as shown in Figure 8. The trajectory depicted is followed by the vehicle operating under an autonomous guidance and autopilot system based on line of sight (LOS) and proportional–integral–derivative (PID) control, respectively, under the constraint of maintaining a constant forward speed of 3 knot or 1.54 m s^{-1} . Additionally, the value of n_d though calculated by the autopilot is subsequently hard limited to within $\pm 300 \text{ rpm}$, with a maximum permitted variation from one time step to the next of $\pm 20 \text{ rpm}$, reflecting the physical limitations of the hardware. In this simulation, the vehicle's actual heading was assumed available to feed back to the guidance and autopilot systems. Further details of this setup can be found in Annamalai et al.;¹ as for the study undertaken herein, only aspects relative to estimation of the heading of the vehicle given a model of the same and a specified control input are needed.

In order to estimate the heading of the vehicle from the noisy measurements of the compass, a KF may be used. However, as discussed previously, under incorrect modelling assumptions, the KF estimate may become biased. Let it be assumed that although the exact model of the vehicle's dynamics is not known, it is certain to be contained within the interval model described by equations (4)–(6). In this scenario, the technique

described previously may be put into practice: a set of dynamics contained within the interval model may be assumed as the ‘true’ vehicle dynamics, and the estimates of both a KF based on it and a KF based on the nominal model (equations (1)–(3)) can be simulated, together with the interval estimates from an IKF founded on the interval model. The desired weighting sequence can then be obtained as that which is necessary for the wIKF values to match those of the KF estimates that were obtained using the assumed ‘true’ dynamics. Finally, an ANN can be trained to obtain these desired weights from the innovation sequence of biased KF estimates (those obtained from the KF that uses the nominal system model).

In order to train the ANN, rather than use the way-point mission described earlier to generate the required input and target data, a different mission was used. This allows the mission described previously to be used to test the method in order to evaluate its performance. The training mission chosen consists of 14 way-points which are shown in Figure 9(a). These way-points were

chosen to provide a variety of turning angle requirements assuming that the vehicle reaches the way-point along the ideal trajectory. For instance, if the vehicle reaches way-point 1 facing east, then it is initially required to turn 20° towards the left to head towards the following way-point. When way-point 2 is reached, it is required to turn left an additional 40°, and so on until 100° from way-point 5 to way-point 6, before heading back towards the initial coordinates. It then has to repeat the trajectory but this time in a clockwise direction. This forces a large range of the vehicle’s dynamics to be used for generating training data for the ANN and thus should favour its capability of accurate prediction for any standard trajectory.

Another point for consideration is what model to choose from the interval model to represent the ‘true’ dynamics of the vehicle for simulation. Instead of choosing a single model for the whole mission, different sets of models were chosen during different time intervals. The values of **B** for simulating the vehicle’s dynamics during the course of the training mission

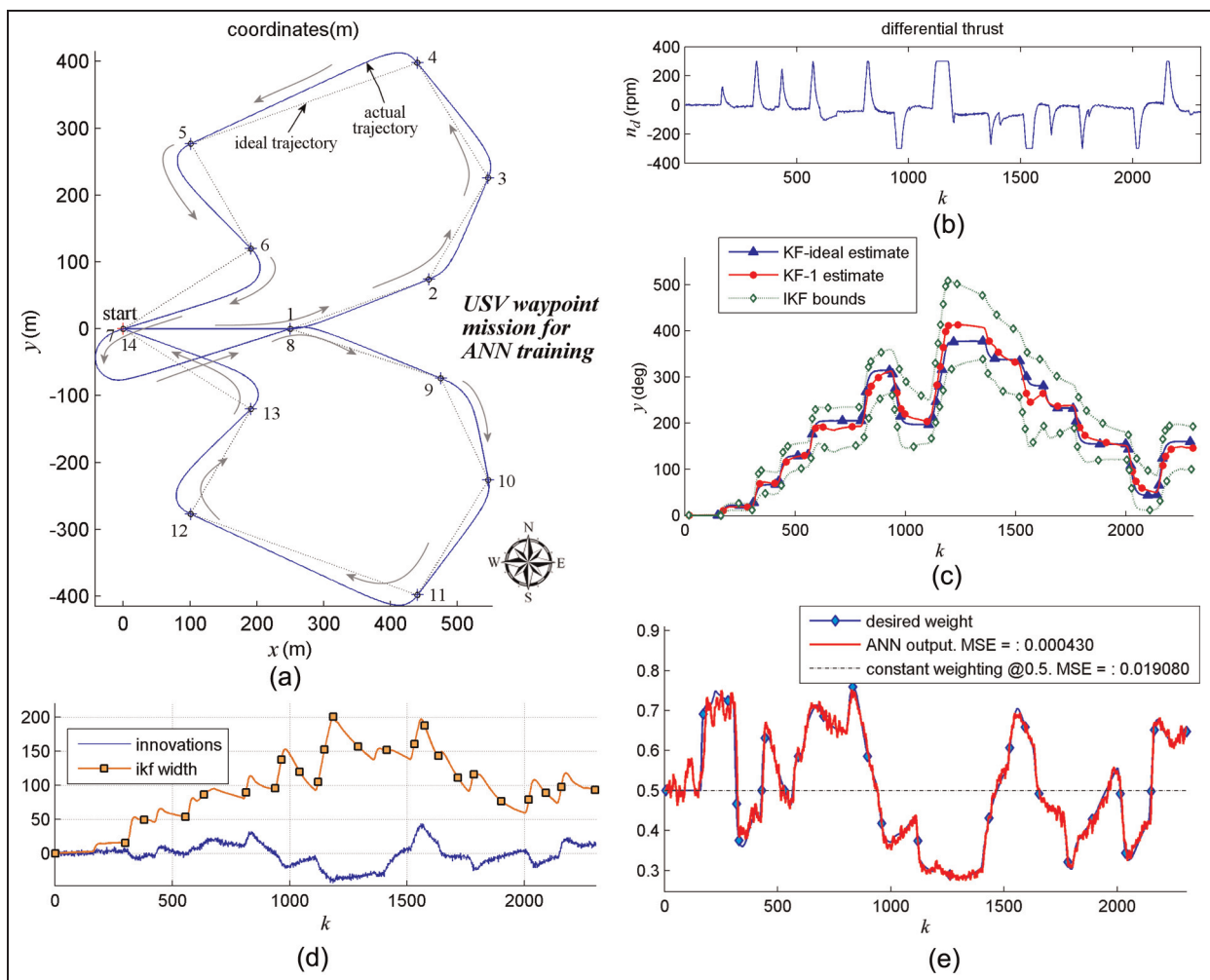


Figure 9. Way-point following mission for training the ANN. (a) Way-point specification and trajectory followed by the USV for the training mission, (b) controller input: differential thrust calculated by the autopilot, (c) heading estimates, (d) KF-I innovation sequence and Δikf and (e) comparison of desired weighting sequence and trained ANN output. USV: uninhabited surface vehicle; ANN: artificial neural network; KF: Kalman filter; IKF: interval Kalman filter; MSE: mean square error.

Table 2. Models for generating test data.

Test case 1	Test case 2	Test case 3
$\mathbf{B}(1) = \mathbf{B}_m(1) + 0.25\mathbf{B}_m(1)$ $\mathbf{B}(2) = \mathbf{B}_m(2) - 0.25\mathbf{B}_m(2)$	$\mathbf{B}(1) = \mathbf{B}_m(1) - 0.25\mathbf{B}_m(1)$ $\mathbf{B}(2) = \mathbf{B}_m(2) + 0.25\mathbf{B}_m(2)$	$\mathbf{B}(1) = \mathbf{B}_m(1) + 0.25\mathbf{B}_m(1)$ $\mathbf{B}(2) = \mathbf{B}_m(2) + 0.25\mathbf{B}_m(2)$

were set to initially coincide with those of \mathbf{B}_m up until reaching the first way-point and subsequently to vary as follows

$$\mathbf{B}(1) = \mathbf{B}_m(1) + 0.25\mathbf{B}_m(1)\text{sign}\left[\sin\left(\frac{2\pi}{T_1}k\right)\right], \quad T_1 = 400 \text{ s} \quad (16)$$

$$\mathbf{B}(2) = \mathbf{B}_m(2) + 0.25\mathbf{B}_m(2)\text{sign}\left[\sin\left(\frac{2\pi}{T_2}k\right)\right], \quad T_2 = 450 \text{ s} \quad (17)$$

The training data set will thus be generated from multiple dynamic systems covering several combinations of values selected from the extremes of the interval model and provides a richer training set than would be generated using a constant set of dynamics for simulating the vehicle for the entire duration of the mission.

Based on these varying dynamics, Figure 9(a) shows the actual path taken by the vehicle using LOS guidance and a PID autopilot, while the PID-generated differential propeller speed is shown in Figure 9(b). For the trajectory followed, both guidance and control systems were given the actual heading of the vehicle. Additionally, a KF was used to estimate the heading based on simulated noisy compass measurements, using the actual dynamics of the vehicle for prediction. The estimates of this KF are labelled as KF-ideal in Figure 9(c). A second KF, one that used the nominal system model instead, was used to generate the estimates labelled as KF-1 in Figure 9(c). Finally, an IKF was also implemented and the interval estimates therefrom are plotted on the same figure.

Figure 9(d) shows the innovation sequence of the biased KF along with the IKF interval widths, both of which are used as inputs to the ANN. Finally, Figure 9(e) shows the desired (target) weighting sequence (calculated so that the wIKF estimate matches the unbiased KF estimate (KF-ideal)). This input and target data set were used to train the ANN of Figure 3. The trained network's output is plotted alongside the target output in Figure 9(e). The training accuracy is quantified by an MSE of 0.000430, in contrast to the average mean difference of 0.019080 between the desired weighting and a constant value of 0.5, some 1.5 orders of magnitude lower.

To test the trained ANN on the initial way-point mission (Figure 8), three test sets were generated by choosing different vehicle dynamics within the interval model (equations (4)–(6)). In all three, the initial vehicle dynamics coincides with that of the nominal model (equations (1)–(3)), and the KF-ideal and KF-1

estimates coincide. From way-point 2 onwards, the values of $\mathbf{B}(1)$ and $\mathbf{B}(2)$ (of the 'true', or simulated, vehicle dynamics) were chosen according to the values shown in Table 2.

For each test case, the innovations of KF-1 and the IKF widths were calculated, as well as the desired weighting sequence (by imposing that the wIKF estimate equals the KF-ideal estimate). The previously trained ANN was then used to predict the desired weight based on innovations and IKF widths. The results for each test case are depicted in Figures 10(a)–(c), respectively, and Table 3 summarises for each one the reduction in MSE with respect to the desired weighting sequence.

The results show that the trained ANN can be applied successfully to predict the desired weights required for the wIKF estimate to approximate the optimal (KF-ideal) estimate. It should also be noted from the graphs that the majority of the error between the predicted and desired weights occurs during the initial stages of the simulation, when the IKF intervals are relatively small. (Although these are not shown for the test cases, it can be observed to be so in Figure 9(d) for the training data and is generally the case due to the nature of IKF interval computations where the initially narrow intervals tend to grow.²⁾ When the IKF widths are small, errors in the weights become less significant, as both IKF bounds are themselves already close to the optimal estimate, and hence, so is any weighted average of these (with weight between 0 and 1). Therefore, a performance measure that takes this into account would flaunt even better numbers than those presented in the table.

It should also be said that the ANN architecture presented here (Figure 3), and its particular characteristics (layer sizes, etc.), was found to provide a good balance between prediction accuracy and number of neurons employed, but is by no means the only valid architecture that can be used, and furthermore, for each system, the most suited type and size of ANN should be explored.

Discussion and conclusion

Although the KF has been used extensively and successfully in numerous applications,⁵ when the model used by the KF is deficient, then as demonstrated in the preceding sections, the estimates tend to become biased. For applications that increasingly require robust state variable estimation at reduced costs, carrying out precise modelling of the system becomes prohibitive. Such

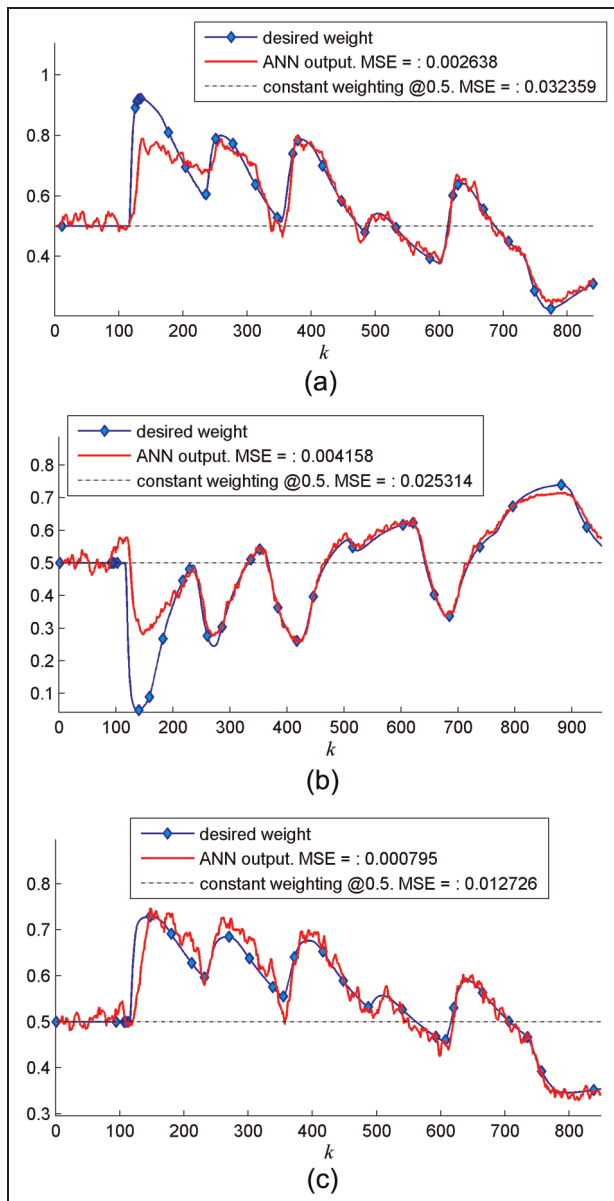


Figure 10. Comparison of desired weighting sequence and trained ANN output for (a) test data 1, (b) test data 2 and (c) test data 3.

ANN: artificial neural network; MSE: mean square error.

is the case, for example, with low-cost USV systems which have seen a recent surge in the number of applications, and for which ever-increasing levels of autonomy are desired at reduced costs.⁸ SI is a popular modelling strategy to this end, but because it is based on empirical data, the model values obtained must be understood to be precise to within a certain tolerance. Even if accurate models could be developed, tolerance to changes in system dynamics is also a quality that is increasingly required; for example, in the case of USVs, varying mission objectives could mean being able to operate under varying payload. For this reason, robust navigation systems that can handle imprecisely modelled or varying dynamics are needed.

The IKF was developed to extend the KF to systems described in terms of intervals rather than precise

point-valued quantities. However, the main problem with interval filtering is that due to the conservative nature of interval computation, the estimates tend to be over-conservative, limiting their practical usage.² In practice, a single estimate is often required, and several studies have been aimed at inferring point-valued estimates from the interval estimates of the IKF. Chui and Chen⁹ suggested using a weighted average of the IKF boundaries, and, in the absence of any weighting criteria, to take the arithmetic average of the boundaries. As demonstrated in this article, the wIKF methodology developed provides estimates that are much improved over taking the simple arithmetic average of the interval bounds. Another method was proposed by Weng et al.¹⁰ in which evolutionary programming is used as a global search method to find the point estimate that minimises the maximum estimation error covariance. However, on the one hand, this method requires running an iterative search algorithm at each time step, and on the other hand, it does not use actual measurement data to infer the desired point-valued estimate, being based on statistical principles alone. In the approach used here, the training of the network is done offline, so that it is only used for prediction during an actual mission. This only requires forward propagation of information through the network, which can be computed efficiently using a vectorised implementation.

Other robust filtering approaches have been proposed which alter the basic hypotheses or structure of the IKF algorithm,^{11,12} resulting in either loss of the optimality quality of the IKF's interval estimate or rigour in the sense of guaranteeing to contain any optimal estimate of a particular realisation of the interval system. In the method presented here, the IKF original estimate is maintained, while its boundaries are simply used to infer a point-valued estimate for practical purposes. Thus, while this latter estimate can be used, referring for example to the case study presented here, as input to a guidance and control system, the IKF intervals themselves can be used to compute guaranteed bounds to the trajectory followed by the vehicle and to trigger an alarm should these bounds permeate into an undesired region.

A pending comment with regard to the particular example used in this article to demonstrate the technique developed is the use of $\Delta ikf(k)$ as additional input to the ANN. Its use was seen to increase the predictive accuracy of the network, especially to correct the scaling of the prediction. A heuristic explanation for this phenomenon is the following. The IKF intervals themselves inevitably tend to widen over-conservatively as mentioned earlier, and in fact, depending on the sharpness of the interval computation, the width may vary significantly. However, they all represent the same 'optimal interval' that would be obtained if interval computation could be carried out with infinite sharpness. In other words, the ANN developed here should be immune to the exact width of the IKF interval and sensitive only to the innovations of the biased KF

Table 3. Test performances for the trained ANN.

Test case 1			Test case 2			Test case 3		
MSE_0.5	MSE_ANN	Reduction	MSE_0.5	MSE_ANN	Reduction	MSE_0.5	MSE_ANN	Reduction
0.032359	0.002638	91.85%	0.025314	0.004158	83.57%	0.012726	0.000795	93.75%

MSE: mean square error; ANN: artificial neural network.

estimate. It thus requires information of the former, which should somehow be incorporated into the ANN prediction process since its target output, the optimal wIKF estimate, is computed from the IKF bounds themselves.

To conclude, it has been demonstrated how an ANN can be trained successfully to use residual KF data (the innovation sequence) to infer advantageous weightings for obtaining point-valued estimates from IKF boundaries, as compared to simply using, for example, the arithmetic mean of the boundaries (which provides similar estimates to that of the KF that uses the incorrect nominal model). The test results for the case study presented here (Table 3) show that the trained ANN is capable of generalising well to new situations. Depending on the application, it is always possible to develop an adequate training set that will enable effective prediction for new missions within the scope of the application. This required analysis, the training process and evaluation of the trained network on a set of new missions can all be done beforehand and via simulation alone, rendering this method cost-effective, reliable and practically realisable.

Declaration of conflicting interests

The authors declare that there is no conflict of interest.

Funding

This work was supported by the Engineering and Physical Sciences Research Council (grant no. EP/I012923/1).

References

1. Annamalai A, Motwani A, Sharma S, et al. A robust navigation technique for integration in the guidance and control of an uninhabited surface vehicle. *The Journal of Navigation* 2013. (Under review)
2. Motwani A, Sharma SK, Sutton R, et al. Interval Kalman filtering in navigation system design for an uninhabited surface vehicle. *J Navigation* 2013; 66(5): 639–652.
3. Chen G, Wang J and Shieh LS. Interval Kalman filtering. *IEEE T Aero Elec Sys* 1997; 33(1): 250–258.
4. Shimkin N. Derivations of the discrete-time Kalman filter. In: *Estimation and identification in dynamical systems* (Lecture Notes, Fall). Department of Electrical Engineering, Israel Institute of Technology, 2009, www.webee.technion.ac.il/people/shimkin/Estimation09
5. Motwani A. *Adaptive and interval Kalman filtering techniques in autonomous surface vehicle navigation: a survey*. MIDAS Technical report: MIDAS.SMSE.2012.TR.002, 4 June 2012. Plymouth: Plymouth University.
6. Abdi H, Valentin D and Edelman B. *Neural networks* (SAGE University Paper Series on Quantitative Applications in the Social Sciences, Series no. 07-124). Thousand Oaks, CA: SAGE, 1999.
7. Naeem W, Xu T, Sutton R, et al. The design of a navigation, guidance, and control system for an unmanned surface vehicle for environmental monitoring. *Proc IMechE, Part M: J Engineering for the Maritime Environment* 2008; 222(2): 67–80.
8. Motwani A. *A survey of uninhabited surface vehicles*. MIDAS technical report: MIDAS.SMSE.2012; TR.001, 22 April 2012. Plymouth: Plymouth University.
9. Chui CK and Chen G. *Kalman filtering with real-time applications*. 3rd ed. Berlin, Heidelberg: Springer-Verlag, 2008.
10. Weng Z, Chen G, Shieh LS, et al. Evolutionary programming Kalman filter. *Inform Sciences* 2000; 129(1–4): 197–210.
11. Reece S. *Interval arithmetic Kalman filtering*. Oxford: Department of Engineering Science, University of Oxford, 2000.
12. Ahn H, Kim Y and Chen Y. An interval Kalman filtering with minimal conservatism. *Appl Math Comput* 2012; 218(18): 9563–9570.

Appendix I

Consider the following linear interval stochastic-deterministic system

$$\mathbf{x}^I(k+1) = \mathbf{A}^I \mathbf{x}^I(k) + \mathbf{B}^I \mathbf{u}(k) + \boldsymbol{\omega}(k) \quad (18)$$

$$\mathbf{y}^I(k) = \mathbf{C}^I \mathbf{x}^I(k) + \boldsymbol{\nu}(k) \quad (19)$$

where the components of \mathbf{A}^I , \mathbf{B}^I , \mathbf{C}^I , the state vector $\mathbf{x}^I(k)$ and the output vector $\mathbf{y}^I(k)$ are interval values; $\mathbf{u}(k)$ is the (deterministic) system input and $\boldsymbol{\omega}(k)$ and $\boldsymbol{\nu}(k)$ are white noise sequences with zero-mean Gaussian distributions with known covariances $\text{cov}(\boldsymbol{\omega}) = \mathbf{Q}$, $\text{cov}(\boldsymbol{\nu}) = \mathbf{R}$, and $\mathbf{E}[\boldsymbol{\omega}(l)\boldsymbol{\nu}^T(k)] = 0 \forall l, k$, $\mathbf{E}[\mathbf{x}^I(0)\boldsymbol{\omega}^T(k)] = 0$, $\mathbf{E}[\mathbf{x}^I(0)\boldsymbol{\nu}^T(k)] = 0 \forall k$.

Then, given successive measurements of the output, the IKF equations (equations (20)–(24)) provide an interval enclosure of the statistically optimal (unbiased and minimum error variance) estimates of the system state vector, for every point-valued system contained in the interval model. The state estimate at each time step is obtained from the previous estimate and the new observed measurement, $\mathbf{y}(k)$, assuming initial estimates for $\mathbf{x}^I(0)$ and the error covariance matrix $\mathbf{P}^{I+}(0)$. Note that the measurement vector is a realisation of the uncertain interval vector $\mathbf{y}^I(k)$ and is an ordinary vector (with point-valued elements).

Prediction

$$\hat{x}^{I+}(k+1) = \mathbf{A}^I \hat{x}^{I+}(k) + \mathbf{B}^I \mathbf{u}(k) \quad (20)$$

$$\mathbf{P}^{I+}(k+1) = \mathbf{A}^I \mathbf{P}^{I+}(k) \mathbf{A}^{I\top} + \mathbf{Q} \quad (21)$$

Kalman gain

$$\mathbf{K}^I(k) = \mathbf{P}^{I-}(k) \mathbf{C}^{I\top} \left\{ \mathbf{C}^I \mathbf{P}^{I-}(k) \mathbf{C}^{I\top} + \mathbf{R} \right\}^{-1} \quad (22)$$

Correction

$$\hat{x}^{I+}(k) = \hat{x}^{I-}(k) + \mathbf{K}^I(k) [y(k) - \mathbf{C}^I \hat{x}^{I-}(k)] \quad (23)$$

$$\mathbf{P}^{I+}(k) = [\mathbf{I} - \mathbf{K}^I(k) \mathbf{C}^I] \mathbf{P}^{I-}(k) \quad (24)$$

and the output estimate at time k is simply

$$\mathbf{y}^{I^{KF}}(k) = \mathbf{C}^I \hat{x}^{I+}(k) \quad (25)$$

When the elements of equations (18) and (19) are all point-valued so that \mathbf{A}^I , \mathbf{B}^I , \mathbf{C}^I , \mathbf{x}^I and \mathbf{y}^I can be replaced by \mathbf{A} , \mathbf{B} , \mathbf{C} , \mathbf{x} and \mathbf{y} , respectively, then the recursive equations (20)–(25) describe the ordinary KF algorithm by replacing \mathbf{A}^I , \mathbf{B}^I , \mathbf{C}^I , \mathbf{K}^I , \hat{x}^{I-} , \hat{x}^{I+} , \mathbf{P}^{I-} , \mathbf{P}^{I+} and $\mathbf{y}^{I^{KF}}$ with \mathbf{A} , \mathbf{B} , \mathbf{C} , \mathbf{K} , \hat{x}^- , \hat{x}^+ , \mathbf{P}^- , \mathbf{P}^+ and \mathbf{y}^{KF} , respectively.

Appendix 2

The RMLP of Figure 3 has a hidden layer with five units, all of which incorporate hyperbolic tangent activation functions. Information is propagated forward through the network at each time step according to equation (26).

Forward propagation

$$\mathbf{a}^{(1)} = \begin{bmatrix} 1 \\ \mathbf{x} \end{bmatrix}; \mathbf{a}^{(2)} = \begin{bmatrix} 1 \\ \tanh((\Theta^{(1)} \mathbf{a}^{(1)})) \end{bmatrix}; \hat{w} = \mathbf{a}^{(3)} = \Theta^{(2)} \mathbf{a}^{(2)} \quad (26)$$

where $\mathbf{a}^{(l)}$ are the outputs of the nodes of layer (l), and $\Theta^{(1)} \in \mathbb{R}^{5 \times 13}$ and $\Theta^{(2)} \in \mathbb{R}^{1 \times 6}$ are the matrices of parameters of the network such that $\Theta_{ij}^{(l)}$ represents the strength of the connection between node $\mathbf{a}_j^{(l)}$ and $\mathbf{a}_i^{(l+1)}$ (Figure 3).

Training the network consists of finding the parameters $\Theta_{ij}^{(l)}$ that minimise the cost function

$$J = \frac{1}{m-5} \sum_{k=6}^m \frac{1}{2} (w_t(k) - \hat{w}(k))^2 \quad (27)$$

m being the number of training samples. This process was carried out recursively via the gradient descent (GD) algorithm: after assigning random initial values to $\Theta_{ij}^{(l)}$, the parameters are updated as

$$\Theta_{ij}^{(l)} := \Theta_{ij}^{(l)} - \alpha \frac{\partial J}{\partial \Theta_{ij}^{(l)}} \text{ for all } \Theta_{ij}^{(l)} \quad (28)$$

until convergence is reached, where α is the learning rate, chosen adequately based on trial and error. The gradient of the cost function with respect to the network's parameters was computed using the back-propagation (BP) method.⁶

BP. For each training pattern $\mathbf{x}(k)$ (equation (12)) and target $w_t(k)$

1. Compute \hat{w} (equation (26));
2. $\delta^{(3)} = w_t - \hat{w}$;

$$\delta_i^{(2)} = \left[1 - \tanh^2 \left(\sum_j \Theta_{ij}^{(1)} \mathbf{a}_j^{(1)} \right) \right] \Theta_i^{(2)} \delta^{(3)}; \quad i = 1, \dots, 5$$

$$\Delta_i^{(2)} := \Delta_i^{(2)} - \delta^{(3)} \mathbf{a}_i^{(2)}; \quad i = 0, \dots, 5$$

$$\Delta_{ij}^{(1)} := \Delta_{ij}^{(1)} - \delta_i^{(2)} \mathbf{a}_j^{(1)}; \quad i = 1, \dots, 5; j = 0, \dots, 12 \quad (29)$$

end

$$\frac{\partial J}{\partial \Theta_{ij}^{(l)}} = \frac{1}{m} \Delta_{ij}^{(l)} \quad (30)$$

The GD process was applied in two stages, depending on how the gradient was calculated. During the first set of iterations, the (delayed) target values w_t were used to construct $\mathbf{x}(k)$ for computation of \hat{w} in the first step of the BP process, effectively training a network without feedback. During a second stage, (past) predictions of the network \hat{w} were used to construct $\mathbf{x}(k)$ in accordance with the true feedback architecture of the network. Although the gradients computed with the BP algorithm in this case are approximations to the true gradients, the errors are small as after the first set of iterations, the network is sufficiently trained to output predictions close to the target values.