Proceedings of the
44th IEEE Conference on Decision and Control, and
the European Control Conference 2005
Seville, Spain, December 12-15, 2005

GPS Navigation Using Fuzzy Neural Network Aided Adaptive Extended Kalman Filter

Dah-Jing Jwo and Hung-Chih Huang

Abstract—**GPS navigation state processing using the extended Kalman filter provides optimal solutions (in the mean square sense) if the noise statistics for the measurement and system are completely known. Covariance matching method is a conventional adaptive approach for estimation of noise covariance matrices. This innovation-based adaptive estimation shows noisy result if the window size is small. To overcome the problem, the fuzzy method combined with NN to identify the noise covariance matrix is proposed. The structure of FNN can automatically identify the fuzzy rules and tune the membership functions by modifying the connection weights of the network using back-propagation algorithm. Numerical simulations show that the adaptation accuracy based on the proposed approach is substantially improved.**

I. INTRODUCTION

THE Kalman filter [1-2] provides optimal (minimum mean square error) estimate of the system state vector has been widely applied to the fields of GPS receiver position/velocity determination [3], radar target tracking, and integrated navigation system design. In practice, the Kalman filter will provide the optimal result if the complete *a priori* knowledge of the process noise covariance matrix and the measurement noise covariance matrix are available.

Many efforts have been made to improve the estimation of the covariance matrices. Mehra [4] classified the adaptive approaches into four categories: Bayesian, maximum likelihood, correlation and covariance matching. These methods can be applied to the Kalman filtering algorithm for realizing the adaptive Kalman filtering [4-6]. However, the first two methods are computationally demanding so that their practical applications are limited. As for the correlation methods, a set of equations is derived to relate the functions to the unknown parameter. The covariance matching technique attempts to make the filter residuals consistent with their theoretical covariances. Results from such innovation-based adaptive estimation are too noisy if the window size is small. On the other hand, the transient time needed to reach the converged value will increase if the window size is increased.

The application of artificial intelligence to adaptive Kalman filter has been essentially based on the use of fuzzy logic, e.g., [7-10]. Sasiadek, Wang, and Zeremba introduced the Fuzzy Logic Adaptive System for adapting the process and measurement noise covariance matrices for navigation data fusion [7]. Abdelnour, Chand, and Chiu used the exponential-weighting algorithm for detecting and correcting the divergence of the Kalman filter [8]. Kobayashi, Cheok, and Watanabe proposed a method for generating an accurate estimate of the absolute speed of a vehicle from noisy acceleration and erroneous wheel speed information [9]. The method employed the fuzzy logic rule-based Kalman filter to handle abrupt wheel skid and slip, and poor signal-to-noise sensor data. Mostov and Soloviev proposed the method to increase the Kalman filter order, which in turn enhances the accuracy of smoothing and thus location finding for kinematic GPS [10]. Nevertheless, the use of artificial neural networks for adaptive Kalman filter has not been widely seen in the open literature.

Neural networks (NNs) [11] are trainable, dynamic systems that can estimate input-output functions. NNs have been applied to a wide variety of problems. The NN is motivated by their ability to approximate an unknown nonlinear input-output mapping through supervised training. They have been applied to a wide variety of problems since they are model-free estimator, i.e., without a mathematical model. Fuzzy modeling is the method of describing the characteristics of a system using fuzzy inference rules. Takagi and Sugeno proposed a fuzzy modeling approach to model nonlinear systems [12]. The method has a distinguishing feature in that it can express complex nonlinear systems linguistically. However, it is difficult to identify the fuzzy rules and tune the membership functions of the fuzzy reasoning. Identification of the rules takes a lot of times and tuning membership functions of the fuzzy reasoning needs "the fuzzy reasoning of experts". Several researches have been done to combine the learning capability of NN and fuzzy reasoning [13-15]. The scheme is fuzzy neural network (FNN) or neurofuzzy network.

The FNN can be realized as a neural network structure, and the parameters of fuzzy rules can be expressed as the connection weights of the neural network. It is easy to translate the "expert priori knowledge" into the fuzzy if-then rules. The back-propagation (BP) neural network has been the most popular learning algorithm throughout all neural applications. It is simple and requires a minimal amount of storage. BPNN is a neural system with a BP algorithm that can learn input-output functions from a series of samples. It is a gradient-based algorithm, in the sense that the weight update is performed along the direction of the gradient of an appropriate error function.

Manuscript received March 5, 2005. This work has been supported by the National Science Council of the Republic of China through Grant NSC 93-2212-E-019-004 and NSC 94-2212-E-019-003.

The authors are with the Department of Communications and Guidance Engineering, National Taiwan Ocean University, Keelung 20224, TAIWAN (Corresponding author: Dah-Jing Jwo, phone: +886-2-24622192 ext. 7209; fax: +886-2-24633492; e-mail: djjwo@ mail.ntou.edu.tw).

Utilizing the FNN approach to aid the Kalman filter for estimating the time varying variances results in better performance than covariance matching method does. The noise covariance is a complicated mapping with the innovation. The innovation produced by the Kalman filter is used as inputs of the fuzzy neural network, and the desired outputs are the corresponding noise spectral strength. The FNN is then trained off-line using the steepest descent (SD) technique to minimize the differences between the outputs of FNN and the desired outputs. Consequently, the estimation accuracy of the noise parameters can be substantially improved when the FNN is utilized to correctly estimate the noise covariance matrices in the adaptive Kalman filter mechanism.

This paper is organized as follows. In Section II, the conventional adaptive extended Kalman filter (AEKF) approaches are introduced. In Section III, the proposed FNN aided AEKF algorithm is presented and in Section IV, the results by applying FNN aided AEKF to GPS navigation solution is presented. The conclusion is given in Section V.

II. COVENTIONAL ADPTIVE EXTENDED KALMAN FILTER (AEKF)

The process model and measurement model of the Kalman filter are represented as

$$\text{Process model: } \dot{\mathbf{x}} = \mathbf{F}\mathbf{x} + \mathbf{G}\mathbf{u} \tag{1a}$$

$$\text{Measurement model: } \mathbf{z} = \mathbf{H}\mathbf{x} + \mathbf{v} \tag{1b}$$

where the vectors $\mathbf{u}(t)$ and $\mathbf{v}(t)$ are white noise sequences both with zero means and mutually independent:

$$E[\mathbf{u}(t)\mathbf{u}^{\mathrm{T}}(\tau)] = \mathbf{Q}\delta(t-\tau) \tag{2a}$$

$$E[\mathbf{v}(t)\mathbf{v}^{\mathrm{T}}(\tau)] = \mathbf{R}\delta(t-\tau) \tag{2b}$$

$$E[\mathbf{u}(t)\mathbf{v}^{\mathrm{T}}(\tau)] = \mathbf{0} \tag{2c}$$

where $\delta(t)$ is the Dirac delta function. Expressing (1a) and (1b) in discrete-time equivalent form leads to

$$\mathbf{x}_{k+1} = \mathbf{\Phi}_{\mathrm{k}}\mathbf{x}_{\mathrm{k}} + \mathbf{w}_k \tag{3a}$$

$$\mathbf{z}_k = \mathbf{H}\mathbf{x}_k + \mathbf{v}_k \tag{3b}$$

where the vectors $\mathbf{w}_k$ and $\mathbf{v}_k$ are both zero mean white sequences having zero crosscorrelation with each other:

$$\mathbf{E}[\mathbf{w}_k\mathbf{w}_i^{\mathrm{T}}] = \begin{cases} \mathbf{Q}_k, & i = k \\ 0, & i \neq k \end{cases} \tag{4a}$$

$$\mathbf{E}[\mathbf{v}_k\mathbf{v}_i^{\mathrm{T}}] = \begin{cases} \mathbf{R}_k, & i = k \\ 0, & i \neq k \end{cases} \tag{4b}$$

$$\mathbf{E}[\mathbf{w}_k\mathbf{v}_i^{\mathrm{T}}] = \mathbf{0} \tag{4c}$$

The flow chart for the GPS navigation solutions using Kalman filter approach is shown inside the right-hand-side block of Fig. 1. Further discussion can be referred to [1-2] for the KF, and [3] for the GPS receiver position/velocity determination.

The implementation of Kalman filter requires the *a priori* statistical knowledge of the process noise and measurement noise. Poor knowledge of the noise statistics may seriously degrade the Kalman filter performance, and even provoke the filter divergence. To fulfil the requirement, an adaptive Kalman filter can be utilized as the noise-adaptive filter to estimate the noise covariance matrices.

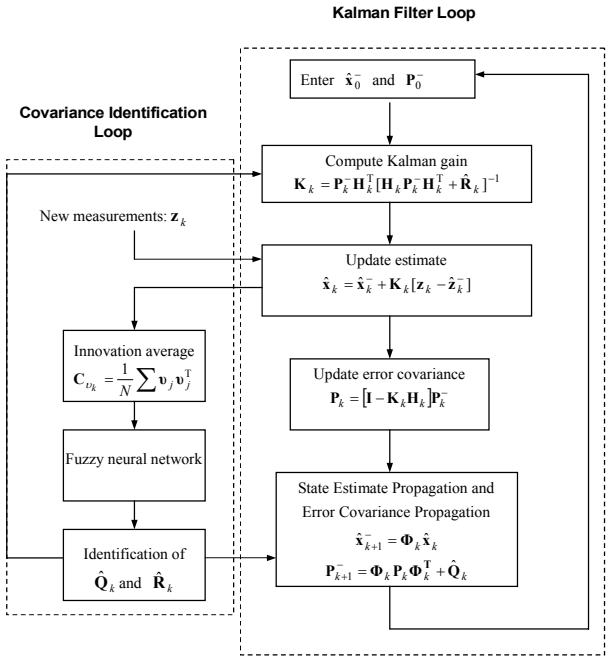

Fig. 1. Flow chart for the fuzzy neural network aided adaptive Kalman filter.

Mehra [4] classified the adaptive approaches into four categories: Bayesian, maximum likelihood, correlation and covariance matching. The innovation sequences have been utilized by the correlation and covariance-matching techniques to estimate the noise covariances. The basic idea behind the covariance-matching approach is to make the actual value of the covariance of the residual consistent with its theoretical value. From the incoming measurement $\mathbf{z}_k$ and the optimal prediction $\hat{\mathbf{x}}_k^-$ obtained in the previous step, the innovations sequence is defined as

$$\upsilon_k = \mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}_k^- \tag{5}$$

The innovation represents the additional information available to the filter as a consequence of the new observation $\mathbf{z}_k$. The weighted innovation, $\mathbf{K}_k(\mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}_k^-)$, acts as a correction to the predicted estimate $\hat{\mathbf{x}}_k^-$ to form the estimation $\hat{\mathbf{x}}_k$. Substituting the measurement model (3b) into (5) gives

$$\upsilon_k = \mathbf{H}_k(\mathbf{x}_k - \hat{\mathbf{x}}_k^-) + \mathbf{v}_k \tag{6}$$

which is a zero-mean Gaussian white noise sequence. By taking variances on both sides, we have the theoretical covariance

$$\mathbf{C}_{\upsilon_k} = \mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^{\mathrm{T}} + \mathbf{R}_k \tag{7}$$

This leads to an estimate of $\mathbf{R}_k$:

$$\hat{\mathbf{R}}_k = \hat{\mathbf{C}}_{\upsilon_k} - \mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^{\mathrm{T}} \tag{8}$$

where $\hat{\mathbf{C}}_{\upsilon_k}$ is the statistical sample variance estimate of $\mathbf{C}_{\upsilon_k}$. Matrix $\hat{\mathbf{C}}_{\upsilon_k}$ can be computed through averaging inside a moving estimation window of size N

$$\hat{\mathbf{C}}_{\upsilon_k} = \frac{1}{N}\sum_{j=j_0}^{k}\mathbf{v}_j\mathbf{v}_j^{\mathrm{T}} \tag{9}$$

where $j_0 = k - N + 1$ is the first sample inside the estimation window. Based on the residual based estimate, the estimate of process noise $\mathbf{Q}_k$ is obtained:

$$\hat{\mathbf{Q}}_k = \frac{1}{N}\sum_{j=j_0}^{k}\Delta\mathbf{x}_j\Delta\mathbf{x}_j^{\mathrm{T}} + \mathbf{P}_k - \mathbf{\Phi}_k\mathbf{P}_{k-1}\mathbf{\Phi}_k^{\mathrm{T}} \tag{10}$$

where $\Delta\mathbf{x}_k = \hat{\mathbf{x}}_k - \hat{\mathbf{x}}_k^-$. This equation can be written in terms of the innovation sequence:

$$\hat{\mathbf{Q}}_k \approx \mathbf{K}_k\hat{\mathbf{C}}_{\upsilon_k}\mathbf{K}_k^{\mathrm{T}} \tag{11}$$

For further information of these equations, see Mohamed & Schwarz [6]. If the window size *N* is too small, the estimation of measurement covariance will be too noisy. On the other hand, if a large window size is utilized, the estimation of measurement covariance will be smoother, however, at the expense of long transient time. The window size is chosen empirically to give some statistical smoothing. In some practical applications, there are chances in which the noise spectral amplitudes rapidly change, then the conventional approach will not suffice the adaptation requirement.

III. THE PROPOSE FUZZY NEURAL NETWORK AIDED AEKF SCHEME

The NN is a network structure consisting of a number of nodes connected through directional links. Each node represents a process unit, and the links between nodes specify the casual relationship between the connected nodes. The learning rule specifies how these parameters should be updated to minimize a prescribed error measure, which is a mathematical expression that measures the discrepancy between the network's actual output and a desired output. The FNN can be realized as a NN structure, and the parameters of fuzzy rules can be expressed as the connection weights of the NN. It is easy to translate the "expert priori knowledge" into the fuzzy if-then rules. The new scheme is based on the use of the BP algorithm, and it can acquire the fuzzy inference rules and tune the membership functions simultaneously through learning from experts' inference data.

A. Structure of the Fuzzy Neural Network

The FNN to be employed in this paper is adopted from [13], in which it was applied to a controller for robotic gait synthesis. Fig. 2 shows topology of the FNN. In the network structure, the fuzzy rules and membership functions are expressed by the connection weights of a neural network. The FNN employed in the present work is made of five layers. There are $p+1$ innovations from the Kalman filter as the inputs, one output, which represents the noise covariance strength, in the output layer and two membership functions in each premise. The premise part stands for the condition of the fuzzy rules and the consequence part stands for the result of the fuzzy rules.

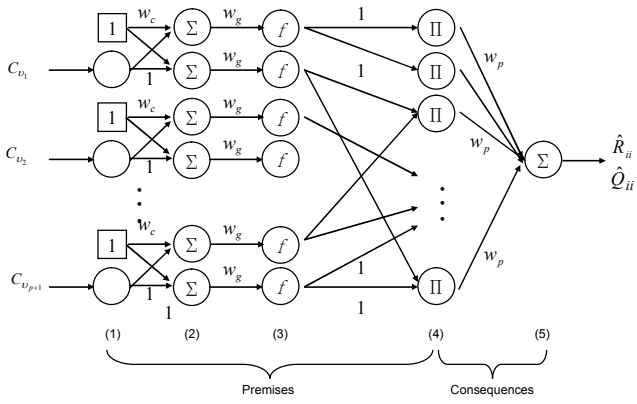

Fig .2. The topology of a fuzzy neural network.

In first layer, the innovations and bias were inputted to the network directly and the weights were set as 1 and w_c, respectively. The output from the first layer is

$$O_i^1 = C_{\upsilon_i} \tag{12}$$

where i represents the number of inputs, and 1 is the output in the first layer. The symbol Σ is the summation and the output in the second layer is

$$O_i^2 = C_{\upsilon_i} + w_c \tag{13}$$

In the third layer, the inputs are the values by multiplying the outputs from the second layer by the weights w_g. The function $f(x)$ employed in the current work is

$$f(x) = \frac{1}{1+e^{-x}} \tag{14}$$

and therefore the output of the third layer is

$$O_i^3(C_{\upsilon_i}) = \frac{1}{1+\exp[-w_g O_i^2]} \tag{15}$$

The membership function in the premises A_{mn} ($m = 1,2; n = 1,2,\cdots,p+1$) can be allocated to the universe of discourse by appropriately initializing the weights w_c and w_g. The membership functions are shown as in Fig. 3. The symbol Π in the fourth layer represents:

$$Input: \mu_k = \prod_n A_{mn} \tag{16}$$

$$Output: O_i^4 = \hat{\mu}_k = \frac{\mu_k}{\sum\mu_k} \tag{17}$$

where A_{mn} is the fuzzy variables in the premises, μ_k is the true value of the *k*th fuzzy rule and $\hat{\mu}_k$ is the normalized value of μ_k. The number of neurons in the fourth layer is 2^i, and these neurons represent the amount of fuzzy rules. Moreover, $\sum\mu_k$ represents the summation of μ_k from different O_i^3.

In the consequence part, the connection weights w_p represent control rules. The center of gravity defuzzifier (COG) is used to infer the output of network. The final inferred value is obtained from the sum of $w_p.\hat{\mu}_k$ and the network realizes the following fuzzy rules:

R^k : If C_{υ_1} is A_{m1}, C_{υ_2} is A_{m2}, ⋯, $C_{\upsilon_{p+1}}$ is $A_{m(p+1)}$

then O_i^5 is B_k

The above fuzzy rule is equivalent to

$$O_i^5 = \sum_{k=1}^{R_f} \hat{\mu}_k B_k \tag{18}$$

where R^k is kth fuzzy rule, B_k is a constant and R_f is the number of rules.

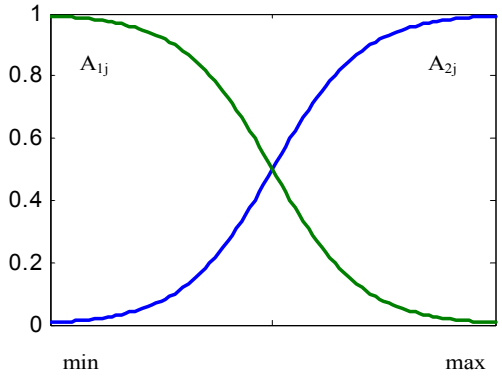

Fig. 3. Membership function in premise.

B. Back-Propagation Algorithm for Training the FNN

The back-propagation learning algorithm is employed to train the connection weights. The adjustments to the connection weights, w_c, w_g and w_p, are computed for the purpose of minimizing a cost function defined as the sum of squared errors:

$$E(n) = \frac{1}{2}\sum [d(n) - y(n)]^2 \tag{19}$$

where $E(n)$ is the total error energy value in n iteration, $d(n)$ is the desired output and $y(n)$ is the output of FNN. Let M denote the total number of patterns contained in the training set, the *average percentage error* is defined as

$$Eav = \frac{1}{M}\sum_{j=1}^{M}\left(\frac{d_j - y_j}{d_j}\right)^2 \tag{20}$$

The weights w_c, w_g and w_p are modified to identify fuzzy rules and tune the membership functions in the premises. The adjustment of weights is implemented by

$$w_{ji}^N(n+1) = w_{ji}^N(n) + \Delta w_{ji}^N(n) \tag{21}$$

where

$$\begin{aligned}\Delta w_{ji}^N(n) &= \eta \cdot [d(n) - y(n)] \cdot f_j'(n) \cdot O_i^{N-1}(n) \\ &= \eta \cdot \delta_j^N(n) \cdot O_i^{N-1}(n)\end{aligned} \tag{22}$$

where η is a learning rate , f' denotes the derivative of f with respect to the weights, δ_j^N is local gradient of the j unit in N th layer and O_i^{N-1} is the output of the i unit in $(N-1)$ th layer. In the backward pass, the local gradient in the fifth layer is

$$\delta^5 = [d(n) - y(n)]\, f'(I_i^5) = d(n) - y(n) \tag{23}$$

and the weights w_p can be update via

$$w_p(n+1) = w_p(n) + \eta \cdot \delta^5 \cdot O_i^4 \tag{24}$$

In the above equations, O_i^4 in (24) is actually equal to I_i^5 in (23), due to the fact that the output from the fourth layer is the input to the fifth layer. In the fourth layer, the weights are set as 1 so that the local gradient is calculated as

$$\delta_j^4 = f'(I_j^4)\sum_k \delta_k^5 w_{kj}^5 = \frac{1}{\sum \mu_k} \cdot \delta^5 \cdot w_p \tag{25}$$

The connection weight $w_{kj}^5 = w_p$, which represents the j th unit in the fourth layer to k th unit in the fifth layer. Since the multiplication is done in the fourth layer, the local gradient in the third layer is

$$\begin{aligned}\delta_j^3 &= f'(I_j^3)\sum_k \delta_k^4 \left(\prod_i w_{ki}^4 \prod_{i\neq j} O_i^3\right) \\ &= O_j^3 \cdot (1 - O_j^3) \cdot \sum_k \left(\delta^4 \cdot \prod_{i\neq j} O_i^3\right)\end{aligned} \tag{26}$$

then the weights w_g can be updated by

$$w_g(n+1) = w_g(n) + \eta \cdot \delta^3 \cdot O_i^2 \tag{27}$$

In the second layer, the weights w_c need to be updated and the local gradient is

$$\delta_j^2 = f'(I_j^2)\sum_k \delta_k^3 w_{kj}^3 = \delta_j^3 \cdot w_g \tag{28}$$

The connection weight $w_{kj}^3 = w_g$, and the weights w_c can be updated by

$$w_c(n+1) = w_c(n) + \eta \cdot \delta^2 \cdot O_i^1 \tag{29}$$

Off-line training of network is conducted using the SD technique to minimize the differences between the outputs of FNN and the desired outputs. During the training phase, the innovation $\mathbf{C}_{\upsilon_k}$ produced by the Kalman filter is employed as the input to the FNN. Referring to Fig. 2, the inputs of neural network are the innovations from the present instant t to time $(t-p)$, i.e., $C_{\upsilon_1} \equiv C_\upsilon(t)$, $C_{\upsilon_2} \equiv C_\upsilon(t-1)$,… $C_{\upsilon_{p+1}} \equiv C_\upsilon(t-p)$. The network output vectors ideally describe the actual noise strength. It should be noted that the innovations need to be normalized as inputs and the weights w_c and w_g are initialized so that the membership functions in the premises can be allocated to the normalized universe of discourse.

At the time of recall, when the AEKF receives the measurement $\mathbf{z}_k$, it provides the estimations of the state vector and the $\mathbf{z}_k - \hat{\mathbf{z}}_k^-$ to calculate the innovation. When the input nodes receive the innovation, the appropriate covariance $\hat{\mathbf{R}}_k$ is obtained. Thus, the Kalman filter is provided with the adaptive capability for estimation by combining the filter and the FNN. A flow chart of the FNN aided AEKF is presented in Fig. 1, in which there are two main blocks in dash lines. The left-hand-side block represents the covariance identification loop, while the right-hand-side block is the standard Kalman filter loop.

As for the selection of training data, the FNN designer

should have a rough idea on the ranges of noise strengths, $\mathbf{Q}_k$ and $\mathbf{R}_k$, for certain environment where the navigation will be performed. The proposed algorithm may not be a good choice when the designer has not any knowledge about the ranges of the noise strengths. Although the FNN still demonstrate its adaptation capability, however, the performance will be degraded if range of the training data does not cover all possible range (in the testing phase). Wider range of training data results in larger training time consumption for obtaining a well-trained network.

IV. APPLYING FNN AIDED AKEF TO GPS NAVIGATION SOLUTION

In this section, FNN aided AEKF to GPS navigation application will be conducted. For simplicity, yet, without loss of generality, a simple double-integrator model is first employed to test the adaptive capability. Application to the GPS navigation problem will then be performed. Simulation was conducted using a personal computer with Pentium 4 1.7 GHz CPU. The computer codes were constructed using the Matlab® 6.0 version software.

A. Algorithm Validation

Consider the continuous-time model governed by (1a) and (1b) where

$$\mathbf{F}=\begin{bmatrix}0 & 1\\ 0 & 0\end{bmatrix};\ \mathbf{G}=\begin{bmatrix}0\\ 1\end{bmatrix};\ \mathbf{H}=\begin{bmatrix}1 & 0\end{bmatrix}$$

and these signals satisfy the following

$$E[\mathbf{u}(t)\mathbf{u}^{\mathrm{T}}(\tau)]=q\delta(t-\tau) \tag{30a}$$

$$E[\mathbf{v}(t)\mathbf{v}^{\mathrm{T}}(\tau)]=r\delta(t-\tau) \tag{30b}$$

$$E[\mathbf{u}(t)\mathbf{v}^{\mathrm{T}}(\tau)]=0 \tag{30c}$$

In this example, q value is assumed *a priori* known and the measurement noise variance r is to be identified. For testing the identification capability, the r value is originally set as $1\,m^2$ and suddenly increases to $5\,m^2$. Based on the selection of $p=6$, the numbers of neurons employed in the present work is as follows: 7 neurons in the first (i.e., input) layer; 14 neurons in both the second and third layers; $2^7=128$ neurons in the fourth layer; 1 neuron in the fifth layer. The following parameter values are used: $bias=1$; three learning rates $\eta=0.5,0.7,0.9$, respectively in the second, third and fifth layers; three momentum coefficients $\alpha=0.1,0.3,0.5$, respectively in the second, third and fifth layers. The parameters are tuned based on our experience. There may be other sets of values that will provide better adaptation capability. However, if one chooses the values deviating from the present values within some range (for example, about ±0.2 for η's and about $\pm50\%$ for α's, in the present case), the change of performance is not seen significant. A comparison on the identification results using different approaches for the example is provided in Fig. 4. It is seen that substantial improvement on noise identification capability has been achieved by using the proposed approach (by which the variance is $0.16\ m^2$) as compared to the conventional covariance-matching method (by which the variance is $0.72\ m^2$).

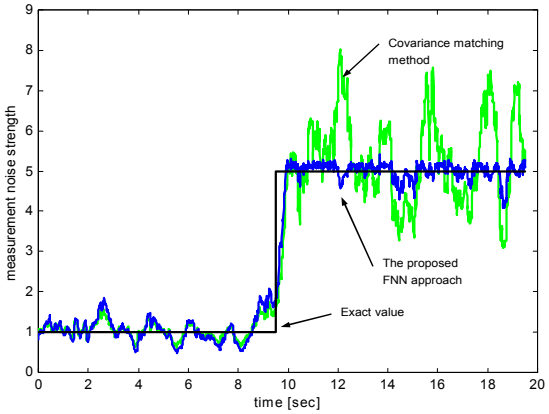

Fig. 4. Comparison of measurement noise (r) identification results using different approaches for the double integrator example.

B. Application to GPS Navigation

When selecting Kalman filtering as the navigation state estimator in the GPS receiver [2-3], using b and d to represent the GPS receiver clock bias and drift, the differential equation for the clock error is written as

$$\begin{aligned}\dot{b}&=d+u_b\\ \dot{d}&=u_d\end{aligned} \tag{31}$$

where $u_b\sim N(0,S_f)$ and $u_d\sim N(0,S_g)$ are independent Gaussianly distributed white sequences. The dynamic process of the GPS receiver in medium dynamic environment can be represented by the PV (Position-Velocity) model [2]. Let each of the white-noise spectral amplitudes that drive the random walk position states be $S_p=1.0(m/\sec)^2/rad/\sec$. Also, let the clock model spectral amplitudes be $S_f=0.4(10^{-18})\sec$ and $S_g=1.58(10^{-18})\sec^{-1}$. If only the pseudorange observables are available, the measurement equation based on n observables leads to

$$\underbrace{\begin{bmatrix}\rho_1\\ \rho_2\\ \vdots\\ \rho_n\end{bmatrix}-\begin{bmatrix}\hat{\rho}_1\\ \hat{\rho}_2\\ \vdots\\ \hat{\rho}_n\end{bmatrix}}_{\mathbf{z_k}}=\underbrace{\begin{bmatrix}h_x^{(1)} & 0 & h_y^{(1)} & 0 & h_z^{(1)} & 0 & 1 & 0\\ h_x^{(2)} & 0 & h_y^{(2)} & 0 & h_z^{(2)} & 0 & 1 & 0\\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots\\ h_x^{(n)} & 0 & h_y^{(n)} & 0 & h_z^{(n)} & 0 & 1 & 0\end{bmatrix}}_{\mathbf{H_k}}\begin{bmatrix}x_1\\ x_2\\ x_3\\ x_4\\ x_5\\ x_6\\ x_7\\ x_8\end{bmatrix}+\begin{bmatrix}v_{\rho_1}\\ v_{\rho_2}\\ \vdots\\ v_{\rho_n}\end{bmatrix} \tag{32}$$

Assuming measurement errors among satellites are uncorrelated, we have

$$\mathbf{R}_k=\begin{bmatrix}r_{\rho_1} & & & \mathbf{0}\\ & r_{\rho_2} & & \\ & & \ddots & \\ \mathbf{0} & & & r_{\rho_n}\end{bmatrix} \tag{33}$$

Fig. 5 illustrates the system architecture for performing GPS navigation solutions using FNN aided AEKF. The scenario for simulation is as follow. The kinematics of the user is assumed to move at a constant velocity with mean value

$10m/s$ to East and $10\sqrt{3}m/s$ to North (which results in a speed of $20m/s$), starting from the position of North 25.15 degrees, East 121.78 degrees. Assuming that the differential GPS mode is used and most of the errors can be corrected, but the multipath and receiver measurement thermal noise cannot be eliminated. The initial measurement variance for each of the pseudorange observables is assumed to be $r_{\rho_i} = 9m^2$ (which is the sum of the variances of the code multipath and measurement noise), for $i = 1 \cdots n$. After a while, the measurement noise standard deviations are raised by ten times of the original one. Again, the FNN employed in the present case has the parameter values same as those used in the double integrator model. Based on the parameter values and scenario presented above, the simulation is conducted. A comparison of GPS positioning errors based on three cases: (1) without adaptation; (2) with adaptation by proposed approach; (3) perfect adaptation, are presented in Fig. 6. For clarity, the corresponding error statistics are summarized in Table 1. Substantial accuracy improvement is achieved by using the proposed adaptive technique.

V. Conclusion

This paper has presented a fuzzy neural network aided adaptive extended Kalman filtering approach for GPS navigation. The fuzzy neural network was employed as a noise identification mechanism to implement the on-line identification of noise covariance matrices. Based on the proposed approach, the noise adaptive capability can be significantly improved as compared to the conventional innovation-based algorithms. The algorithm presented in this paper can be applied to all the Kalman filtering related applications, such as GPS/INS integration, radar target tracking, and so forth.

References

[1] A. Gelb, *Applied optimal estimation*, M. I. T. Press, MA, 1974.

[2] R. G. Brown, P. Y. C. Hwang, *Introduction to random signals and applied Kalman filtering*, John Wiley & Sons, New York, 3rd edn. 1997.

[3] P. Axelrad, R. G. Brown, "GPS navigation algorithms," in *Global Positioning System: Theory and Applications*, B. W. Parkinson, J. J. Spilker, P. Axelrad, and P. Enga, Ed., Volume I, AIAA, Washington DC, Chap. 9, 1996.

[4] R. K. Mehra, "Approaches to adaptive filtering," *IEEE Trans. Automat. Contr*. AC-17, 1972, pp. 693-698.

[5] R. K. Mehra, "On-line identification of linear dynamic systems with applications to Kalman filtering," *IEEE Trans. Automat. Contr*. AC-16, 1970, pp. 12-21.

[6] A.H. Mohamed, K. P. Schwarz, "Adaptive Kalman filtering for INS/GPS," *Journal of Geodesy*, vol. 73, 1999, pp. 193-203.

[7] J. Z. Sasiadek, Q. Wang, M. B. Zeremba, "Fuzzy adaptive Kalman filtering for INS/GPS data fusion," in *Proc. 15th IEEE int. Symp. on intelligent control*, Rio, Patras, Greece, 2000, pp.181-186.

[8] G. Abdelnour, S. Chand, S. Chiu, "Applying fuzzy logic to the Kalman filter divergence problem," in *Proc. IEEE Int. Conf. on Syst., Man and Cybernetics*, Le Touquet, France, 1993, pp. 630-634.

[9] K. Kobayashi, K. Cheok, K. Watanabe, "Estimation of the absolute vehicle speed using fuzzy logic rule-based Kalman filter," in *Proc. American Control Conf.*, Seattle, 1995, pp. 3086-3090.

[10] K. Mostov, A. Soloviev, "Fuzzy adaptive stabilization of higher order Kalman filters in application to precision kinematic GPS," in *Proc. ION GPS-96*, vol. 2, Kansas City, 1996, pp. 1451-1456.

[11] S. Haykin, *Neural networks: a comprehensive foundation*, Prentice-Hall, 1999.

[12] T. Takagi, M. Sugeno, "Fuzzy identification of systems and its application to modelling and control," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-15, 1985, pp. 116-132.

[13] J. G., Juang, "Fuzzy neural network approaches for robotic gait synthesis," *IEEE Trans. on Syst., Man and Cybern.*, Part B: Cybernetics, vol. 30, 2000, pp. 594-601.

[14] S., Horikawa, T. Furuhashi, S. Okuma and Y. Uchikawa, "Composition methods of fuzzy neural networks," *Proc. IEEE Int. Conf. Industrial Electronics, Control and Instrumentation*, 1990, pp. 1253-1258.

[15] S., Horikawa, T. Furuhashi, Y. Uchikawa, "On fuzzy modeling using fuzzy neural network with the back-propagation algorithm," *IEEE Trans. Neural Network*, vol. 3, 1992, pp. 801-806.

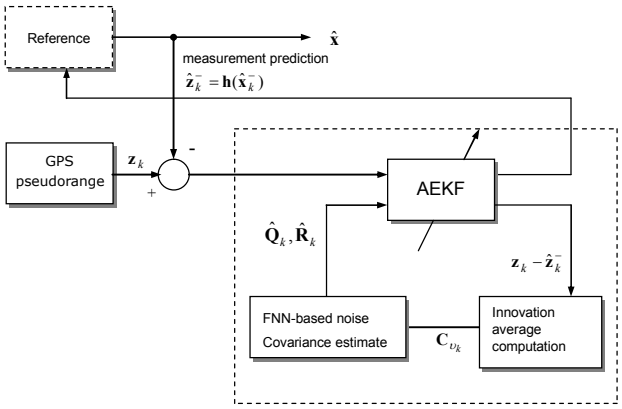

Fig. 5. GPS navigation solutions using fuzzy neural network aided adaptive extended Kalman filter.

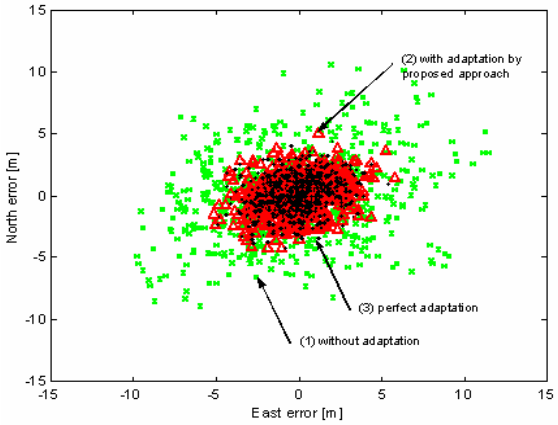

Fig. 6. Comparison of three GPS positioning errors: (1) without adaptation (x); (2) with adaptation by proposed approach (Δ); (3) perfect adaptation (•).

TABLE I

VARIANCES OF THE POSITIONING ERRORS (UNIT: m^2)

	Without adaptation	With adaptation	Perfect adaptation
East	17.5	3.2	2.8
North	12.1	2.3	2.1