



## Decentralized discrete-time neural control for a Quanser 2-DOF helicopter

M. Hernandez-Gonzalez<sup>a</sup>, A.Y. Alanis<sup>c</sup>, E.A. Hernandez-Vargas<sup>b,\*</sup>

<sup>a</sup> CINVESTAV, Campus Guadalajara, Zapopan, Mexico

<sup>b</sup> HZI, SIMM, Inhoffenstrabe 7, 38124 Braunschweig, Germany

<sup>c</sup> CUCEI, Universidad de Guadalajara, Zapopan, Mexico

### ARTICLE INFO

#### Article history:

Received 1 January 2011

Received in revised form 3 January 2012

Accepted 21 February 2012

Available online 20 March 2012

### ABSTRACT

Control design for helicopters is a complicated and challenging problem due to the strong inter-couplings and nonlinear uncertainties in the system model. This paper deals with the decentralized control problem for the output trajectory tracking in a Quanser 2 degree of freedom (DOF) helicopter. High order neural network (HONN) is an important technique to approximate non-linearities in the model. Two different discrete-time schemes with a decentralized structure are used. Neural backstepping and neural sliding mode block control techniques are considered in order to control pitch and yaw positions. On one hand, backstepping control divides the whole system into two subsystems which are used to track the pitch and yaw references respectively. Real and virtual controls are approximated by HONNs. On the other hand, block control technique is applied to HONNs which can identify the system helicopter model. Each discrete-time high order neural network is trained on-line with an extended Kalman filter based algorithm. Without the previous knowledge of the plant parameters neither its model, we show via simulations the good performance of both strategies. The block control technique presents slightly better results than backstepping algorithm.

© 2012 Elsevier B.V. All rights reserved.

### 1. Introduction

Helicopters have become very popular for short-distance transportation because of its ability to land and take off in small areas. Helicopters have been adopted for a wide range of services, including air-sea rescue, fire fighting, traffic control, tourism among others [1]. A helicopter has four flight control inputs; the cyclic, the collective, the anti-torque pedals, and the throttle. The control is called the cyclic because it changes the pitch of the rotor blades cyclically. The collective changes the pitch angle of all the main rotor blades collectively and independently of their position. The anti-torque pedals serve a similar purpose, namely to control the direction in which the nose of the aircraft is pointed. The throttle maintains enough engine power to keep the rotor RPM within allowable limits in order to keep the rotor producing enough lift for flight [2].

Helicopter dynamics are highly nonlinear, high cross-coupled, unstable, and difficult to model [3]. Moreover, identification and control of a helicopter are complex problems. Identification of the helicopter model Humudoft CE 150 parameters is described in [4], where the determination of each parameter requires a specific

prepared experiment. For this reason, Velagic and Osmic [5] proposed a genetic algorithm for identification of the physical structure of helicopter systems.

Neural Networks (NNs) can play an important role due to the fact that they might be used to model nonlinear systems with any degree of accuracy. This is possible due to the learning and adaptation capabilities of the NNs [6]. Neural structures can be variously classified, the most relevant of them to the purposes of this work are High Order Neural Networks (HONNs) and Recurrent High Order Neural Networks (RHONNs). These offer better results for modeling and controlling nonlinear systems [7].

Several stable NNs control approaches have been proposed based on Lyapunov's stability theory [6,8]. In nonlinear control systems, radial basis function (RBF) neural networks, high order neural networks [8], multilayer neural networks (MNNs) [9] and recurrent high order neural networks are sort of NNs which have widely been used [6].

Various training methods for neural networks are suggested in the literature; among them are the ones based on extended Kalman filtering (EKF) [8] and the well-known back-propagation algorithm [10]. Kalman filter (KF) estimates the state of a linear system with additive state and additive output white noises [11,12]. For KF-based neural network training method, the NNs weights become the states to be estimated by the EKF. The EKF training of neural networks, both feedforward and recurrent ones, has proved to be reliable and practical for many applications [13,14].

\* Corresponding author.

E-mail addresses: [mglez@gdl.cinvestav.mx](mailto:mglez@gdl.cinvestav.mx) (M. Hernandez-Gonzalez), [almayalanis@gmail.com](mailto:almayalanis@gmail.com) (A.Y. Alanis), [abelardo.81@hotmail.com](mailto:abelardo.81@hotmail.com) (E.A. Hernandez-Vargas).

Helicopter dynamics are unstable, therefore control techniques are needed to realize superior performances and high agilities [2,3]. A robust and stable control for a wide azimuth and elevation angles is proposed by Velagic and Osmic [5] using a fuzzy approach. In similar direction, Rios et al. [15] presented a practical design to the stabilization of a three degrees of freedom helicopter.

However, all these approaches are designed in continuous time, discrete-time algorithms are necessary in order to achieve better performances. We study the design of discrete-time neural control algorithms applied to a Quanser 2-DOF helicopter for tracking two reference signals. The first decentralized strategy, discrete-time backstepping control, considers that the whole system is divided into subsystems and each one of them is presented in the strict feedback form or can be transformed in it. The idea behind backstepping is as follows: if we consider the system description as a one-step ahead predictor, then we can transform the one-step ahead predictors into an equivalent maximum  $r$ -step ahead predictor which can predict the futures states, the causality contradiction is avoided by the backstepping controller when it is constructed based on the maximum  $r$ -step ahead predictor [6]. For the second strategy, discrete-time sliding mode control, a decentralized control approach is employed, i.e. the whole system is divided into subsystems, each one of them has an input control and an output control to be controlled. The discrete-time sliding mode control is a continuous function of the states and does not have the problem of chattering [16].

The Quanser 2-DOF helicopter model used in this paper is explained in detail in Section 3. The model considers four states: two states stand for the pitch and yaw position and the remainder states are for the pitch and yaw velocities. Two control inputs are used to force the output to track two references [17], i.e. an input voltage pitch and an input voltage yaw.

The paper is organized as follows. We introduce in Section 2 the mathematical preliminaries and the necessary background of neural networks. The helicopter model and its description are presented in Section 3. The discrete-time backstepping control law development and the class of neural network used for this application are provided in Section 4. Section 5 presents the discrete-time block control strategy. Simulation results and comparisons between the controllers are given in Section 6. We conclude the paper in Section 7.

## 2. Mathematical preliminaries

Throughout this paper,  $\mathbb{R}$  denotes the field of real number,  $\mathbb{R}^n$  stands for the vector space of all  $n$ -tuples of real numbers,  $\mathbb{R}^{n \times n}$  is the space of  $n \times n$  matrices with real entries, and  $\mathbb{Z}^+$  denotes the set of positive natural numbers.

We use  $k$  as the sampling step,  $k \in 0 \cup \mathbb{Z}^+$ ,  $|\bullet|$  as the absolute value and  $\|\bullet\|$  as the Euclidean norm for vectors and as any adequate norm for matrices. To follow a discrete-time approach, we consider the next nonlinear model;

$$x(k+1) = f(x) + Bu(k) \tag{1}$$

where  $x(k) \in \mathbb{R}^n$  is the state of the system,  $u(k) \in \mathbb{R}^m$  is the input control,  $f(x) \in \mathbb{R}^n$ ,  $B \in \mathbb{R}^{n \times m}$ . The system (1) is said to be forced or have inputs. In contrast the system described without explicit presence of an input  $u$  is said to be unforced. For the design of the controller, the next stability definitions are introduced;

**Definition 1.** A subset  $S \in \mathbb{R}^n$  is bounded if there exists  $r > 0$  such that  $|\chi| \leq r$  for all  $\chi \in S$ .

**Definition 2.** The solution of the system (1) is said to be semiglobally uniformly ultimately bounded (SGUUB), if for any  $\Omega$  a compact subset of  $\mathbb{R}^n$  and all  $\chi(k_0) \in \Omega$ , there exists an  $\epsilon > 0$  and a number  $N(\epsilon, \chi(k_0))$  such that  $|\chi(k)| < \epsilon$  for all  $k \geq k_0 + N$ .

### 2.1. Discrete-time HONNs

HONNs are networks that utilize higher combination of input and states. HONNs have impressive computational storage and learning capabilities. In this work, we consider the HONN described by Alanis et al. [8].

$$\psi(w_i(k), z_i(k)) = w_i^T(k)S_i(z(k)) \tag{2}$$

$$S_i(z(k)) = [s_{i_1}(z) s_{i_2}(z) \dots s_{i_{L_i}}(z)]^T \tag{3}$$

$$S_i(z(k)) = \left[ \prod_{j \in I_1} [s(z_j)]^{d_j(i_1)} \prod_{j \in I_2} [s(z_j)]^{d_j(i_2)} \dots \prod_{j \in I_{L_i}} [s(z_j)]^{d_j(i_{L_i})} \right]^T \tag{4}$$

where  $[z_1 \ z_2 \ \dots \ z_q] \in \Omega_z \subset \mathbb{R}^q$ ,  $q$  denotes the number of external inputs,  $L$  denotes the NN node number,  $\psi \in \mathbb{R}$ ,  $I_1, I_2, \dots, I_{L_i}$  is a collection of not ordered subsets of  $1, 2, \dots, q$ ,  $S_i(z) \in \mathbb{R}^{L_i}$ ,  $d_j(i_j)$  is a non-negative integer,  $w \in \mathbb{R}^{L_i}$  is an adjustable synaptic weight vector, and  $s_i(z_j)$  is chosen as a hyperbolic tangent function

$$s(z_j) = a \tanh(\beta z_j) + \gamma \tag{5}$$

where  $a, \beta$ , and  $\gamma$  are constant parameters.

**Assumption 1.** For a desired function  $u^*(z) \in \mathbb{R}$ , assume there exists an ideal weight vector  $w_i^* \in \mathbb{R}^{L_i}$  such that the smooth function  $u^*(z)$  can be approximated by an ideal NN on a compact set  $\Omega_z \subset \mathbb{R}^q$

$$u^*(z) = w_i^{*T} S(z) + \epsilon_z \tag{6}$$

where  $\epsilon_z \in \mathbb{R}$  is the bounded NN approximation error,  $\|\epsilon_z\|$  can be reduced by increasing the number of adjustable weights. The ideal weight vector  $w_i^*$  is an artificial quantity required for analytical purposes. Assume that this vector exists, but it is an unknown constant whose estimate is given by  $w_i(k) \in \mathbb{R}^{L_i}$ . The estimation error can be defined as:

$$\tilde{w}_i(k) = w_i(k) - w_i^* \tag{7}$$

### 2.2. Extended Kalman filter

Kalman filter is a set of mathematical equations that provides an efficient computational (recursive) solution of the least-square method which estimates the state of a linear system with additive state and output with noises [9,21]. For KF-based neural network training, the network weights become the states to be estimated. In this case, the error between the neural network output and the measured plant output is considered as additive white noise. The training goal is to find the optimal weight values which minimize the predictions error. The on-line EKF-based training algorithm is described by the following system;

$$w_i(k+1) = w_i(k) + \eta_i K_i(k) e_i(k) \tag{8}$$

$$K_i(k) = P_i(k) H_i(k) M_i(k) \quad i = 1, \dots, r$$

$$P_i(k+1) = P_i(k) - K_i(k) H_i^T(k) P_i(k) + Q_i(k)$$

with

$$M_i(k) = [R_i(k) + H_i^T(k) P_i(k) H_i(k)]^{-1} \tag{9}$$

$$e_i(k) = x_i(k) - \chi_i(k)$$

where  $e_i(k) \in \mathbb{R}$  is the respective identification error in each subsystem,  $P_i(k) \in \mathbb{R}^{L_i \times L_i}$  is the weight estimation error covariance matrix at step  $k$ ,  $w_i(k) \in \mathbb{R}^{L_i}$  is the weight (state) vector,  $x_i(k)$  is the  $i$ th plant input control to be approximated by the NN,  $\chi_i(k)$  is the  $i$ th neural network virtual control,  $K_i(k) \in \mathbb{R}^{L_i}$  is the Kalman gain vector,  $Q_i \in \mathbb{R}^{L_i \times L_i}$  is the NN weight estimation noise covariance matrix,  $R_i \in \mathbb{R}$  is the error noise covariance, and  $H_i(k) \in \mathbb{R}^{L_i}$  is a vector, in

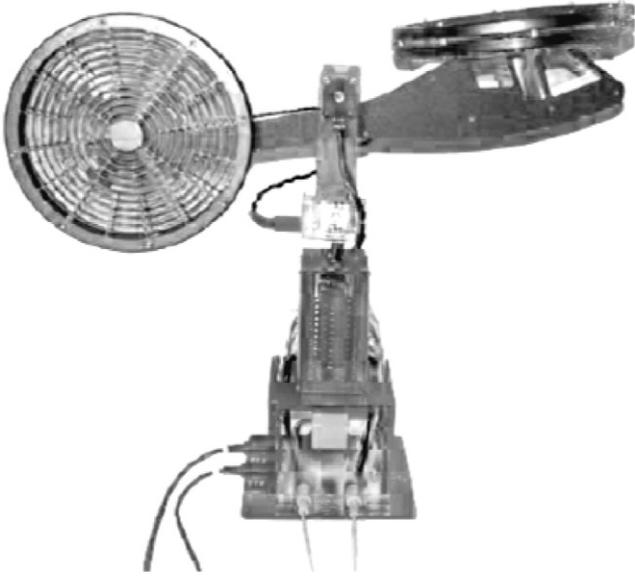


Fig. 1. Quanser 2 degree of freedom helicopter.

Table 1  
Helicopter parameters.

Parameter	Value	Description
$m_{heli}$	1.3872 kg	Mass of the helicopter
$g$	9.8 m/s	Gravity
$l_{cm}$	0.186 cm	Center-of-mass length along the helicopter
$J_{eqp}$	0.0384 kg/m <sup>2</sup>	Moment of inertia about pitch pivot
$J_{eqy}$	0.0432 kg/m <sup>2</sup>	Moment of inertia about yaw pivot
$K_{pp}$	0.204 Nm/V	Thrust torque constant acting on pitch axis from pitch motor
$K_{py}$	0.0068 Nm/V	Thrust torque constant acting on pitch axis from yaw motor
$K_{yy}$	0.072 Nm/V	Thrust torque constant acting on yaw axis from yaw motor
$K_{yp}$	0.0219 Nm/V	Thrust torque constant acting on yaw axis from pitch motor
$B_p$	0.800 N/V	Equivalent viscous damping about pitch axis
$B_y$	0.318 N/V	Equivalent viscous damping about yaw axis
$T$	0.0001 seg	Sampling period

$$c_4 = \frac{K_{pp}}{J_{eqp} + m_{heli}l_{cm}^2}, \quad c_5 = \frac{K_{py}}{J_{eqp} + m_{heli}l_{cm}^2},$$

$$c_6 = \frac{B_y}{J_{eqy} + m_{heli}l_{cm}^2 \cos^2(x_1(k))}, \quad c_7 = \frac{2m_{heli}l_{cm}^2}{J_{eqy} + m_{heli}l_{cm}^2 \cos^2(x_1(k))},$$

$$c_8 = \frac{K_{yp}}{J_{eqy} + m_{heli}l_{cm}^2 \cos^2(x_1(k))}, \quad c_9 = \frac{K_{yy}}{J_{eqy} + m_{heli}l_{cm}^2 \cos^2(x_1(k))}$$

and  $x_1(k)$  represents the angle in pitch axis and  $x_2(k)$  in the yaw axis. The pitch position is defined positive when the nose of the helicopter goes up, and the yaw position is defined positive for a clockwise rotation.  $x_3(k)$  is the pitch velocity and  $x_4(k)$  is the yaw velocity. The input pitch motor voltage is  $V_{mp}(k)$  and  $V_{my}$  is the input yaw motor voltage. Parameter descriptions and values are shown in Table 1.

which each entry  $H_{ij}$  is the derivative of the neural network virtual control,  $\chi_i(k)$ , with respect to one neural network weight,  $w_{ij}(k)$ , given as follows:

$$H_{ij}(k) = \left[ \frac{\partial \chi_i(k)}{\partial w_{ij}(k)} \right]^T \quad (10)$$

where  $i = 1, \dots, r$  and  $j = 1, \dots, L_i$ . Usually  $P_i$  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  $P_i(k)$  to the EKF are bounded, i.e.  $\|H(k)\| \leq \bar{H}$ ,  $\|K(k)\| \leq \bar{K}$ ,  $\|P(k)\| \leq \bar{P}$ , for a detailed explanation of this fact see [10].

### 3. Description of the system

The Quanser 2-DOF helicopter consists of a helicopter model mounted on a fixed base with two propellers that is driven by DC motors, see Fig. 1. The front propeller controls the elevation of the helicopter nose about the pitch axis, and the back propeller controls the side to side motions of the helicopter about the yaw axis [17]. The model is described in continuous-time, however in the present work a discrete-time approach is considered. Therefore, we propose below the discretization of the helicopter model using Euler technique [18]:

$$\begin{aligned} x_1(k+1) &= x_1(k) + x_3(k)T \\ x_2(k+1) &= x_2(k) + x_4(k)T \\ x_3(k+1) &= x_3(k) - c_1 \cos(x_1(k))T - c_2 x_3(k)T \\ &\quad - c_3 \sin(x_1(k)) \cos(x_1(k)) x_4^2(k)T + c_4 TV_{mp} + c_5 TV_{my}(k) \\ x_4(k+1) &= x_4(k) - c_6 x_4(k)T + c_7 \sin(x_1(k)) \cos(x_1(k)) x_3(k) x_4(k)T \\ &\quad + c_8 TV_{mp} + c_9 TV_{my}(k). \end{aligned} \quad (11)$$

The output to be controlled is given by:

$$y(k) = \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} \quad (12)$$

where the parameters  $c_i$ ,  $i = 1, \dots, 9$  in (11) are given by:

$$c_1 = \frac{m_{heli}g l_{cm}}{J_{eqp} + m_{heli}l_{cm}^2}, \quad c_2 = \frac{B_p}{J_{eqp} + m_{heli}l_{cm}^2}, \quad c_3 = \frac{m_{heli}l_{cm}^2}{J_{eqp} + m_{heli}l_{cm}^2}$$

### 4. Discrete-time backstepping control law development

It is supposed that the system in (1) can be represented as  $r$  nonlinear subsystems [28]:

$$\begin{aligned} x_1^j(k+1) &= f_1^j(x_1^j(k)) + b_1^j x_2^j(k) + \Gamma_{1k} \\ x_2^j(k+1) &= f_2^j(x_2^j(k)) + b_2^j x_3^j(k) + \Gamma_{2k} \\ &\vdots \\ x_i^j(k+1) &= f_i^j(x_i^j(k)) + b_i^j x_{i+1}^j(k) + \Gamma_{ik} \\ &\vdots \\ x_r^j(k+1) &= f_r^j(x_r^j(k)) + b_r^j u^j(k) + \Gamma_{rk} \end{aligned} \quad (13)$$

where  $\bar{x}_i^j(k) = [x_1^j(k) x_2^j(k) \dots x_i^j(k)]$ ,  $i = 1, \dots, r$ ,  $x_i^j$  is the  $i$ th subsystem of the  $j$  subplant,  $b_i^j$  is a constant of appropriate dimension. The interconnection terms  $\Gamma_{rk}$  reflect the interconnection between the  $i$ th and the  $k$ th subsystem. The following theorem establishes the properties of the learning law.

**Theorem 1.** For the system (13) the HONN (2) trained with the EKF-based algorithm (8) to approximate the virtual and control laws, ensures that the tracking error (9) is semiglobally uniformly ultimately bounded (SGUUB); moreover the HONN weights remain bounded. Proof can be found in [8].

#### 4.1. Helicopter HONN

The development of the virtual and real control laws for the helicopter model is derived as follows: the system model (11) is

divided into two subsystems (13) and a control law (2) for each subsystem is designed. The first subsystem is used for tracking the pitch reference, and the second one is used for tracking the yaw reference. The pseudo-control for the axis-pitch is approximated by a HONN; the error which is minimized by the EKF (8)–(9) is the difference between the pitch-axis position and the reference signal. Now, the real control is approximated by another HONN and the error difference between the virtual control and the pseudo-control is reduced by an EKF. The virtual and real control law for controlling yaw position is similar as the control law for the pitch. Activation function for the neural networks is as presented in (5). To the author’s knowledge there is no a procedure of selecting the number of neurons for each neural network neither the degree of each activation function, it depends on the expertise of the designer. It is important to remark that neural networks are trained on-line. The virtual and real controls proposed are described as follows:

$$\begin{aligned} \alpha_1(k) = & w_{11}(k)S^2(x_1(k)) + w_{12}(k)S^2(x_2(k)) + w_{13}(k)S^2(x_{1ref}(k)) \\ & + w_{14}(k)S^2(x_{2ref}(k)) + w_{15}(k)S(x_1(k))S(x_2(k)) \\ & + w_{16}(k)S(x_{1ref}(k))S(x_{2ref}(k)) + w_{17}(k)S(x_1(k))S(x_{1ref}(k)) \\ & + w_{18}(k)S(x_2(k))S(x_{2ref}(k)) + w_{19}(k)S(x_1(k)) \\ & + w_{110}(k)S(x_2(k)) + w_{111}(k)S(x_{1ref}(k)) + w_{112}(k)S(x_{2ref}(k)) \end{aligned} \tag{14}$$

$$\begin{aligned} \alpha_2(k) = & w_{21}(k)S^2(x_1(k)) + w_{22}(k)S^2(x_2(k)) + w_{23}(k)S^2(x_{1ref}(k)) \\ & + w_{24}(k)S^2(x_{2ref}(k)) + w_{25}(k)S(x_1(k))S(x_2(k)) \\ & + w_{26}(k)S(x_{1ref}(k))S(x_{2ref}(k)) \\ & + w_{28}(k)S(x_2(k))S(x_{2ref}(k)) + w_{29}(k)S(x_1(k)) \\ & + w_{210}(k)S(x_2(k)) + w_{27}(k)S(x_1(k))S(x_{1ref}(k)) \end{aligned} \tag{15}$$

$$\begin{aligned} V_{mp}(k) = & w_{31}(k)S(x_1)S(x_2(k)) + w_{32}(k)S(x_1(k))S(x_3(k)) \\ & + w_{33}(k)S(x_2(k))S(x_4(k)) + w_{34}(k)S(x_3(k))S(x_4(k)) \\ & + w_{35}(k)S(\alpha_1(k))S(\alpha_2(k)) + w_{36}(k)S(x_2(k))S(\alpha_2(k)) \\ & + w_{37}(k)S(x_4(k))S(\alpha_1(k)) + w_{38}(k)S(x_2(k))S(x_3(k)) \\ & + w_{39}(k)S(x_1(k)) + w_{310}(k)S(x_2(k)) + w_{311}(k)S(x_3(k)) \\ & + w_{312}(k)S(x_4(k)) \end{aligned} \tag{16}$$

$$\begin{aligned} V_{my}(k) = & w_{41}(k)S(x_1(k))S(x_2(k)) + w_{42}(k)S(x_1(k))S(x_3(k)) \\ & + w_{43}(k)S(x_2(k))S(x_4(k)) + w_{44}(k)S(x_3(k))S(x_4(k)) \\ & + w_{45}(k)S(\alpha_1(k))S(\alpha_2(k)) + w_{46}(k)S(x_2(k))S(\alpha_2(k)) \\ & + w_{47}(k)S(x_4(k))S(\alpha_1(k)) + w_{48}(k)S(x_2(k))S(x_3(k)) \\ & + w_{49}(k)S(x_1(k)) + w_{410}(k)S(x_2(k)) + w_{411}(k)S(x_3(k)) \\ & + w_{412}(k)S(x_4(k)) \end{aligned} \tag{17}$$

where  $\alpha_1(k)$  is a virtual control for  $x_3(k)$ ,  $V_{mp}(k)$  is a control for  $x_1(k)$ ,  $\alpha_2(k)$  is a virtual control for  $x_4(k)$  and  $V_{my}(k)$  is a control for  $x_2(k)$ . The weights  $w_{ij}(k)$  are updated by (8) and  $S(x)$  is a hyperbolic tangent function given by (5). Simulations results are presented in Section 6.

### 5. Discrete-time block control

The second algorithm for controlling the helicopter model (11) uses a discrete-time decentralized RHONN to identify it. A series parallel structure and a Nonlinear Block Control (NBC) form are used as [22]:

$$\begin{aligned} \chi_i^1(k+1) &= \sum_{p=1}^{L_{1p}} w_{i1p} \prod_{j \in I_{1p}} S_{1p}^{d_{1j}^p} + \sum_{m=L_{1p}+1}^{L'_{1p}} w'_{i1m} \chi_i^1(k), \\ \chi_i^2(k+1) &= \sum_{p=1}^{L_{2p}} w_{i2p} \prod_{j \in I_{2p}} S_{2p}^{d_{2j}^p} + \sum_{m=L_{2p}+1}^{L'_{2p}} w'_{i2m} \chi_i^2(k), \\ &\vdots \\ \chi_i^q(k+1) &= \sum_{p=1}^{L_{qp}} w_{iqp} \prod_{j \in I_{qp}} S_{qp}^{d_{qj}^p} + \sum_{m=L_{qp}+1}^{L'_{qp}} w'_{ipm} \chi_i^r(k), \quad q = 3, \dots, r-1 \\ \chi_i^r(k+1) &= \sum_{p=1}^{L_{rp}} w_{irp} \prod_{j \in I_{rp}} S_{rp}^{d_{rp}^p} + w_i u_i \end{aligned} \tag{18}$$

where  $\chi_i(k)$  are the states of the neural network,  $x_i(k)$  are the states of the plant,  $L_{qp}$  is the number of high order connections,  $L'_i$  is the number of fixed parameters  $w'$ , which depend on the on the plant structure and are incorporated to the neural network model in order to obtain a block controllable structure,  $\{I_{1p}, I_{2p}, \dots, I_{rp}\}$  is a collection of subsets  $\{1p, 2p, \dots, m+n\}$ ,  $w_i$  is the on-line adapted weights for the neural network by a Kalman learning algorithm,  $d_i$  is a non-negative integer which gives the high order connections to the neural network,  $S$  is a vector which has the sigmoid activation functions and is defined as in (3).

It is considered the problem of approximating the  $i$ th state of the nonlinear system (11) by the following  $i$ th discrete-time RHONN in its series-parallel representation [23,25]:

$$\chi_i(k+1) = w_i^{*T} z_i(x(k), u(k)) + \epsilon_{z_i}, \quad i = 1, \dots, n \tag{19}$$

where  $\chi_i(k)$  is the  $i$ th plant state,  $\epsilon_{z_i}$  is a bounded approximation error, which can be reduced by increasing the number of adjustable weights [25]. Assume that there exists an ideal weight vector  $w_i^*$  such that  $\|\epsilon_{z_i}\|$  can be minimized on a compact set  $\Omega_{z_i} \in \mathbb{R}^{L_i}$ . In general, it is assumed that this vector exists and is constant but unknown. The weights  $w_i(k)$  are trained on-line with the Kalman filtering algorithm (8).

**Theorem 2.** *The RHONN (18) trained with the EKF-based algorithm (8), ensures that the identification error (9) is SGUUB; moreover the RHONN weights remain bounded. The proof can be found in [24].*

#### 5.1. RHONN model

In this section, neural models are proposed for this application. We observe that (11) can be represented in the form described in (18), thus a series parallel representation neural model is used. In order to achieve control objectives, for each tracking control we proposed two subsystems  $\Sigma_1$  and  $\Sigma_2$ . The whole tracking control problem is as follows: once subsystems  $\Sigma_1$  and  $\Sigma_2$  have identified the helicopter model (11), then neural sliding mode control laws based on these new subsystems are applied to both helicopter and neural models to track the references. Neural networks are trained on-line with EKF. Based on the mathematical models (11) and (18) we have the next subsystem  $\Sigma_1$

$$\begin{aligned} \chi_1(k+1) &= w_{11}(k)S(x_1(k)) + w_{12}(k)S^2(x_1(k)) \\ &\quad + w_{13}(k) + w_{14}(k)x_2(k) \\ \chi_2(k+1) &= w_{21}(k)S(x_1(k)) + w_{22}(k)S(x_2(k)) \\ &\quad + w_{23}(k)S(x_1(k))S(x_2(k)) + w_{24}(k)V_{mp}(k) \\ y_1(k) &= \chi_1(k) \end{aligned} \tag{20}$$

and the subsystem  $\Sigma_2$  is given by:

$$\begin{aligned} \chi_3(k+1) &= w_{31}(k)S(x_3(k)) + w_{32}(k)S^2(x_3(k)) \\ &\quad + w_{33}(k) + w_{34}(k)x_4(k) \\ \chi_4(k+1) &= w_{41}(k)S(x_3(k)) + w_{42}(k)S(x_4(k)) \\ &\quad + w_{43}(k)S(x_3(k))S(x_4(k)) + w_{44}(k)V_{my}(k) \\ y_2(k) &= \chi_3(k) \end{aligned} \tag{21}$$

where  $\chi_1(k)$  identifies  $x_1(k)$ ,  $\chi_2(k)$  identifies  $x_3(k)$ ,  $\chi_3(k)$  identifies  $x_2(k)$  and  $\chi_4(k)$  identifies  $x_4(k)$ . These subsystems have a NBC form and  $y_1(k)$  is the output for the first subsystem,  $y_2(k)$  is the output for the second subsystem,  $w_{ij}(k)$  are updated by (8), activation functions  $S(x(k))$  are given by (5), the weights  $w_{12}(k)$ ,  $w_{22}(k)$ ,  $w_{34}(k)$  and  $w_{44}(k)$  remain fixed during the entire simulation, by doing so, it is avoided the zero crossing of the weights and loss of controllability.

### 5.2. Control law synthesis

Subsystems  $\Sigma_1$ ,  $\Sigma_2$  and discrete time sliding mode control are used to achieve pitch and yaw position tracking. The first subsystem  $\Sigma_1$  is used for controlling pitch position and the second one  $\Sigma_2$  is used for controlling yaw position. Now, for the pitch-position tracking, we define the following error:

$$z_1(k) = y_1(k) - x_{1ref}(k) \tag{22}$$

where  $x_{1ref}(k)$  is the pitch-position reference. By taking one step ahead in (22), it is obtained:

$$\begin{aligned} z_1(k+1) &= w_{11}(k)S(x_1(k)) + w_{12}(k)S^2(x_1(k)) + w_{13} \\ &\quad + w_{14}x_2(k) - x_{1ref}(k+1) \end{aligned}$$

and  $x_2(k)$  is seen as an input control. Now, if we define  $x_2^d(k)$  as

$$\begin{aligned} x_2^d(k) &= w_{14}^{-1}(-w_{11}(k)S(x_1(k))) - w_{12}S^2(x_1(k)) \\ &\quad - w_{13} + w_{14}^{-1}(x_{1ref}(k+1) + K_1z_1(k)) \end{aligned} \tag{23}$$

which is a desired value for  $x_2(k)$ , we have the following dynamic for  $z_1(k)$

$$z_1(k+1) = K_1z_1(k) \tag{24}$$

where  $K_1$  is a Schur matrix, To ensure  $w_{14}$  has inverse, it is chosen as a constant value. The new error  $z_2(k)$  can be defined as

$$z_2(k) = x_2(k) - x_2^d(k) \tag{25}$$

and taking one step ahead in (25), we have the following equations:

$$\begin{aligned} z_1(k+1) &= K_1z_1(k) + w_{14}(k)z_2(k) \\ z_2(k+1) &= w_{21}(k)S(x_1(k)) + w_{22}(k)S(x_2(k)) \\ &\quad + w_{23}(k)S(x_1(k))S(x_3(k)) + w_{24}V_{mp} - x_2^d(k+1) \end{aligned} \tag{26}$$

where  $V_{mp}$  is the input control and  $\chi_2(k+1) = x_2(k+1) + \delta$ , where  $\delta$  is the identification error which is SGUUB by Theorem 2. Therefore, we use  $\chi_2(k+1)$  like an approximation for  $x_2(k+1)$ . Let us define the manifold like  $S(k) = z_2(k) = 0$ , then (26) has the following structure

$$\begin{aligned} z_1(k+1) &= K_1z_1(k) + w_{14}(k)S(k) \\ S(k+1) &= w_{21}(k)S(x_1(k)) + w_{22}(k)S(x_2(k)) \\ &\quad + w_{23}(k)S(x_1(k))S(x_3(k)) \\ &\quad + w_{24}V_{mp} - x_2^d(k+1) \end{aligned} \tag{27}$$

In order to design a control law such that the tracking is achieved, a discrete sliding mode is implemented as [9]:

$$V_{mp}(k) = \begin{cases} V_{mpeq}(k) & \text{if } |V_{mpeq}(k)| \leq u_{01} \\ u_{01} \frac{V_{mpeq}(k)}{|V_{mpeq}(k)|} & \text{if } |V_{mpeq}(k)| > u_{01} \end{cases} \tag{28}$$

where

$$\begin{aligned} V_{mpeq}(k) &= -w_{24}^{-1}(w_{21}S(x_1(k)) + w_{22}(k)S(x_2(k)) \\ &\quad - w_{23}(k)S(x_1(k))S(x_3(k)) + x_2^d(k+1)) \end{aligned} \tag{29}$$

is calculated from (26) by taking  $S(k+1) = 0$ ,  $u_{01}$  is the control bound. In this control law, if  $|V_{mpeq}| \leq u_{01}$  the control  $V_{mpeq}(k)$  forces the motion through the sliding surface in one step, in the other case, the term  $u_{01} \frac{V_{mpeq}(k)}{|V_{mpeq}(k)|}$  avoids a high value of control by forcing it to decrease until  $|V_{mpeq}(k)| \leq u_{01}$  is satisfied. For a proof of these facts see [12]. Again  $w_{24}$  is fixed to avoid controllability loss. Once the sliding mode is reached, the following dynamic is obtained:

$$z_1(k+1) = K_1z_1(k) \tag{30}$$

where  $K_1$  is a matrix Schur. Therefore the error (22) will converge asymptotically to zero. The following result is useful to ensure the error between the output of the plant and the reference converges to zero [27]:

$$\|y_1(k) - y_r(k)\| \leq \|y_1(k) - y_n(k)\| + \|y_n(k) - y_r(k)\| \tag{31}$$

where  $y_1(k) - y_r(k)$  is the output tracking error between output of the plant and the desired reference.  $y_1(k) - y_n(k)$  is the output identification error between the output of the plant and the neural network, which is reached by the EKF learning algorithm.  $y_n(k) - y_r(k)$  is the output tracking error between the output of the neural network and the desired reference, which is satisfied by the sliding mode control algorithm (28). For yaw-position tracking, we repeat the same steps as before, but in this case, we define the new error:

$$z_3(k) = y_2(k) - x_{2ref}(k) \tag{32}$$

The control law for the second subsystem  $\Sigma_2$  is given by:

$$V_{my}(k) = \begin{cases} V_{mpeq}(k) & \text{if } |V_{mpeq}(k)| \leq u_{01} \\ u_{02} \frac{V_{mpeq}(k)}{|V_{mpeq}(k)|} & \text{if } |V_{mpeq}(k)| > u_{01} \end{cases} \tag{33}$$

where

$$\begin{aligned} V_{mpeq}(k) &= -w_{44}^{-1}(w_{41}S(x_3(k)) + w_{42}(k)S(x_4(k)) \\ &\quad + w_{43}(k)S(x_3(k))S(x_4(k)) + x_4^d(k+1)) \end{aligned} \tag{34}$$

and  $u_{02}$  is a bounded for this control law, and  $x_4^d$  is given by

$$\begin{aligned} x_4^d(k) &= w_{34}^{-1}(-w_{31}(k)S(x_3(k))) - w_{32}S^2(x_3(k)) \\ &\quad + w_{34}^{-1}(-w_{33} + x_{2ref}(k+1) + K_2z_3(k)) \end{aligned} \tag{35}$$

where  $K_2$  is Schur. Next section presents simulation results to this control problem.

## 6. Simulation results

For simulation purposes, we consider the parameters listed in Table 1 [17]. Notice that although all states of the plant must be accessible for implementation of control laws (16)–(17), (28) and (33) it is not necessary to know the plant parameters neither its model. We suggest a square reference  $y_{1ref}(k)$  for the pitch axis and a sine reference  $y_{2ref}(k) = 0.6 \sin((\pi/5)kT)$  rad/s for the yaw axis.

### 6.1. Backstepping results

The backstepping strategy is portrayed in Fig. 2. We observe that neural controller is updated by the EKF based on the tracking error. Then, the controller generates a control law applied to the system. Notice that the plant model is not needed for the control design.

The output tracking performance of the plant and reference signals are shown in Figs. 3 and 4. The neural backstepping control law

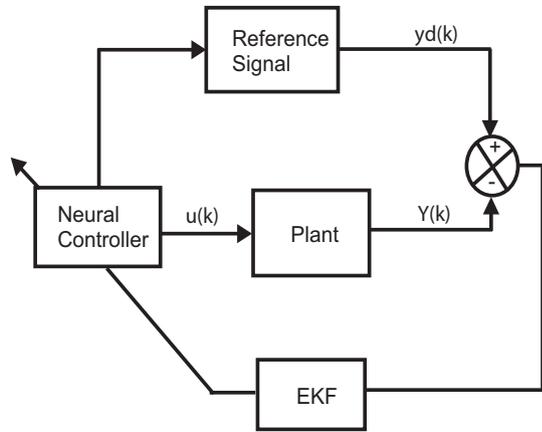


Fig. 2. Backstepping control law scheme.

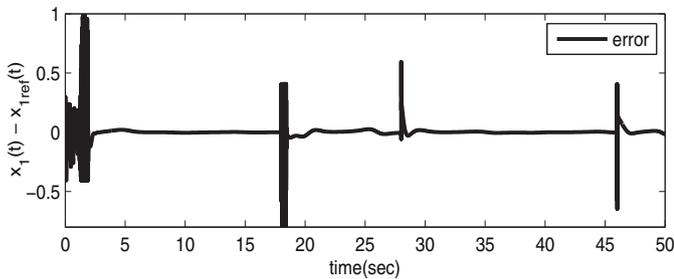
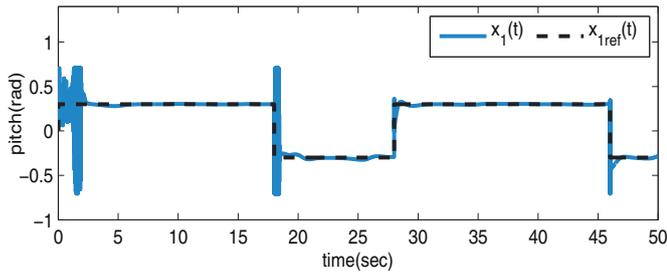


Fig. 3. (Above) Graphs of the pitch position  $x_1(k)$  (blue line) and its reference  $x_{1ref}(k)$  (black line). (Below) Tracking error. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of the article.)

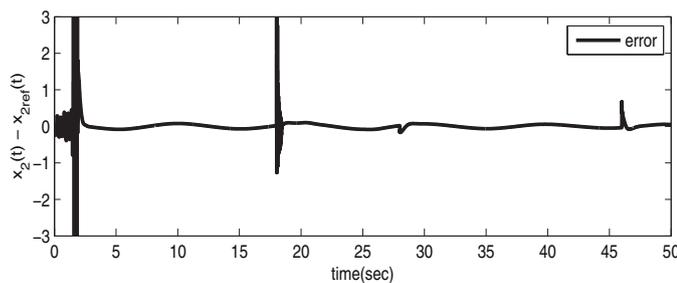
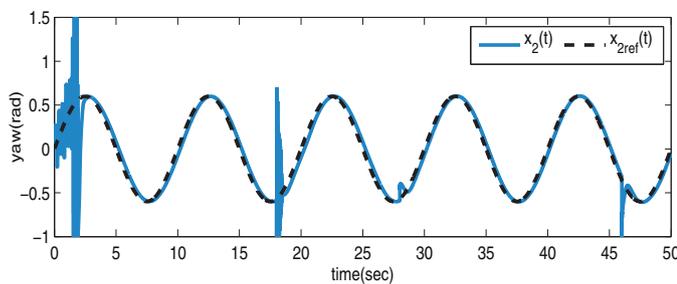


Fig. 4. (Above) Graphs of the yaw position  $x_2(k)$  (blue line) and its reference  $x_{2ref}(k)$  (black line). (Below) Tracking error. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of the article.)

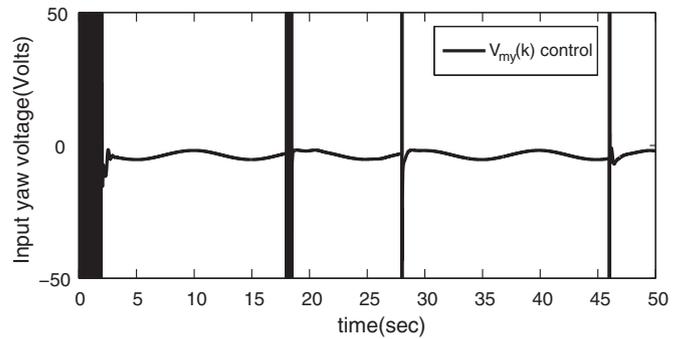
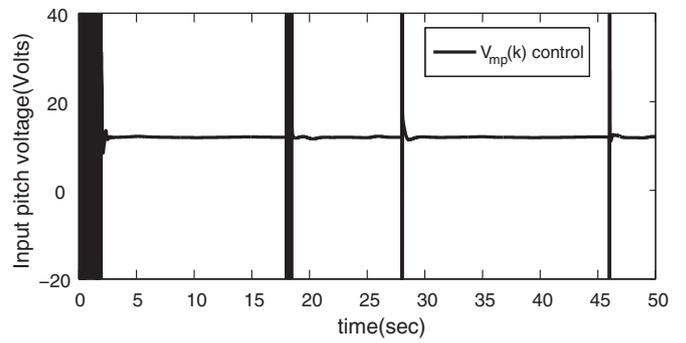


Fig. 5. Backstepping control law. (Above) Pitch voltage. (Below) Yaw voltage.

(16)–(17) applied to the helicopter model (11) shows a good tracking for both pitch and yaw positions even when the plant is treated as two separated independent subsystems and the references are of square and sine type. The small transitory for the pitch and yaw position can be explained for the time necessary to identify adequately the plant. Tracking error results reveal that backstepping strategy is able to maintained the reference error bounded for pitch and yaw positions.

Control laws  $V_{mp}$  and  $V_{my}$  applied to the helicopter model are displayed in Fig. 5. Due to the discrete (square) type reference for the pitch position, sudden peaks of input voltages are required during the switching of this reference signal. After the transient response for pitch and yaw position has passed, the reference signals are well followed and the control inputs are reduced to acceptable values. Voltage requirements are reasonable for the motor, that is the reason for the good trajectory tracking.

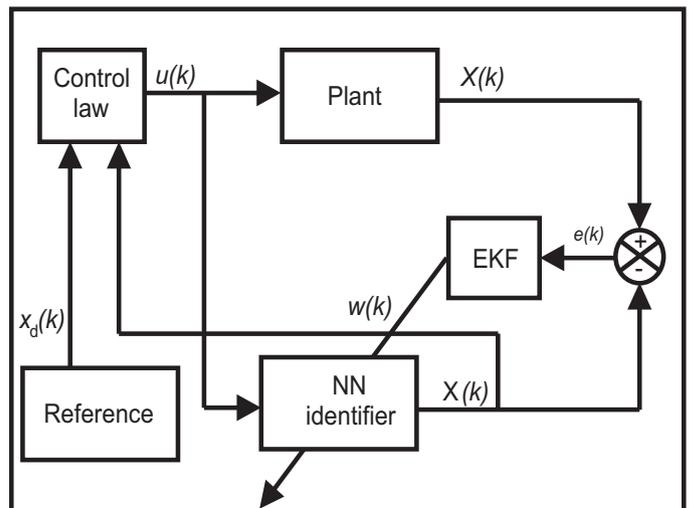
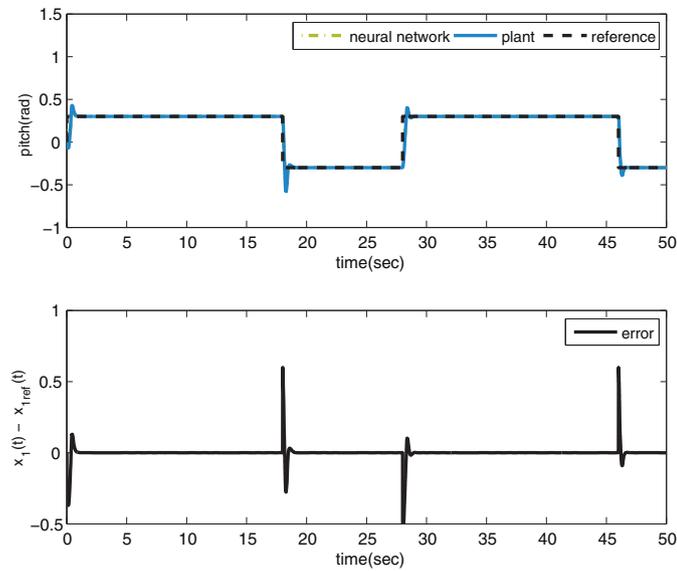


Fig. 6. Block control law scheme.

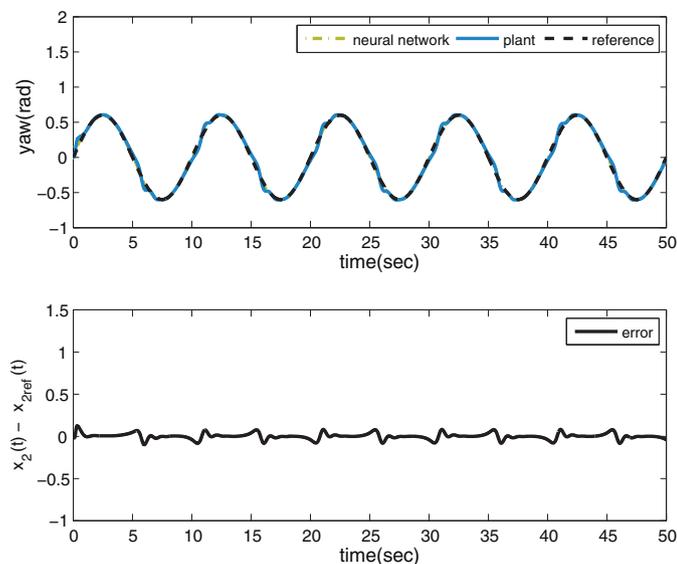


**Fig. 7.** (Above) Graphs of the pitch position  $x_1(k)$  (blue line), its reference  $x_{1ref}(k)$  (black line) and the neural network identification  $\hat{x}_1(k)$  (yellow line). (Below) Tracking error between pitch and reference signals. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of the article.)

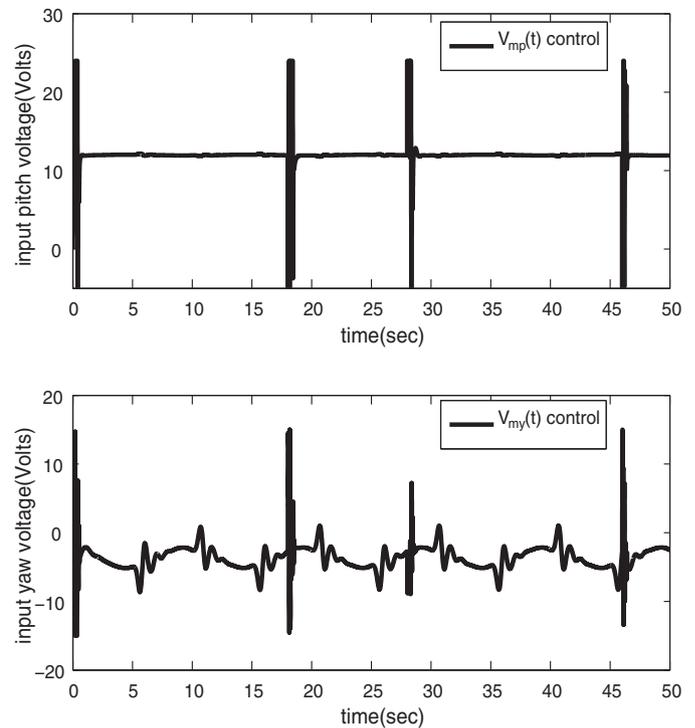
## 6.2. Block control results

The identification and block control scheme is presented in Fig. 6. Block control is more complex than the backstepping one because it uses a neural model which has an approximated dynamic of the plant. The weights of the RHONN are updated every  $k$  steps by an EKF, the filter is based on the error between the states of the plant and the estimated ones.

Figs. 7 and 8 show the output  $y(k)$  of the helicopter model and the references to be tracked along with their tracking errors. We observe a good performance for tracking the reference signals. In comparison with the backstepping control, reference signals for both pitch and yaw positions are better followed by the block control strategy because it receives more information than



**Fig. 8.** (Above) Graphs of the pitch position  $x_2(k)$  (blue line), its reference  $x_{2ref}(k)$  (black line) and the neural network identification  $\hat{x}_2(k)$  (yellow line). (Below) Tracking error between pitch and reference signals. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of the article.)



**Fig. 9.** Block control law. (Above) Pitch voltage. (Below) Yaw voltage.

backstepping does. Tracking errors exhibit how the trajectories are achieved with a small bounded deviation.

The control laws (28) and (33) applied to the helicopter model can be seen in Fig. 9. Even though the plant model is divided in two independent subsystems and one control law is designed for each subsystem, each helicopter output  $y_1(t), y_2(t)$  follows its own reference. Therefore, one control law applied to one subsystem has no effect to the other one; i.e. one output signal follows a reference without taking into account the dynamics produced by the other control law. Voltage requirements are less demanding than those require for the backstepping. Pitch and yaw voltages are reasonable for the motor, that is neural block control strategy could provide a better implementation for the Quanser 2-DOF helicopter.

## 7. Conclusion

This paper has presented an important application of neural networks for controlling a 2-DOF helicopter model. Neural backstepping and neural sliding mode block control techniques have been considered. Although both algorithms have exhibited good performances, block control algorithm has presented a better behaviour. The reason is that the block control receives more information than backstepping control does. We remark that neural networks can be used for identification of the system and for approximating virtual and practical controls without previous knowledge of the system neither the interconnection terms. Training of this type on neural networks is done on-line. Numerical results show that neural networks are able to track not only continuous but also discontinuous references.

## Acknowledgement

The authors thank to Conacyt Mexico.

## References

- [1] R. Lozano, *Unmanned Aerial Vehicles Embedded Control*, ISTE Ltd. and John Wiley & Sons Inc., London, Great Britain, 2010.
- [2] G.R. Yu, H.T. Liu, Sliding mode control of a two-degree-of-freedom helicopter via linear quadratic regulator, *IEEE International Conference on Systems, Man and Cybernetics 4* (2005) 3299–3304.
- [3] P. Castillo Garcia, R. Lozano, A.E. Dzul, *Modelling and Control of Mini-Flying Machines*, Springer-Verlag, England, 2005.
- [4] Z.Y. Sheng, Robust stabilization and disturbance attenuation of a class of mimo nonlinear systems with multi-operation points, in: *Proceedings of the 26th Chinese Control Conference*, Hunan, China, 2007, pp. 700–704.
- [5] J. Velagic, N. Osmic, Identification and control of 2DOF nonlinear helicopter model using intelligent methods, in: *IEEE International Conference on Systems Man and Cybernetics (SMC)*, Istanbul, 2010, pp. 2267–2275.
- [6] S. Ge, J. Zhang, T.H. Lee, Adaptive neural network control for a class of MIMO nonlinear systems with disturbances in discrete-time, *IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics 34* (2004) 1630–1645.
- [7] E.N. Sanchez, L.J. Ricalde, Trajectory tracking via adaptive recurrent neural control with input saturation, *Proceeding of the International Joint Conference on Neural Networks 1* (2003) 359–364.
- [8] A.Y. Alanis, E.N. Sanchez, A.G. Loukianov, Discrete-time adaptive backstepping nonlinear control via high-order neural networks, *IEEE Transactions on Neural Networks 18* (2007) 1185–1195.
- [9] F.L. Lewis, S. Jagannathan, A. Yesildirak, *Neural Network Control of Robot Manipulators and Nonlinear Systems*, Taylor & Francis, New York, 1999.
- [10] R.J. Williams, D. Zipser, A learning algorithm for continually running fully recurrent neural networks, *Neural Computation 1* (1989) 270–280.
- [11] R. Grover Brown, P.Y.C. Hwang, *Introduction to Random Signals and Applied Kalman Filtering*, John Wiley & Sons Inc., New York, 1996.
- [12] Y. Song, J.W. Grizzle, The extended Kalman filter as local asymptotic observer for discrete-time nonlinear systems, *Journal of Mathematical Systems, Estimation and Control 5* (1995) 59–78.
- [13] E.A. Hernandez-Vargas, A. Alanis, E.N. Sanchez, M. Hernandez-Gonzalez, Functional fuzzy supervisor applied to a wastewater treatment plant based on a neural observer, *Ingenieria Quimica 36* (2009) 14–22.
- [14] L.A. Feldkamp, D.V. Prokhorov, T.M. Feldkamp, Simple and conditioned adaptive behavior from Kalman filter trained recurrent networks, *Neural Networks 16* (2003) 683–689.
- [15] H. Rios, A. Rosales, A. Davila, Global non-homogeneous quasi-continuous controller for a 3-DOF helicopter, in: *11th International Workshop on Variable Structure Systems*, 2010.
- [16] V. Utkin, J. Guldner, J. Shi, *Sliding Mode Control in Electromechanical Systems*, Taylor & Francis, Philadelphia, USA, 1999.
- [17] Quanser 2-DOF Helicopter, User and Control Manual, 2006.
- [18] N. Kazantzis, C. Kravaris, Time-discretization of nonlinear control systems via Taylor methods, *Computer and Chemical Engineering 23* (1999) 763–784.
- [21] Y. Li, S. Qiang, X. Zhuang, O. Kaynak, Robust and adaptive backstepping control for nonlinear systems using RBF neural networks, *IEEE Transactions on Neural Networks 15* (2004) 693–701.
- [22] V.H. Benitez, E.N. Sanchez, A.G. Loukianov, Decentralized adaptive recurrent neural control structure, *Artificial Intelligence 20* (2007) 1125–1132.
- [23] E.B. Kosmatopoulos, M.M. Polycarpou, M.A. Christodoulou, P.A. Ioannou, High-order neural network structures for identification of dynamical systems, *Neural Networks 6* (1995) 422–431.
- [24] A.Y. Alanis, E.N. Sanchez, A.G. Loukianov, Discrete-time recurrent neural induction motor control using Kalman learning, in: *Neural Networks, Vancouver, 2006*.
- [25] G.A. Rovithakis, M.A. Chistodoulou, *Adaptive Control with Recurrent High-Order Neural Networks*, Springer Verlag, Berlin, Germany, 2000.
- [27] R.A. Felix, E.N. Sanchez, A.G. Loukianov, Avoiding controller singularities in adaptive recurrent neural network, in: *Proceedings of the IFAC'05, Prague, 1996*.
- [28] V.H. Benitez, E.N. Sanchez, A.G. Loukianov, Decentralized adaptive recurrent neural control structure, *Engineering Applications of Artificial Intelligence 20* (2007).