

Neural identifier for unknown discrete-time nonlinear delayed systems

Alma Y. Alanis¹ · Jorge D. Rios¹ · Nancy Arana-Daniel¹ · Carlos Lopez-Franco¹

Received: 28 November 2014 / Accepted: 9 August 2015 / Published online: 29 August 2015
© The Natural Computing Applications Forum 2015

Abstract This work proposes a discrete-time nonlinear neural identifier based on a recurrent high-order neural network trained with an extended Kalman filter-based algorithm for discrete-time deterministic multiple-input multiple-output systems with unknown dynamics and time-delay. To prove the semi-globally uniformly ultimately boundedness of the proposed neural identifier, the stability analysis based on the Lyapunov approach is included. Applicability of the proposed identifier is shown via simulation and experimental results, all of them performed under the presence of unknown external and internal disturbances as well as unknown time-delays.

Keywords Recurrent high-order neural networks · Neural identification · MIMO systems · Unknown time-delay · Lyapunov · Discrete-time nonlinear systems

1 Introduction

The study of time-delay systems (TDS) has become an important field of research due to its frequent presence in engineering applications [4, 7, 14, 33], and some examples of this kind of systems are: chemical processes, engine cooling systems, hydraulic systems, irrigation channels, metallurgical processing systems, network control systems and supply networks [4, 14, 33].

Delays in systems happen due to the limited capabilities of their components of processing data and transporting

information and materials [18]. Therefore, the main sources of time-delay in systems are [19]:

- *Nature of the process* which arises when the system has to wait a process in order to continue to the next step, for example chemical reactors, diesel engines and recycled processes.
- *Transport delay* which occurs when systems must transport materials and the controller takes time to affect the process, for example rolling mills, cooling and heating systems.
- *Communication delay* can occur due to:
 - *Propagation time-delay* of signals among actuators, controllers, sensors, especially in networking control and fault-tolerant systems.
 - *Access time-delay* as a result of finite time required to access to a shared media. The data at the controller are a delayed version of the current state, and the control action suffers time-delay when is sent. It is found in network systems.

Delays can be constant or time-varying, known or unknown, deterministic or stochastic depending on the system under consideration [18]. It is well known that delay in systems is a source of instability and poor performance. A number of methodologies have been proposed to handle these problems [7, 14, 20, 21, 33], some of them based on neural networks where neural networks are used to deal with unknown dynamics [5, 7, 20, 31, 32] and most of them developed for continuous-time systems. Besides, loss of packets in network systems can happen as the result of delays which are introduced in the system due to the limited capacity of data transmission between devices [15] which can also result in instability [31, 32].

✉ Alma Y. Alanis
almayalanis@gmail.com

¹ CUCEI, Universidad de Guadalajara, Apartado Postal 51-71, Col. Las Águilas, C.P. 45080 Zapopan, Jalisco, Mexico

In order to control a system, a model is usually required, which is a mathematical structure knowledge about the system represented in the form of differential or difference equations. Some motives for establishing mathematical descriptions of dynamical systems are: simulation, prediction, fault detection and control system design, among them [22].

There are two ways to obtain a system model: It can be derived in a deductive manner using laws of physics, or it can be inferred from a set of data collected during a practical experiment of the system through its operating range [22]. The first method can be simple; however, in many cases, necessary time for getting the system model can be excessive; moreover, obtaining an accurate model this way would be unrealistic because of unknown dynamics and delays that cannot be modeled. The second method, which is known as system identification, could be a helpful shortcut for deriving the model. Despite the fact that system identification does not always result in an accurate model, a satisfactory one can be obtained with reasonable efforts [22].

There are a considerable number of methods to accomplish system identification, to name a few: based on neural network, based on fuzzy logic, auxiliary model identification and hierarchical identification [6]. Neural networks stand out for their characteristics, such as no need to establish the model structure of the actual system, any linear and nonlinear model can be identified, and a neural network is a model and an actual system. Neural networks allow us to identify and obtain mathematical models which are close to the actual system behaviors even in the presence of variations and disturbances [6, 22].

Recurrent neural networks (RNNs) are a special kind of neural networks which have feedback connections that have an impact on the learning capabilities and performance of the network; moreover, unit-delay elements on the feedback loops result in a nonlinear dynamical behavior [8].

Recurrent high-order neural networks (RHONNs) are an extension of first-order Hopfield network. A RHONN has more interaction among its neurons and characteristics which make them ideal for modeling complex nonlinear systems, such as: approximation capabilities, easy implementation, robustness against noise and online training, which make them ideal for modeling complex nonlinear systems [28].

Backpropagation through time learning is the most used training algorithm for RNNs; however, it is a first-order gradient descent method, that even if it gives great results in several occasions, it presents some problems such as slow convergence, high complexity, bifurcations, instability and limited applicability due to high computational costs [10, 28], and it is not able to discover contingencies

spanning long temporal intervals due to the vanishing gradient [3, 12]. On the other hand, training algorithms that are based on the extended Kalman filter (EKF) improve learning convergence, reduce the epoch number and the number of required neurons. Moreover, EKF training methods for training feedforward and RNNs have proven to be reliable and practical [9, 28].

In this work, a RHONN identifier trained with a EKF-based algorithm for discrete-time nonlinear deterministic multiple-input multiple-output (MIMO) systems with unknown time-delay is presented, and simulation results of the proposed RHONN identifier are compared with the results presented in [20] for the same system. In order to prove the semi-globally uniformly ultimately boundedness (SGUUB) of the RHONN identifier trained with a EKF-based algorithm, a Lyapunov stability analysis is included.

It is important to note that the proposed discrete-time RHONN identifier does not require previous knowledge of the system model nor an estimation of the time-delay or system perturbation or their bounds. These characteristics make it ideal for devices which require of a real-time implementation. Most of the work on the field of system identification of nonlinear TDS addressed the continuous-time case [1, 24, 31, 32] and assumed that the time-delay is known, approximated or at least a bound is previously known [2]. In [13, 24], the authors present methodologies for system identification of nonlinear continuous-time-delayed systems in which an approximation of the time-delay is needed. In [24], the authors present an approach where the knowledge of the system model is necessary at least nominally. Then, all the above-explained properties allow the proposed discrete-time identifier, an excellent option for real-time implementations in digital devices.

Therefore, the main contributions of this paper are:

1. A RHONN identifier trained with an EKF-based algorithm for discrete-time deterministic MIMO systems with unknown dynamics and time-delay.
2. Results of simulation and real-time for three time-delay benchmarks. Systems are presented in order to show the performance of the identifier.
3. The stability proof based on the Lyapunov approach for the neural identifier without the need of previous knowledge for plant dynamics, disturbances, delays nor its bounds.

This work is organized as follows: In Sect. 2, time-delay systems are introduced, Sects. 3 and 4 give a brief explanation of RHONNs and EKF-trained algorithm, respectively, in Sect. 5, the RHONN-based neural identifier is described, in Sect. 6, results are shown, important conclusions are presented in Sect. 7, and finally, at the “Appendix”, the Lyapunov analysis is included.

2 Time-delay nonlinear system

Time-delay is the property of physical systems by which the response to an applied force is delayed as well as its effects [36]. Whenever a material, information or energy is transmitted, there is an associated delay and its value is determined by distance and speed of the transmission [36].

TDS, also known as systems with after effect or dead-time, hereditary systems, equations with deviating argument or differential-difference equations, are systems which have a significant time-delay between the application of the inputs and their effects. TDS inherit time-delay in its components or from introduction of delays for control design purposes [25, 30, 34].

In general, TDS can be classified as [18]:

- Systems with lumped delays: where a finite number of parameters can describe their delay phenomena.
- Systems with distributed delays: where it is not possible to find a finite number of parameters which described their time-delay phenomena.

The presence of delays in the systems makes system analysis and control design complex and also can degrade the performance or induce instability [34, 36], this is why it is important to understand delays in systems; otherwise, the system could become unstable [36].

Let us consider the following discrete-time-delay nonlinear MIMO system described by:

$$x(k + 1) = F(x(k - l), u(k)) \tag{1}$$

$$y(k) = h(x(k)) \tag{2}$$

where $x \in \mathfrak{R}^n$, $u \in \mathfrak{R}^m$, $F \in \mathfrak{R}^n \times \mathfrak{R}^m \rightarrow \mathfrak{R}^n$ is a nonlinear function and $l = 1, 2, \dots$ is the unknown delay.

3 Recurrent high-order neural network

Recurrent neural networks are considered as good candidates for identification and control of general nonlinear and complex systems, RNNs deal with uncertainties and modeling error on the systems, and also they are attractive due to their characteristics as easy implementation, simple structure, robustness and capacity to adjust their weights online [23, 28].

RNN outputs flow in forward and backward directions, and they are feedback to the same neurons or neurons in different layers. This two-way connectivity between units distinguished RNN from feedforward neural networks, where the output of one unit is only connected to the units in the next layer [16, 23].

RNNs are mostly based on the Hopfield model [23], and their recurrent structure has a profound impact on their learning capabilities and performance [35].

Recurrent high-order neural networks are the result of including high-order interactions represented by triplets $(y_i y_j y_k)$, quadruplets $(y_i y_j y_k y_l)$, etc. to the first-order Hopfield model [16, 35]. RHONN model is very flexible, and it allows to incorporate *a priori* information about the system into the neural network model [28]

Consider the following discrete-time recurrent High-Order neural network:

$$\hat{x}_i(k + 1) = w_i^\top(k) z_i(x(k - l), u(k)), \quad i = 1, \dots, n \tag{3}$$

where \hat{x}_i ($i = 1, 2, \dots, n$) is the state of the i -th neuron, w_i is the respective online adapted weight vector, n is the state dimension, and $z_i(x(k - l), u(k))$ is given by

$$z_i(x(k - l), u(k)) = \begin{bmatrix} z_{i_1} \\ z_{i_2} \\ \vdots \\ z_{i_{L_i}} \end{bmatrix} = \begin{bmatrix} \prod_{j \in I_1} \zeta_j^{d_{ij}(1)} \\ \prod_{j \in I_2} \zeta_j^{d_{ij}(2)} \\ \vdots \\ \prod_{j \in I_{L_i}} \zeta_j^{d_{ij}(L_i)} \end{bmatrix} \tag{4}$$

with L_i as the respective number of high-order connections, $\{I_1, I_2, \dots, I_{L_i}\}$ is a collection of nonordered subsets of $\{1, 2, \dots, n + m\}$, m is the number of external inputs, $d_{ij}(k)$ being nonnegative integers, and ζ_i defined as follows:

$$\zeta_i = \begin{bmatrix} \zeta_{i_1} \\ \vdots \\ \zeta_{i_n} \\ \zeta_{i_{n+1}} \\ \vdots \\ \zeta_{i_{n+m}} \end{bmatrix} = \begin{bmatrix} S(x_1) \\ \vdots \\ S(x_n) \\ u_1 \\ \vdots \\ u_m \end{bmatrix} \tag{5}$$

In (5), $u = [u_1, u_2, \dots, u_m]^\top$ is the input vector to the neural network, and $S(\cdot)$ is defined by

$$S(\varsigma) = \frac{1}{1 + \exp(-\beta\varsigma)}, \quad \beta > 0 \tag{6}$$

where ς is any real value variable.

4 Kalman filter learning algorithm

There are a number of training algorithms for neural networks; however, most of them have problems such as local minimal, slow learning and high sensitivity to initial conditions. Kalman filter (KF)-based algorithms show up as an alternative which overcome these problems [9, 28].

KF estimates the state of a linear system with additive state and output white noise [28]. KF has a recursive solution; in each update, the state is estimated from the previous estimated state and the new input data. The fact that it is only necessary to store the previous estimated state in memory makes KF computationally more efficient than

computing the estimated state directly from the entire past observed data of previous steps [9, 28].

For KF-based neural network training, the network weights become the states to be estimated, and for this case, the error between the neural network output and the measured plant output is considered as additive noise [9, 28].

Extended Kalman filter is used due to the fact that neural network mapping is a nonlinear problem [28]. EKF is an extension of KF obtained through a linearization procedure [9].

4.1 Extended Kalman filter training algorithm

The EKF-based training algorithm used for training the RHONN identifier on this work is:

$$\begin{aligned} \omega_i(k+1) &= \omega_i(k) + \eta_i K_i(k) e_i(k) \\ K_i(k) &= P_i(k) H_i(k) M_i(k) \\ P_i(k+1) &= P_i(k) - K_i(k) H_i^T(k) P_i(k) + Q_i(k) \end{aligned} \tag{7}$$

$i = 1, \dots, n$

with

$$M_i(k) = [R_i(k) + H_i^T(k) P_i(k) H_i(k)]^{-1} \tag{8}$$

$$e_i(k) = x_i(k) - \hat{x}_i(k) \tag{9}$$

The dynamics of (9) can be expressed as

$$e_i(k+1) = \tilde{w}_i(k) z_i(x(k-l), u(k)) + \epsilon_{z_i} \tag{10}$$

$$H_{ij} = \left[\frac{\partial \chi_i(k)}{\partial \omega_{ij}(k)} \right]^T \tag{11}$$

where $i = 1, \dots, n$, $e_i \in \mathfrak{R}$ is the identification error, $P_i \in \mathfrak{R}^{\Omega_i \times \Omega_i}$ is the weight estimation error covariance matrix, $\omega_i \in \mathfrak{R}^{\Omega_i}$ is the online adapted weight vector, χ_i is the i -th state variable of the neural network, $K_i \in \mathfrak{R}^{\Omega_i}$ is the Kalman gain vector, $Q_i \in \mathfrak{R}^{\Omega_i \times \Omega_i}$ is the estimation noise covariance matrix, $R_i \in \mathfrak{R}$ is the error noise covariance matrix, $H_i \in \mathfrak{R}^{\Omega_i}$ is a vector in which each entry H_{ij} is the derivative of the neural network state (χ_i) with respect to one neural network weight (ω_{ij}), and it is given by (11), where $i = 1, \dots, n$ and $j = 1, \dots, L_i$. P_i and Q_i are initialized as diagonal matrices with entries $P_i(0)$ and $Q_i(0)$, respectively. It is important to remark that $H_i(k)$, $K_i(k)$ and $P_i(k)$ for the EKF are bounded [29].

5 Neural identification

Neural Identification consists in selecting a neural network model, adjusting its weights according an adaptation law in order to approximate the real system response to the same input [22].

5.1 RHONN identifier

Let us consider the problem to approximate the general discrete-time nonlinear delayed systems (1), by the following discrete-time RHONN series–parallel representation:

$$x_i(k+1) = w_i^{*\top} z_i(x(k-l), u(k)) + \epsilon_{z_i}, \quad i = 1, \dots, n \tag{12}$$

ϵ_{z_i} is a bounded approximation error, which can be reduced by increasing the number of the adjustable weights [27]. Assume that there exists a ideal weight vector w_i^* such that $\|\epsilon_{z_i}\|$ can be minimized on a compact set $\Omega_{z_i} \subset \mathfrak{R}^{\Omega_i}$. The ideal weight vector w_i^* is an artificial quantity required for analytical purpose [27]. In general, it is assumed that this vector exists and is constant but unknown. Let us define its estimate as w_i and the estimation error as

$$\tilde{w}_i(k) = w_i^* - w_i(k) \tag{13}$$

then, considering (8), the dynamics of (13) can be defined as

$$\tilde{w}_i(k+1) = \tilde{w}_i(k) - \eta_i K_i(k) e(k) \tag{14}$$

Theorem 1 *The RHONN model (12) trained with the modified EKF-based algorithm (8) to identify the delayed nonlinear plant (1) ensures that the identification error (10) is SGUUB; moreover, the RHONN weights remain bounded.*

Proof Please, see the “Appendix”. □

6 Results

6.1 Example 1. Simulation results

Consider the following nonlinear time-delay system:

$$\begin{aligned} \dot{x}_1(t) &= x_2(t) + 0.001x_1(t)u(t) \\ \dot{x}_2(t) &= (1 - x_1^2(t))x_2(t) - x_1(t) + x_3(t)u(t) + 2\cos(x_1(t-3)) \\ \dot{x}_3(t) &= x_4(t) + 0.01x_2(t)x_3(t)\exp(u(t)) \\ \dot{x}_4(t) &= (1 - x_3^2(t))x_4(t) - x_3(t) + \frac{u(t)}{(1 + x_2^2(t)x_4^2(t))} \\ &\quad + 2(x_1^2(t-3) + x_2^2(t-3))\sin(x_2(t-3)) \\ y_1(t) &= x_1(t) + x_2(t) \\ y_2(t) &= x_3(t) + x_4(t) \end{aligned} \tag{15}$$

System (15) is a chaotic oscillator similar to a van der Pol system [20].

To conduct the simulation tests, the system (15) is simulated using MATLAB^{®1}\Simulink^{®2} 2013a and its states are discretized by a zero-order hold with sampling

¹ MATLAB is a registered trademark of The MathWorks, Inc.

² Simulink is a registered trademark of The MathWorks, Inc.

time equals to 0.2 s and initial conditions $x(0) = [1 \ 1 \ 1 \ 1]^T$.

6.1.1 High-order neural network observer

Let us consider the following high-order neural network (HONN) observer design in [20]:

$$\begin{aligned} \hat{x}(k+1) &= A\hat{x}(k) + B[\hat{W}_1^T(k)\Phi_1(\hat{x}(k), u(k)) \\ &\quad + h(\hat{x}(k - \hat{d}))] + K[y(k) - \hat{y}(k)] + Du(u) \\ \hat{y}(k) &= C\hat{x}(k) \end{aligned} \tag{16}$$

where $\hat{x}(k)$ is the estimation of $x(k)$, h is a known function vector with constant time-delay, \hat{d} is the estimation of d which is the unknown time-delay, \hat{W}_1 is the HONN weight matrix, $\Phi(\hat{x}, u)$ is the basis function vector, and K is the observer gain matrix [20], with

$$\begin{aligned} \hat{W}_1(k+1) &= (1 - \sigma_1)\hat{W}_1(k) \\ &\quad + \Gamma_1\Phi_1(\hat{x}(k), u(k))\tilde{y}^T(k)F^T \end{aligned} \tag{17}$$

Parameter values are set as in [20]: delay functions (18) are known with estimation of the unknown time-delay $d = 3$ s and $\hat{d} = 3.2$ s. $u(t) = \sin(0.3t)$.

$$\begin{aligned} h_1(x) &= 0 \\ h_2(x) &= 2 \cos(x_1(t)) \\ h_3(x) &= 0 \\ h_3(x) &= 2(x_1^2(t) + x_2^2(t)) \sin(x_2(t)) \end{aligned} \tag{18}$$

Matrices A , C , F and K are set as:

$$\begin{aligned} A &= \begin{bmatrix} 1 & 0.2 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0.2 \\ 0 & 0 & 0 & 1 \end{bmatrix}, & F &= \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{bmatrix} \\ C &= \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}, & K &= \begin{bmatrix} 0.3 & 0 \\ 1 & 0 \\ 0 & 0.3 \\ 0 & 1 \end{bmatrix} \end{aligned}$$

$B = I^{4 \times 4}$, $D = 0$, and as activation function:

$$\sigma(x) = \frac{(1 - e^{-0.01x})}{(1 + e^{-0.01x})} \tag{19}$$

with $\Gamma_1 = \text{diag}(0.2, \dots, 0.2)$ and $\sigma_1 = 0.6$, $l_1 = 16$, and initial conditions $\hat{x}(0) = [0 \ 0 \ 0 \ 0]^T$

The HONN observer (16) is used for comparison purposes against the RHONN identifier (12) for the states of the system (15). It is important to note that an identifier can be considered as an observer with $y(k) = x(k)$. Then, this comparison is a valid one.

6.1.2 RHONN identifier

The RHONN identifier (12) activation function is (19), and the values of the matrices $P_i(0)$, $Q_i(0)$ and R_i of the EKF-based algorithm training (20) are:

$$\begin{aligned} P_1(0) = P_2(0) = P_3(0) = P_4(0) &= 1 \times 10^8 \times \text{diag}(4) \\ Q_1(0) = Q_2(0) = Q_3(0) = Q_4(0) &= 5 \times 10^5 \times \text{diag}(4) \\ R_1 = R_2 = R_3 = R_4 &= 1 \times 10^4 \end{aligned} \tag{20}$$

6.1.3 Simulation results

Figures 1, 2, 3, 4, 5, 6, 7 and 8 show the states x_i versus the observed and the identified states.

Tables 1 and 2 show the root-mean-square errors (RMSE) and the absolute deviation of the HONN observer and RHONN identifier errors for the same states of system (15) for a test with sampling time equals to 0.2 s. Tables 3 and 4 show the RMSE and the absolute deviation of the errors of the HONN observer and RHONN identifier errors for a test with sampling time equals to 0.02 s.

The results show a similar performance between the identifier (12) and the observer (16) for the same states of the system (15); however, the errors of the identifier are in most cases smaller than the ones of the observer. Both

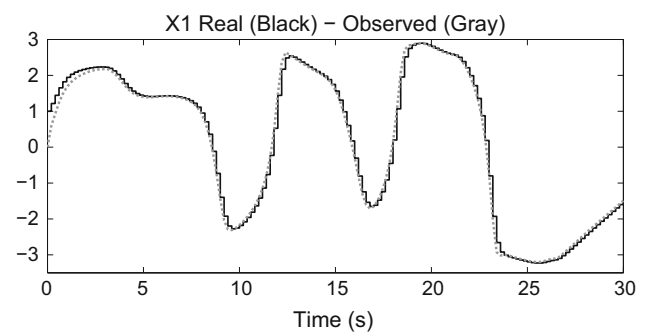


Fig. 1 Real $x_1(k)$ versus observed $\hat{x}_1(k)$

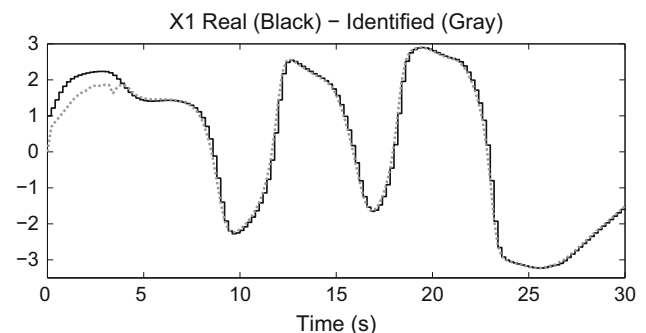


Fig. 2 Real $x_1(k)$ versus identified $\hat{x}_1(k)$

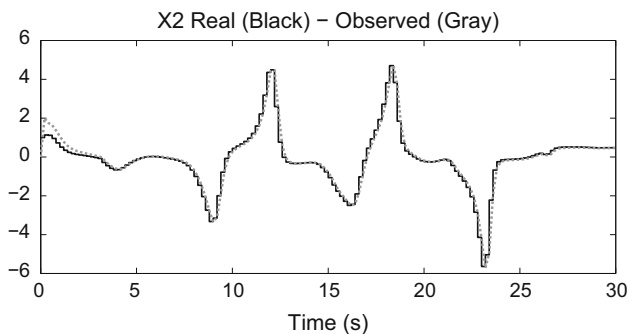


Fig. 3 Real $x_2(k)$ versus observed $\hat{x}_2(k)$

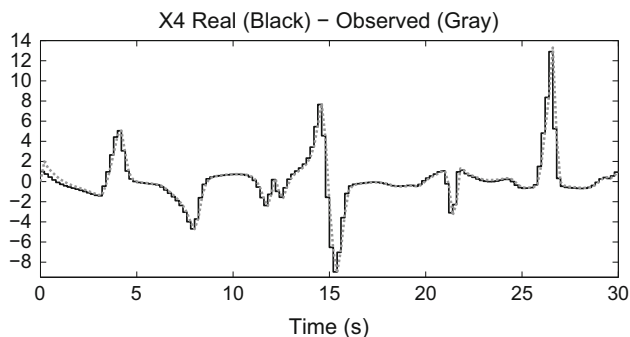


Fig. 7 Real $x_4(k)$ versus observed $\hat{x}_4(k)$

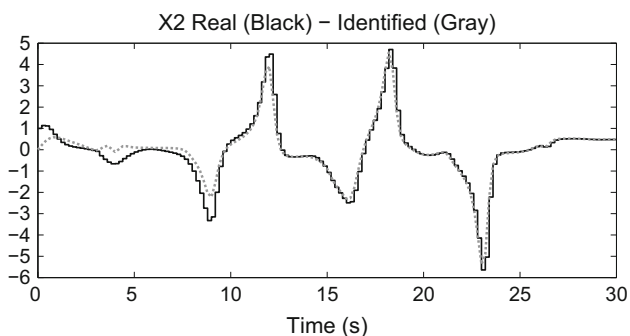


Fig. 4 Real $x_2(k)$ versus identified $\hat{x}_2(k)$

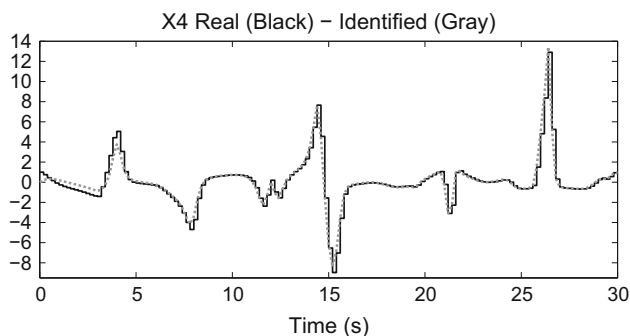


Fig. 8 Real $x_4(k)$ versus identified $\hat{x}_4(k)$

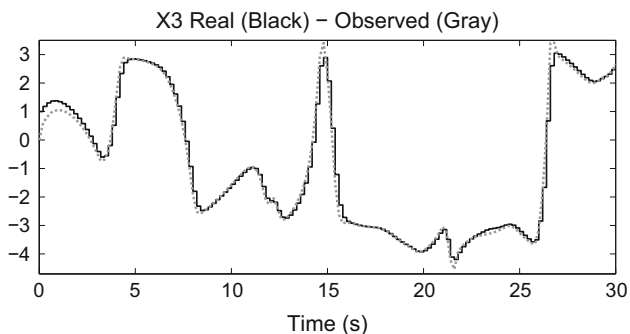


Fig. 5 Real $x_3(k)$ versus observed $\hat{x}_3(k)$

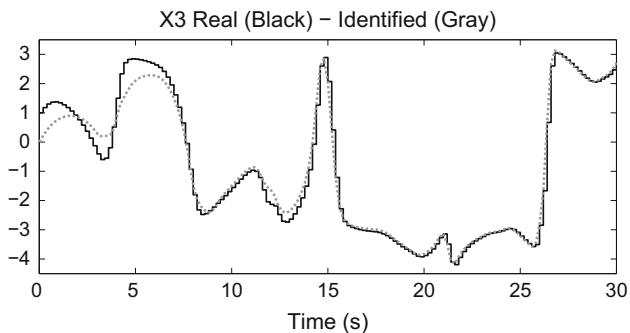


Fig. 6 Real $x_3(k)$ versus identified $\hat{x}_3(k)$

Table 1 RMSE of simulation results with a sampling time equals to 0.2 s

	RMSE			
	x_1	x_2	x_3	x_4
Observer	0.12869	0.53040	0.18226	1.52967
Identifier	0.32413	0.32221	0.32818	0.36338

Table 2 Absolute deviation of simulation results with a sampling time equals to 0.2 s

	Absolute deviation			
	x_1	x_2	x_3	x_4
Observer	0.06331	0.32323	0.09719	0.71550
Identifier	0.19940	0.19558	0.20089	0.36338

Table 3 RMSE of simulation results with a sampling time equals to 0.02 s

	RMSE			
	x_1	x_2	x_3	x_4
Observer	0.88019	0.06228	1.37058	0.14963
Identifier	0.09345	0.07953	0.08431	0.08856

Table 4 Absolute deviation of simulation results with a sampling time equals to 0.02 s

	Absolute deviation			
	x_1	x_2	x_3	x_4
Observer	0.54382	0.03168	0.77253	0.07352
Identifier	0.04952	0.03965	0.03586	0.03708

identifier and observer improved their performance using a smaller sampling time.

6.2 Example 2. Simulation results for a differential robot

In second test for the RHONN identifier (12), a delay is added to each of the seven states of the model of a differential robot presented in [17], and such delays consist in simulating that for 0.1 s, information is not updated. Simulation is made using MATLAB®/Simulink® 2013a, the sampling time for the test is 0.01 s and a total time equals to 15 s and $u_1(t) = \sin(c(t))$ and $u_2(t) = \cos(c(t))$ where $c(t)$ is a chirp signal generated by MATLAB®.

The delays start in this test at 13, 9, 5, 8, 6, 1, 4 s, respectively, for each state.

Table 5 shows the absolute deviation of the error and RMSE of the results of the simulation test.

Figures 9, 10, 11, 12, 13, 14 and 15 show the behavior of the neural identifier before and after the delay where information cannot be updated.

6.3 Example 3. Linear induction motor

As third test for the neural identifier (12), simulation results and real-time result are shown in this section using a linear induction motor (LIM) model and a laboratory prototype for real-time experimental results.

6.3.1 Simulation results for a linear induction motor

A delay has been added to the position state of the model of the LIM in [11]. Such delay consists in simulating that for

Table 5 Absolute deviation of the error and RMSE of simulation test of a differential robot

	Absolute deviation	RMSE
Position X	1.72779×10^{-3}	2.31102×10^{-2}
Position Y	7.16522×10^{-4}	7.59874×10^{-3}
Theta	1.63137×10^{-3}	6.93572×10^{-3}
Angular velocity 1	2.62689×10^{-3}	1.19412×10^{-2}
Angular velocity 2	1.02881×10^{-2}	1.63750×10^{-2}
Current 1	3.60196×10^{-3}	2.26067×10^{-2}
Current 2	6.64885×10^{-3}	2.03897×10^{-2}

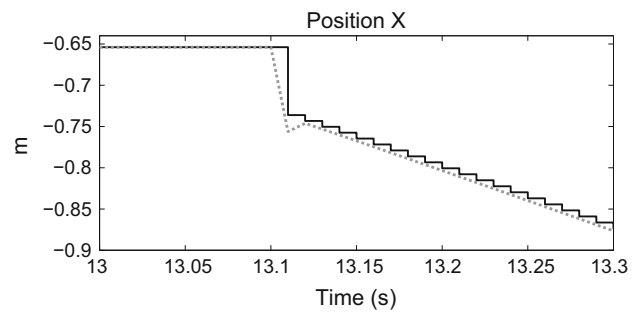


Fig. 9 Identification of the position X of a differential robot. Real signal (black solid line) versus identified signal (gray dotted line)

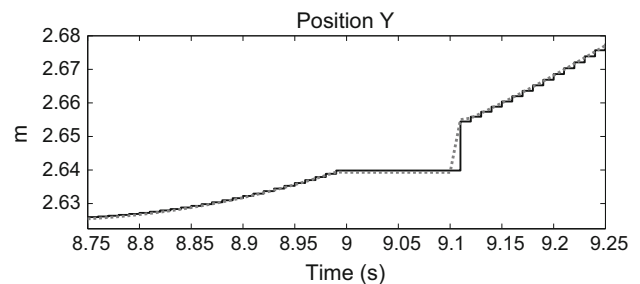


Fig. 10 Identification of the position Y of a differential robot. Real signal (black solid line) versus identified signal (gray dotted line)

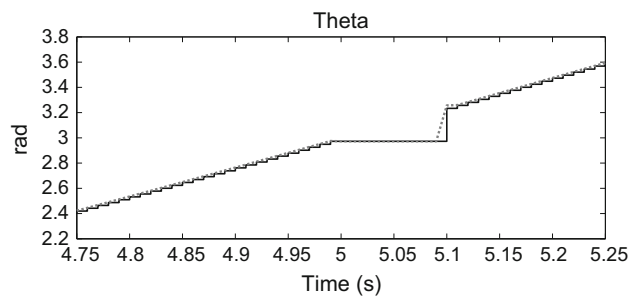


Fig. 11 Identification of theta of a differential robot. Real signal (black solid line) versus identified signal (gray dotted line)

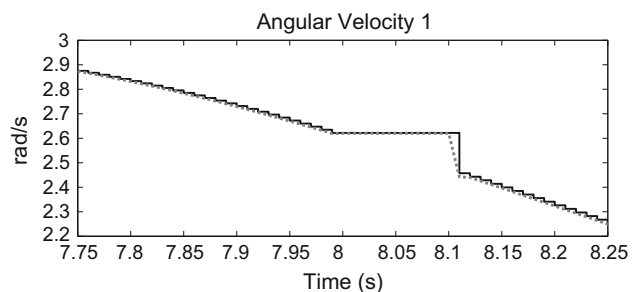


Fig. 12 Identification of the angular velocity 1 of a differential robot. Real signal (black solid line) versus identified signal (gray dotted line)

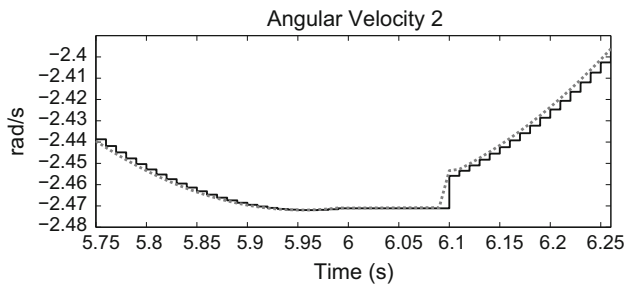


Fig. 13 Identification of the angular velocity 2 of a differential robot. Real signal (black solid line) versus identified signal (gray dotted line)

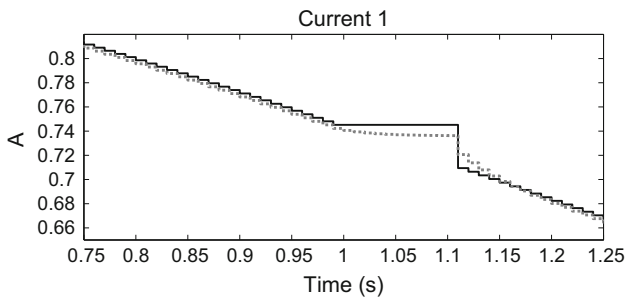


Fig. 14 Identification of the current 1 of a differential robot. Real signal (black solid line) versus identified signal (gray dotted line)

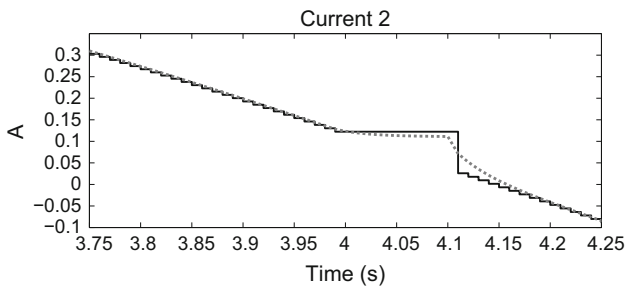


Fig. 15 Identification of the current 2 of a differential robot. Real signal (black solid line) versus identified signal (gray dotted line)

1 s, it is not possible to update the information of the LIM states starting at 2.1 s, sampling time equals to 0.0001 s and u_x is and u_β are defined as (21) and (22), respectively, where t is time.

$$u_x(t) = \begin{cases} 0 & \text{if } t < 2s \\ 255 \left(170 \cos \left(0.3t \left(\frac{98.6}{6}t - \frac{98.6}{3} \right) \right) \right) & \text{if } t \geq 2s \end{cases} \quad (21)$$

$$u_\beta(t) = \begin{cases} 0 & \text{if } t < 2s \\ 255 \left(170 \sin \left(0.3t \left(\frac{98.6}{6}t - \frac{98.6}{3} \right) \right) \right) & \text{if } t \geq 2s \end{cases} \quad (22)$$

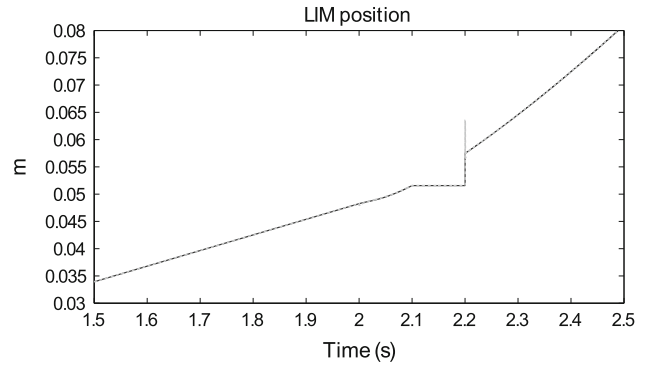


Fig. 16 Identification of LIM position. Position (black solid line) versus identified position (gray dotted line)

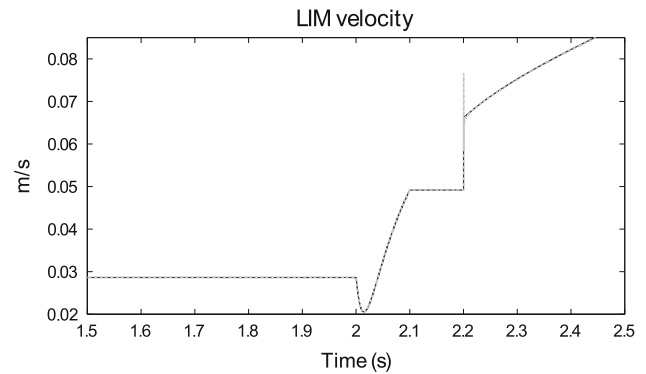


Fig. 17 Identification of LIM velocity. Velocity (black solid line) versus identified velocity (gray dotted line)

Table 6 Absolute deviation of the error and RMSE of LIM position and velocity at simulation test

	Absolute deviation	RMSE
Position	8.7961×10^{-6}	6.8560×10^{-5}
Velocity	4.4238×10^{-5}	1.5947×10^{-4}

Figure 16 shows LIM position and identified LIM position, and Fig. 17 shows LIM velocity and identified LIM velocity. Table 6 shows the RMSE and absolute deviation of the identification errors.

6.3.2 Experimental Results for a LIM

Using the prototype shown in Fig. 18 for real-time test with a LIM presented in [26], the test of the previous section is made at real time.

The prototype consists mainly of a dSPACE³ DS1104 controller board and its connector panel, a Linear Induction Motor Lab-Volt⁴ 8228, a linear encoder SENC 50⁵, a

³ dSPACE is a registered trademark of DSPACE GmbH.

⁴ LAB-Volt is a registered trademark of Lab-Volt Systems, Inc.

⁵ SENC 50 is a registered trademark of ACU-RITE.

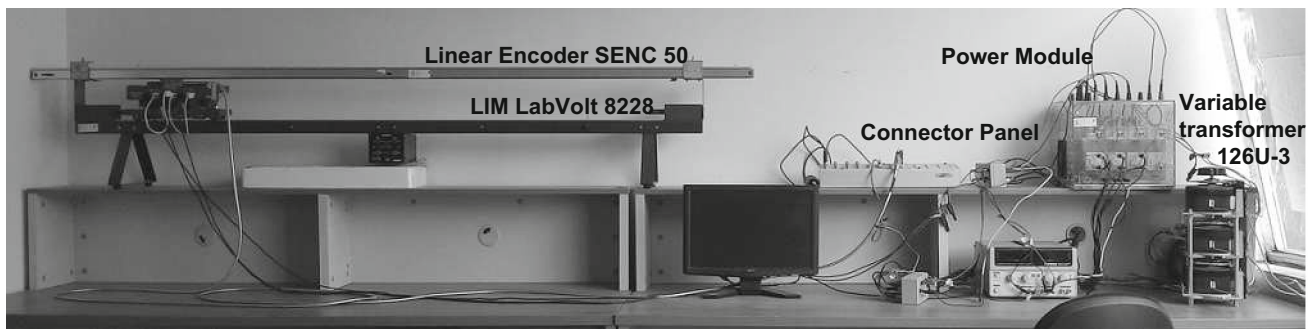


Fig. 18 LIM laboratory prototype

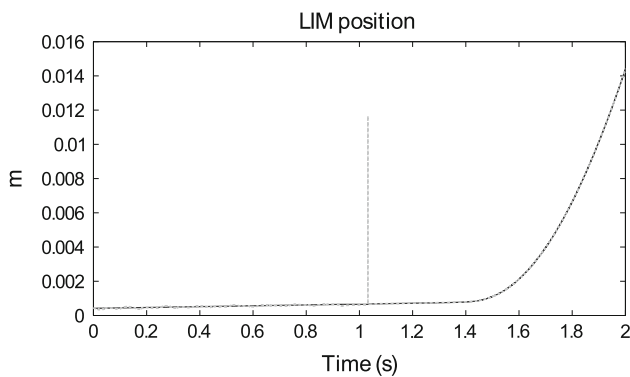


Fig. 19 Real-time identification of LIM position. Position (black solid line) versus identified position (gray dotted line)

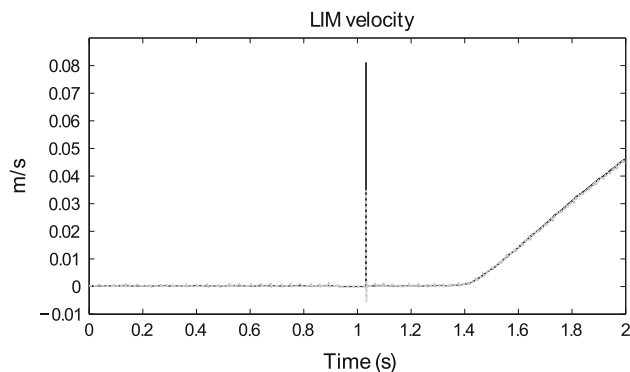


Fig. 20 Real-time identification of LIM velocity. Velocity (black solid line) versus identified velocity (gray dotted line)

POWERSTAT^{®6} variable transformer type 126U-3 and a power module. The prototype is operated using MATLAB[®]/Simulink[®] and dSPACE[®] software ControlDesk^{®7}.

The real-time test has a delay similar to that starts at 1.1 s. Figure 19 shows real LIM position and the identified

Table 7 Absolute deviation of the error and RMSE of LIM position and velocity at real-time test

	Absolute deviation	RMSE
Position	7.5676×10^{-6}	6.8973×10^{-5}
Velocity	1.3028×10^{-4}	6.1873×10^{-4}

LIM position, and Fig. 20 shows real LIM velocity and the identified LIM velocity, a sampling time equals to 0.0003 s and u_α is and u_β are defined as (21) and (22), respectively.

It is important to note that the high-frequency signals presented in Figures [SubEquationDirect] (18a) and (19) are due to encoder limitations. However, from Table 7, it is possible to conclude that the RHONN can identify such signals besides limitations.

7 Conclusion

In this work, the performance of a RHONN identifier (12) trained with a EKF-based training algorithm (8) is shown. Its results for the states of a discrete nonlinear time-delay system (15) are compared with the results of a HONN observer (16) design in [20], and a similar behavior is observed; however, for states x_2 and x_4 the RHONN identifier has a better performance. In order to support the simulation graphs, Table 1 and Table 2 contain the RMSE and absolute deviation of the errors of the simulation test with a sampling time equals to 0.2s. RHONN identifier performance is improved with a smaller sampling time as can be seen in Tables 3 and 4 which contain the RMSE and absolute deviation of the errors of a simulation test with a sampling time equals to 0.02s. Additionally, the identifier (12) is easier to implement in real time due to the fact that it has less parameters to be tuned.

Results of the second test for the RHONN identifier are performed by adding a delay to a LIM model showing an excellent performance.

As future work, it is a plan to extend this work to stochastic systems and loss of packets.

⁶ POWERSTAT is a registered trademark of Superior Electric Holding Group LLC.

⁷ ControlDesk is a registered trademark of DSPACE GmbH.

Acknowledgments The authors thank the support of CONACYT, Mexico, through Projects 106838Y, 156567Y and INFR-229696. They also thank the very useful comments of the anonymous reviewers, which help to improve the paper.

Appendix

Proof of Theorem 1 Step 1, for $\mathbf{V}(k)$. First consider, the equation for each i -th neuron ($i = 1, \dots, n$)

$$\begin{aligned} V_i(k) &= \gamma_i e_i^2(k) + \tilde{w}_i^T(k) P_i(k) \tilde{w}_i(k) \\ \Delta V_i(k) &= V(k+1) - V(k) \\ &= \gamma_i e_i^2(k+1) \\ &\quad + \tilde{w}_i^T(k+1) P_i(k+1) \tilde{w}_i(k+1) \\ &\quad - \tilde{w}_i^T(k) P_i(k) \tilde{w}_i(k) - \gamma_i e_i^2(k) \end{aligned} \tag{23}$$

Using (10) and (14) in (23)

$$\begin{aligned} \Delta V_i(k) &= [\tilde{w}_i(k) - \eta_i K_i(k) e_i(k)]^T [P_i(k) - A_i(k)] \\ &\quad \times [\tilde{w}_i(k) - \eta_i K_i(k) e_i(k)] \\ &\quad + \gamma_i [\tilde{w}_i(k) z_i(x(k-l), u(k)) + \epsilon_{z_i}]^2 \\ &\quad - \tilde{w}_i(k) P_i(k) \tilde{w}_i(k) - \gamma_i e_i^2(k) \end{aligned} \tag{24}$$

with

$$A_i(k) = K_i(k) H_i^T(k) P_i(k) + Q_i(k) \tag{25}$$

then, (24) can be expressed as

$$\begin{aligned} \Delta V_i(k) &= \tilde{w}_i^T(k) P_i(k) \tilde{w}_i(k) \\ &\quad - \eta_i e_i(k) K_i^T(k) P_i(k) \tilde{w}_i(k) \\ &\quad - \tilde{w}_i^T(k) A_i(k) \tilde{w}_i(k) \\ &\quad + \eta_i e_i(k) K_i^T(k) A_i(k) \tilde{w}_i(k) \\ &\quad - \eta_i e_i(k) \tilde{w}_i^T(k) P_i(k) K_i(k) \\ &\quad + \eta_i^2 e_i^2(k) K_i^T(k) P_i(k) K_i(k) \\ &\quad + \eta_i e_i(k) \tilde{w}_i^T(k) A_i(k) K_i(k) \\ &\quad - \eta_i^2 e_i^2(k) K_i^T(k) A_i(k) K_i(k) \\ &\quad + \gamma_i (\tilde{w}_i(k) z_i(x(k-l), u(k)))^2 \\ &\quad + \gamma_i 2\epsilon_{z_i} \tilde{w}_i(k) z_i(x(k), u(k)) \\ &\quad + \gamma_i \epsilon_{z_i}^2 - \tilde{w}_i^T(k) P_i(k) \tilde{w}_i(k) - \gamma_i e_i^2(k) \end{aligned} \tag{26}$$

Using the inequalities

$$\begin{aligned} X^T X + Y^T Y &\geq 2X^T Y \\ X^T X + Y^T Y &\geq -2X^T Y \\ -\lambda_{\min}(P) X^2 &\geq -X^T P X \geq -\lambda_{\max}(P) X^2 \end{aligned} \tag{27}$$

which are valid $\forall X, Y \in \mathbb{R}^n, \forall P \in \mathbb{R}^{n \times n}, \mathfrak{P} = \mathfrak{P}^T >$, then (26) can be rewritten as

$$\begin{aligned} \Delta V_i(k) &\leq -\tilde{w}_i^T(k) A_i(k) \tilde{w}_i(k) \\ &\quad - \eta_i^2 e_i^2(k) K_i^T(k) A_i(k) K_i(k) \\ &\quad + \tilde{w}_i^T(k) \tilde{w}_i(k) + e_i^2(k) \\ &\quad + \eta_i^2 e_i^2(k) K_i^T(k) P_i(k) P_i(k) K_i(k) \\ &\quad + \eta_i^2 \tilde{w}_i^T A_i(k) K_i(k) K_i^T(k) A_i^T(k) \tilde{w}_i(k) \\ &\quad + \eta_i^2 e_i^2(k) K_i^T(k) P_i(k) K_i(k) \\ &\quad + 2\gamma_i (\tilde{w}_i(k) z_i(x(k-l), u(k)))^2 \\ &\quad + 2\gamma_i \epsilon_{z_i}^2 - \gamma_i e_i^2(k) \end{aligned} \tag{28}$$

Then,

$$\begin{aligned} \Delta V_i(k) &\leq -\|\tilde{w}_i(k)\|^2 \lambda_{\min}(A_i(k)) \\ &\quad - \eta_i^2 |e_i(k)|^2 \|K_i(k)\|^2 \lambda_{\min}(A_i(k)) \\ &\quad + \eta_i^2 |e_i(k)|^2 \|K_i(k)\|^2 \lambda_{\max}^2(P_i(k)) \\ &\quad + 2\eta_i^2 |e_i(k)|^2 \|K_i(k)\|^2 \\ &\quad + \|\tilde{w}_i(k)\|^2 \lambda_{\max}(P_i(k)) \\ &\quad + \|\tilde{w}_i(k)\|^2 \lambda_{\max}^2(A_i(k)) \\ &\quad + 2\gamma_i \|\tilde{w}_i(k)\|^2 \|z_i(x(k-l), u(k))\|^2 \\ &\quad + 2\gamma_i \epsilon_{z_i}^2 - \gamma_i |e_i(k)|^2 \end{aligned} \tag{29}$$

Defining

$$\begin{aligned} E_i(k) &= \lambda_{\min}(A_i(k)) - \lambda_{\max}^2(A_i(k)) \\ &\quad - 2\gamma_i \|z_i(x(k-l), u(k))\|^2 + \lambda_{\max}(P_i(k)) \\ F_i(k) &= \gamma_i + \eta_i^2 \|K_i(k)\|^2 \lambda_{\min}(A_i(k)) \\ &\quad - \eta_i^2 \|K_i(k)\|^2 \lambda_{\max}^2(P_i(k)) - 2\eta_i^2 \|K_i(k)\|^2 \end{aligned}$$

and selecting η_i, γ_i, Q_i and R_i , such that $E_i > 0$ and $F_i > 0, \forall k$, then, (29) can be expressed as

$$\begin{aligned} \Delta V_i(k) &\leq -\|\tilde{w}_i(k)\|^2 E_i(k) - |e_i(k)|^2 F_i(k) \\ &\quad + 2\gamma_i \delta \epsilon_{z_i}^2 \end{aligned} \tag{30}$$

Hence, $\Delta V_i(k) < 0$ when

$$\|\tilde{w}_i(k)\| > \sqrt{\frac{2\gamma_i \epsilon_{z_i}^2}{E_i(k)}} \equiv \kappa_1 \tag{31}$$

or

$$|e_i(k)| > \sqrt{\frac{2\gamma_i \epsilon_{z_i}^2}{F_i(k)}} \equiv \kappa_2 \tag{32}$$

Step 2. Now for $\mathbf{V}(k)$, consider the Lyapunov function candidate.

$$\begin{aligned} \mathbf{V}(k) &= \sum_{i=1}^n \tilde{w}_i^T(k) P_i(k) \tilde{w}_i(k) + \gamma_i e_i^2(k) \\ \Delta \mathbf{V}(k) &= \sum_{i=1}^n (\tilde{w}_i^T(k+1) P_i(k+1) \tilde{w}_i(k+1) \\ &\quad + \gamma_i e_i^2(k+1) - \tilde{w}_i^T(k) P_i(k) \tilde{w}_i(k) \\ &\quad - \gamma_i e_i^2(k)) \end{aligned} \tag{33}$$

Using (10) and (14) in (23)

$$\begin{aligned} \Delta \mathbf{V}(k) &= \sum_{i=1}^n ([\tilde{w}_i(k) - \eta_i K_i(k) e_i(k)]^T \\ &\quad \times [P_i(k) - A_i(k)] [\tilde{w}_i(k) - \eta_i K_i(k) e_i(k)] \\ &\quad + \gamma_i [\tilde{w}_i(k) z_i(x(k-l), u(k)) + \epsilon_{z_i}]^2 \\ &\quad - \tilde{w}_i(k) P_i(k) \tilde{w}_i(k) - \gamma_i e_i^2(k)) \end{aligned} \tag{34}$$

Defining

$$\begin{aligned} A_i(k) &= K_i(k) H_i^T(k) P_i(k) + Q_i(k) \\ E_i(k) &= \lambda_{\min}(A_i(k)) - \lambda_{\max}^2(A_i(k)) \\ &\quad - 2\gamma_i \|z_i(x(k-l), u(k))\|^2 + \lambda_{\max}(P_i(k)) \\ F_i(k) &= \gamma_i + \eta_i^2 \|K_i(k)\|^2 \lambda_{\min}(A_i(k)) \\ &\quad - \eta_i^2 \|K_i(k)\|^2 \lambda_{\max}^2(P_i(k)) - 2\eta_i^2 \|K_i(k)\|^2 \end{aligned}$$

and selecting η_i, γ_i, Q_i and R_i , such that $E_i > 0$ and $F_i > 0, \forall k$, then, (24) can be expressed as

$$\begin{aligned} \Delta \mathbf{V}(k) &\leq \sum_{i=1}^n (-\|\tilde{w}_i(k)\|^2 E_i(k) \\ &\quad - |e_i(k)|^2(k) F_i(k) + 2\gamma_i \epsilon_{z_i}^2) \end{aligned}$$

Hence, $\Delta \mathbf{V}(k) < 0$ when (31) or (32) is fulfilled

Therefore, considering Step 1 and Step 2 for (23), the solution of (10) and (14) is SGUUB. \square

Remark 1 Considering Theorem 1 and its proof, it can be easily shown that the result can be extended to a system (12) with multiple delays like $x(k-l_i)$ with $i = 1, 2, \dots$, can be used instead of $x(k-l)$ in (3) and/or for time-varying delays $x(k-l_i(k))$ with $l_i(k)$ bounded by $l_i(k) \leq l$.

References

1. Alfaro-Ponce M, Argüelles A, Chairez I (2014) Continuous neural identifier for uncertain nonlinear systems with time delays in the input signal. *Neural Netw* 60:53–66
2. Bedoui S, Ltaief M, Abderrahim K (2012) New results on discrete-time delay systems identification. *Int J Autom Comput* 9(6):570–577
3. Bengio Y, Simard P, Frasconi P (1994) Learning long-term dependencies with gradient descent is difficult. *IEEE Trans Neural Netw* 5(2):157–166
4. Boukas E, Liu Z (2002) *Deterministic and stochastic time-delay systems*. Control Engineering Birkhäuser, Birkhäuser, Boston

5. Chen C, Wen GX, Liu YJ, Wang FY (2014) Adaptive consensus control for a class of nonlinear multiagent time-delay systems using neural networks. *IEEE Trans Neural Netw Learn Syst Actions* 25(6):1217–1226
6. Fu L, Li P (2013) The research survey of system identification method. In: 2013 5th international conference on intelligent human-machine systems and cybernetics (IHMSC), vol 2. pp 397–401
7. Ge SS, Tee KP (2005) Adaptive neural network control of nonlinear mimo time-delay systems with unknown bounds on delay functionals. In: Proceedings of the 2005 American control conference, 2005, vol 7. pp 4790–4795
8. Haykin S (1999) *Neural networks: a comprehensive foundation*. Prentice Hall International, Englewood Cliffs
9. Haykin S (2004) *Kalman filtering and neural networks*. Wiley, New York
10. Hermans M, Schrauwen B (2010) One step backpropagation through time for learning input mapping in reservoir computing applied to speech recognition. In: Proceedings of 2010 IEEE international symposium on circuits and systems (ISCAS). pp 521–524
11. Hernandez-Gonzalez M, Sanchez E, Loukianov A. (2008) Discrete-time neural network control for a linear induction motor. In: IEEE international Symposium on intelligent control 2008 (ISIC 2008). pp 1314–1319
12. Hochreiter S (1998) Recurrent neural net learning and vanishing gradient. *Int J Uncertain Fuzziness Knowl Based Syst* 6(2):107–116
13. Hong Y, Ren X, Qin H (1996) Neural identification and control of uncertain nonlinear systems with time delay. In: Proceedings of the 35th IEEE conference on decision and control, 1996, vol 4. pp 3802–3803
14. Krstic M, Bekiaris-Liberis N (2012) Control of nonlinear delay systems: A tutorial. In: 2012 IEEE 51st annual conference on decision and control (CDC). pp 5200–5214
15. Kurose J, Ross K (2013) *Computer networking: a top-down approach*. Always Learning. Pearson, London
16. Leondes C (1998) *Neural network systems techniques and applications: advances in theory and applications*. Elsevier Science, Amsterdam
17. Lopez-Franco M, Landa D, Alanis A, Lopez-Franco C, Arana-Daniel N (2014) Discrete-time inverse optimal neural control for a tracked all terrain robot. In: XVI IEEE autumn meeting of power, electronics and computer science ROPEC 2014 international. pp 70–75
18. Mahmoud M (2000) *Robust control and filtering for time-delay systems*. Automation and control engineering. CRC Press
19. Mahmoud M (2010) *Switched time-delay systems: stability and control*. Springer, Berlin
20. Na J, Herrmann G, Ren X, Barber P (2009) Nonlinear observer design for discrete mimo systems with unknown time delay. In: Proceedings of the 48th IEEE conference on decision and control, 2009 held jointly with the 2009 28th Chinese control conference. (CDC/CCC 2009). pp 6137–6142
21. Ngoc P (2015) Novel criteria for exponential stability of nonlinear differential systems with delay. *IEEE Trans Autom Control Actions* 60(2):485–490
22. Norgaard M (2000) *Neural networks for modelling and control of dynamic systems: a practitioner’s handbook*. Springer, London
23. Rajapakse J, Wang L (2004) *Neural information processing: research and development*. Springer, Berlin
24. Ren X, Rad A (2007) Identification of nonlinear systems with unknown time delay based on time-delay neural networks. *IEEE Trans Neural Netw* 18(5):1536–1541
25. Richard JP (2003) Time-delay systems: an overview of some recent advances and open problems. *Automatica* 39:1667–1694

26. Rios J, Alanis A, Rivera J, Hernandez-Gonzalez M (2013) Real-time discrete neural identifier for a linear induction motor using a dSPACE DS1104 board. In: The 2013 international joint conference on neural networks (IJCNN). pp 1–6
27. Rovithakis G, Christodoulou M (2011) Adaptive control with recurrent high-order neural networks: theory and industrial applications. *Advances in industrial control*. Springer, London
28. Sanchez E, Alanis A, Garcia A, Loukianov A (2008) Discrete-time high order neural control: trained with Kalman filtering. Springer, Berlin
29. Song Y, Grizzle J (1992) The extended Kalman filter as a local asymptotic observer for nonlinear discrete-time systems. *Am Control Conf 1992*:3365–3369
30. Xian-Ming T, Jin-Shou Y (2008) Stability analysis for discrete time-delay systems. In: Fourth international conference on networked computing and advanced information management, 2008 (NCM'08), vol 1. pp 648–651
31. Xu H, Jagannathan S (2013) Neural network based finite horizon stochastic optimal controller design for nonlinear networked control systems. In: The 2013 international joint conference on neural networks (IJCNN). pp 1–7
32. Xu H, Jagannathan S (2013) Stochastic optimal controller design for uncertain nonlinear networked control system via neuro dynamic programming. *IEEE Trans Neural Netw Learn Syst* 24(3):471–484
33. Xu Z, Li X (2010) Control design based on state observer for nonlinear delay systems. In: 2010 Chinese control and decision conference (CCDC). pp 1946–1950
34. Yi S (2010) Time-delay systems: analysis and control using the Lambert W function. World Scientific, Singapore
35. Yu W, Sanchez E (2009) *Advances in computational intelligence*. Springer, Berlin
36. Zhong Q (2006) *Robust control of time-delay systems*. Springer, Berlin