Decentralized Sliding Mode Controller Based on Genetic algorithm and a Hybrid approach for Interconnected Uncertain Nonlinear Systems

Allouani Fouad[1], Djamel Boukhetela[2] and Fares Boudjema[2]

[1]*Department of Electrical Engering, University of Khenchela, Algeria*
[2]*Laboratoire de commande des processus, Automatic Control Department, Ecole Nationale Polytechnique, Algiers, Algeria.*

allouani_fouad@yahoo.fr, djamel.boukhetala@enp.edu.dz, fboudjema@yahoo.fr

Abstract

A new type controller, recurrent fuzzy neural networks-fuzzy-sliding mode controller (FRNN-FSMC), is developed for a class of large-scale systems with unknown bounds of high-order interconnections and disturbances. The main purpose is to eliminate the chattering phenomenon and to overcome the problem of the equivalent control computation. The FRNN-FSMC, which incorporates the recurrent fuzzy neural network (RFNN), fuzzy logic controller (FLC) and the SMC, can eliminate chattering using a fixed boundary layer around the switch surface. Within the boundary layer, where the FLC is applied, the chattering phenomenon, which is inherent in a SMC, is avoided by smoothing the switch signal. Moreover, to compute the equivalent controller, a feed-forward RFNN is used. The stability of the whole system is analyzed via the Lyapunov methodology. In this study, we propose an effective method to select some key controller parameters in an optimal manner by using the genetic algorithm (GA), so that a high performance of the overall system's response can be achieved. The effectiveness and efficiency of the proposed controller and optimization method were tested using highly interconnected nonlinear systems as examples.

Keywords: *Interconnected uncertain nonlinear systems, Genetic algorithm, Recurrent Fuzzy neural networks,Sliding mode*

1. Introduction

Interconnected large-scale nonlinear systems can be found in many real-life practical applications, such as power systems, transportation systems, manufacturing processes, communication networks and economic systems. Due to the complexity of dynamic models and the high number of variables involved in the control of these systems, design and implementation of the centralized controller are generally not evident. Therefore, to overcome such problems, a decentralized structure has been proposed. In this structure as in [27], the control system is decomposed into a number of interconnected subsystems; for each one, a local sub-controller is designed independently using only local subsystem's state information. The obtained decentralized controllers are also reliable in the sense that when some local controllers are out of order, the rest of the system can still be in operation [10].

Variable structure control (VSC) via its main mode of operation, sliding mode control (SMC) is recognized to be an effective way for controlling both certain and uncertain, linear and nonlinear systems because the technique not only stabilizes the system, but also provides disturbance rejection and low sensitivity to plant parameter variations [32].

This inspires many automation engineers to use sliding mode methodology in the control of the large-scale nonlinear systems. Indeed, in [2] a design procedure based on a combination of the model coordination concept and a variable structure methodology was proposed. This adaptive scheme achieves asymptotic exactly model reference tracking. However, this control method suffers from the chattering problem because the control actions are discontinuous. In [10], Boukhetala and colleagues have proposed a novel decentralized VSC scheme for multi-joint robot manipulators with a new class of first-order nonlinear sliding surfaces in which the structure is easily obtained by choosing an appropriate nonlinear function. In [34], authors have proposed a decentralized output feedback control scheme using sliding mode techniques, in which class of nonlinear large-scale interconnected systems is considered, where uncertainties and interconnections are considered matched and mismatched. However, the scheme proposed in this work requires that all nominal isolated subsystems are minimum phase; this restricts the application of this method. Moreover, two approaches of decentralized SMC for uncertain interconnected systems are presented in [8]; the first one exploits the use of the common sliding mode strategy of design to simplify the structure of the controller and to give useful results after the reaching phase, while the second takes advantage of a particular sliding surface to give good results starting from any initial conditions. Benitez and Sanchez in [31] have designed a variable structure neural control for the class of block-controllable nonlinear systems. However, this control implementation requires the full local state information.

Nevertheless, in practical decentralized control applications, the SMC suffers from three main disadvantages. The first one is the chattering, which is the high frequency oscillations of the controller output. The second one is the knowledge of bounds of uncertainties and disturbances are required in SMC design. The third one is the equivalent control calculation complexities.

Chattering is generally highly undesirable in practice. This harmful phenomenon is caused by two effects which are commonly conceived [18].The first one is the presence of parasitic dynamics in series with the control systems causes a small-amplitude high-frequency oscillation. The second one is the switching non-idealities which causes high-frequency oscillations. These effects may include short time delays due to sampling (*e.g.*, zero-order hold (ZOH)), and/or additional execution time required for calculation of control, and more recently, transmission delays in networked control systems. Also, it is difficult to estimate the bounds of interconnections and disturbances of the real systems, especially in interconnected uncertain systems where the knowledge of interactions between the subsystems is severely limited.

In general, the commonly used methods to eliminate/or reduce the chattering, are: the use of continuous approximation method in which the sign function in SMC control law [32] is replaced by a continuous approximation [14], the boundary layer of width technique [16], and integral sliding control [29]. The boundary layer of width approach was introduced to eliminate the chattering around the switching surface and the control discontinuity within this thin boundary layer. However, in this method the discontinuous sign term in the SMC is not eliminated or smoothed indeed and it is difficult to make choice between the chattering and the robustness by tuning the width. For the continuous approximation approach it gives rise to a high-gain control when the states are in the close neighborhood of the switching manifold.

The strength of fuzzy systems based on fuzzy logic (FL) has been taken advantage in SMC design due to its simple representation, underpinned by the heuristic nature of human reasoning. One of rationale central to the use of fuzzy systems with SMC is to alleviate the problems associated with chattering. Among the earliest works seen in the literature the work presented in [36], which uses FL in a low-pass filter to “smooth” SMC signals, which would

reduce chattering. This is done by forming a specifically constructed fuzzy rule table. In general, the main approach to dealing with chattering using FL is to fuzzify SMC. There have been extensive works in this area [23, 25, 26]. Moreover, to deal with uncertainties, which are also another cause of chattering, fuzzy system architecture is proposed in [4] to adaptively compensate the nonlinearities and the uncertainties. In [3], two methods are proposed to approximate the unknown system functions in order to reduce the magnitudes of the switching controls. Authors in [9] have proposed an adaptive fuzzy SMC for linear systems to learn the equivalent control and two switching controls without a priori knowledge of the bounds of the uncertainties. Indirect adaptive fuzzy sliding mode control scheme is proposed in [17] for a class of nonlinear systems in which both equivalent control term and switching-type control term in the SMC law are approximated by fuzzy systems.

On the other hand, one way of avoiding the computational burden involved in the calculation of the equivalent control is the use of an estimation technique as suggested in [21]. Generally, NNs offer a "model-free" mechanism to learn from examples the underlying dynamics. One of the early attempts in using NNs in SMC [13] aims to estimate the equivalent control. The same approach is also contemplated in [19]. In the literature, there are several research works in this area. For example, in [6], two parallel layer feed-forward NNs are utilized to estimate the equivalent control and the magnitude of the switching control component using the backpropagation algorithm (BPA). This approach is proved to be effective in eliminating chattering and practically implemented on a seasaw system. Authors in [22] used also two parallel NNs to achieve the same objective as in [6]. However, the corrective control computation of NNs is more complicated than presented in [6]. Moreover, recurrent neural networks (RNNs) have also been considered to be a good candidate for incorporation into SMC structures because of its feedback mechanism which improves convergence. For example, in [35], a special RNN is constructed to approximate the equivalent control in order to reduce chattering, together with a special learning algorithm for updating the parameters to be learnt.

RFNN [5, 33], naturally involves dynamic elements in the form of feedback connections used as internal memories. Thus, the RFNN is a dynamic mapping and demonstrates good control performance in the presence of uncertainties, which usually composed of unpredictable plant parameter variations, external force disturbance, unmodelled and nonlinear dynamics, in practical applications of the dynamic systems compared to the feedforward NNs, RNNs and FNN.

In this paper, the design of a new decentralized SMC for a class of interconnected large-scale nonlinear systems using an approach of hybrid control based on the combination of RFNN, FLC and SMC methodology has been proposed. At first, we develop a linear sliding surface for each subsystem and we discuss globally the SMC existing condition. In order to eliminate the chattering problem and compensate completely the nonlinearities and the uncertainties in each sub-RFNN-FSMC, a single-input single-output (SISO) Mamdani FLC is used to approximate the SMC switching-type control term. In the procedure, a fixed boundary layer is adopted and FLC is applied within the boundary. Outside the boundary, the SMC is applied to drive the system states into the boundary layer. In addition, feedforward RFNN is used to approximate the equivalent control term. The link weights of the feedback unit, the link weights of the fourth network layer, the center and the width of the network Gaussian membership functions are updated using the supervised gradient descent method such that the corrective control term in the SMC goes to zero. Mathematical proof for the stability and convergence of the system is also presented.

Moreover, because of the proposed controller has a lot of parameters to be calculated, in this step of design, it is interesting to look for an appropriate way to adjust these parameters in

an optimal manner. To this end, Genetic Algorithms (GAs) [11] turn out in the last twenty years, to be one of the most frequently employed biologically inspired optimization algorithm for solving complex optimization problems are used in this study.

Finally, two highly interconnected nonlinear systems which are: the two-link robotic manipulator [30] and the two inverted pendulums on carts system [37, 38] are used to test the performance and effectiveness of the proposed control method using a simulation approach.

2. Problem Statement

Consider a large-scale nonlinear system S comprised of N interconnected subsystems s_i defined by:

$$s_i \begin{cases} \dot{x}_{i,1} = x_{i,2} \\ \dot{x}_{i,2} = x_{i,3} \\ \quad \dots \dots \\ \dot{x}_{i,n_i} = f_i(\bar{x}_i) + g_i(\bar{x}_i)u_i(t) + z_i(\bar{x}_s) + d_i(\bar{x}_i) \end{cases} \tag{1}$$

Where $\bar{x}_i = \left[x_{i,1} \dots x_{i,n_i}\right]^T \in R^{n_i}$, $\bar{x}_s = [\bar{x}_1 \dots \bar{x}_N]^T \in R^N$, $n = \sum_{i=1}^{N} n_i$ are the ith subsystem's state vector and the state vector of the global system S, $u_i(t) \in R$ is control input of s_i. The function $f_i(\bar{x}_i)$ represents unknown continuous function, $g_i(\bar{x}_i) \in R^{n_i}$, $z_i(\bar{x}_s)$: $R^n \to R$ and $d_i(\bar{x}_i)$ represent respectively: unknown control gain function, strength of interconnections from other subsystems and uncertainties and the disturbances of the subsystem s_i.

The following assumption is made about the structure of the system given in (1).

Assumption 1. Each $\bar{x}_i, i \in N$, is measurable,

Assumption 2. For $t \in [0, +\infty]$, the desired output x_{id} and it's up to n_i th-order derivatives $\left[x_{id}^{(1)}, \dots, x_{id}^{(n)}\right]$ are bounded, i.e., $\left|x_{jd}^{(i)}\right| \le h_j, i = 1, \dots, n_i, j = 1, \dots, N$, where h_j is a positive constant.

Assumption 3. $\forall \bar{x}_i \in R^{n_i}, \exists k_i > 0$, there exists $|g_i(\bar{x}_i)| \ge k_i > 0, i = 1, .., N$.

Under assumptions (1-3) given above the problem in this paper is to find decentralized robust control schemes for the system (1) so that the state $\bar{x}_i = \left[x_{i,1}, \dots, x_{i,n_i}\right]^T$ can track stably the desired state $\bar{x}_{id} = \left[x_{id}, \dots, x_{id}^{(n_i-1)}\right]^T$, $i = 1, \dots, N$, and the tracking error $E_i = \bar{x}_i - \bar{x}_{id} = \left[e_{i,1}, .., e_{i,n_i}\right]^T, i = 1, \dots, N$, can converge to zero.

3. Design of the Decentralized RFNN-FSMC

3.1. Design of SMC

For the subsystem s_i, define the sliding surface $\sigma_i(E_i)$ in the state space R^{n_i}:

$$\sigma_i(E_i) = C_{i,1}e_{i,1} + C_{i,2}e_{i,2} + \dots + C_{i,(n_i-1)}e_{i,(n_i-1)} + e_{i,n_i},\ i = 1, \dots, N \tag{2}$$

Where $e_{i,1} = x_{i,1} - x_{id}, e_{i,2} = \dot{x}_{i,1} - \dot{x}_{id}, \dots ., e_{i,n_i} = x_{i,1}^{(n_i-1)} - x_{id}^{(n_i-1)}$

The coefficients, $C_{i,1}, C_{i,2}, \ldots, C_{i,(n_i-1)}$ are constants and the polynomial $\lambda^{n_i-1} + C_{i,(n_i-1)}\lambda^{n_i-2} +, \ldots, +C_{i,1}$ is Hurwitzian. The derivative of (2) can be calculated as follows:

$$\begin{aligned}\dot{\sigma}_i &= \sum_{j=1}^{n_i-1} C_{i,j}\, e_{i,(j+1)} + \dot{e}_{i,n_i} \\ &= \sum_{j=1}^{n_i-1} C_{i,j}\, e_{i,(j+1)} + f_i(\bar{x}_i) + g_i(\bar{x}_i)u_i(t) + z_i(\bar{x}_s) + d_i(\bar{x}_i) - x_{id}^{(n_i)} \\ &= u_i^* + \Delta u_i^* + g_i(\bar{x}_i)u_i(t) \end{aligned} \quad (3)$$

Where $u_i^* = \sum_{j=1}^{n_i-1} C_{i,j}\, e_{i,(j+1)} - x_{id}^{(n_i)}$, and $\Delta u_i^* = f_i(\bar{x}_i) + z_i(\bar{x}_s) + d_i(\bar{x}_i)$.

Generally, in SMC design, the following assumptions are needed:

Assumption 4. $|f_i(\bar{x}_i, t)| \le F_i(\bar{x}_i), i = 1, \ldots, N$, where F_i is a nonnegative function,

Assumption 5. $|d_i(\bar{x}_i, t)| \le D_i(\bar{x}_i), i = 1, \ldots, N$, where D_i is a nonnegative function,

Assumption 6. The interconnections are bounded by a *p*th-order polynomial in states, *i.e.*, there exists nonnegative numbers ξ_{ij}^k, such that $|z_i(\bar{x}_s)| \le \sum_{k=0}^{P}\sum_{j=1}^{N} \xi_{ij}^k \left\|\bar{x}_{s_j}\right\|^k$.In this case, vector norms are taken to be Euclidean. In general, the control law of the SMC can be taken as:

$$u_i = \frac{1}{g_i}(-u_i^* - \Delta u_i') \quad (4)$$

Where $\Delta u_i' = \left(F_i + \sum_{k=0}^{P}\sum_{j=1}^{N} \xi_{ij}^k \left\|\bar{x}_{s_j}\right\|^k + D_i + \psi_i\right) sgn(\sigma_i)$ and ψ_i is a positive constant, $sgn(\sigma_i)$ is the sign function. So, we can see easily, that the sliding mode condition is satisfied.

Remark 1: Assumptions 4–6 give the bounds of these functions since these bounds are needed in the existing SMC design.

Remark 2: From (4), we observe that the discontinuous part (sign term) of the control law causes the chattering problem.

Remark 3: The coefficients $C_{i,1}, C_{i,2}, \ldots, C_{i,(n_i-1)}$ are determined using GA.

3.2. Structure of the Proposed Decentralized RFNN-FSMC

The global decentralized controller includes *N* sub-RFNN-FSMC each of them comprises of two parts, the first one computes the corrective term in the SMC using a FLC as an approximation mechanism such as in [1], and the second one calculates the equivalent control term in SMC by a RFNN. The corrective control term generated by the FLC is summed with the output of RFNN to construct the control signal. The corrective control is used as a measure of the error to update the weights of the RFNN. The proposed adaptation scheme directly results in chattering-free control action for the corrective control. Furthermore, the GA is used to fined for each sub-RFNN-FSMC the following parameters: the learning-rate parameter of the BPA used to establish the multilayer feed-forward network, the switching surface coefficients and the membership functions parameters of the FLC, to improve firstly the RFNN leaning ability and increase the speed of network convergence, to enhance the rate of convergence on the sliding surface and to increase the approximation mechanism ability of the FLC. The overall system with the proposed controller and the internal configuration of each sub-RFNN-FSMC are illustrated in Figure 1(a) and Figure 1(b).

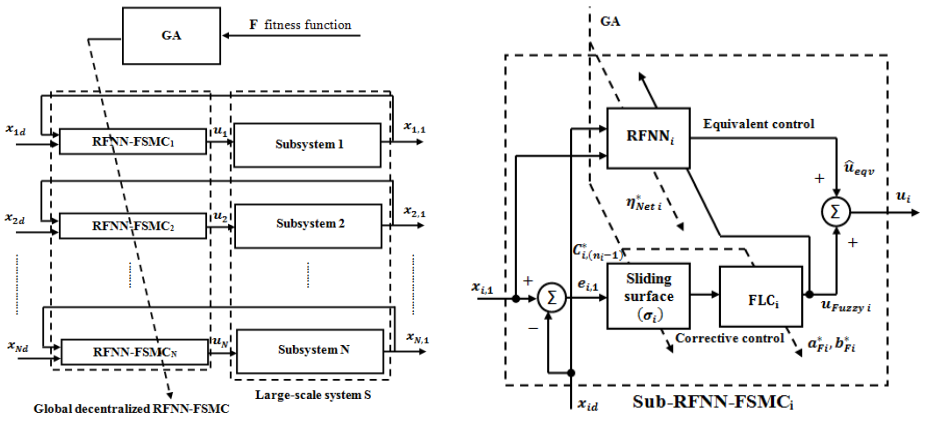


Figure 1. The Overall Structure of the RFNN-FSMC and the Internal Configuration of each sub-RFNN-FSMC

3.3. Computation of the Equivalent Control

The structure of the RFNN adopted in this paper to compute the equivalent control term in each sub-RFNN-FSMC is shown in Figure 2, it comprises: the input layer associated to n input variables, membership layer composed of m nodes for each input variable, rule layer with l nodes and output layer with p output nodes. The term z^{-1} as shown in Figure 2 represents a time delay.

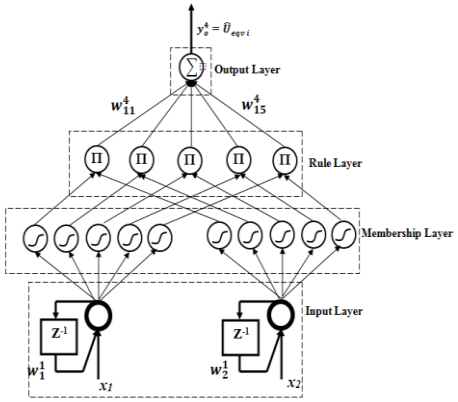


Figure 2. Structure of Four-layer RFNN used to Approximate the Equivalent Control

The signal propagation and the basic function in each layer of the RFNN are introduced as follows:

Layer 1 (Input Layer): The nodes in this layer only transmit input values to the next layer. So, feedback connections are added in this layer to embed temporal relations in the network. For ith node in this layer, the input and output are represented as:

$$net_i^1(k) = x_i^1(k) + w_i^1 y_i^1(k-1), y_i^1(k) = net_i^1(k), i = 1,2. \quad (5)$$

Where x_i^1 represents the ith input to the node of layer1, k is the number of iterations, w_i^1 is the recurrent weights, $y_i^1(k)$ is the output of the ith node in layer1.

Layer 2 (Membership Layer): In this layer, each node performs a membership function. The Gaussian function is adopted as the membership function.

$$net_{ij}^{2}(k) = \frac{\left(x_i^2 - a_{ij}\right)^2}{2\left(b_{ij}\right)^2}, y_{ij}^{2} = exp\left(net_{ij}^{2}(k)\right), i = 1,2,\ j = 1,2 \ldots, m = 5 \quad (6)$$

Where a_{ij} and b_{ij} are the center and the width of the Gaussian membership function; the subscript ij indicates the jth term of the ith input linguistic variable x_i^2 to the node of layer 2, m is the total number of the linguistic variables with respect to the input nodes.

Layer 3 (Rule Layer): This layer forms the fuzzy rules base and realizes the fuzzy inference. Each node in this layer is corresponding to a fuzzy rule. The qth fuzzy rule can be described as:

$$qth \text{ rule: } if\ x_1\ is\ A_1^q, \ldots\ldots, x_n\ is\ A_n^q\ , then\ y_1\ is\ B_1^q, \ldots\ldots, y_p\ is\ B_p^q \quad (7)$$

Where A_i^q is the term of the ith input in the qth rule and B_j^q is the term of the jth output in the qth rule. Then, the qth node of layer 3 performs the AND operation in qth rule. Using $y_{iq_i}^2$ to denote the membership of x_i to A_i^q, where $q_i \in \{1,2, \ldots\ldots, m\}$, then the input and output of qth node can be described as:

$$net_q^3(k) = \prod y_{iq_i}^2, y_q^3 = net_q^3(k), q = 1, \ldots\ldots, l = 5. \quad (8)$$

Layer 4 (Output Layer): Nodes in this layer performs the defuzzification operation. The input and output of oth node can be calculated by:

$$net_o^4(k) = \sum w_{oq}^4 \times y_q^3, y_o^4(k) = net_o^4(k), o = 1, \ldots\ldots, p = 1. \quad (9)$$

Where w_{oq}^4 is the weight, which represents the output action strength of the oth output associated with the qth rule.

In general RFNN, the BPA uses the gradient descent method to establish the multilayer feed-forward network. The training processes use iterative gradient algorithms to minimize the mean square error between the actual output and the desired output, i.e., to minimize the cost function selected as the difference between the desired and the estimated equivalent controls. Therefore, a simple cost function is used as an update formula as follows [6]:

$$E_{eqv}(k) = \frac{1}{2}\left[u_{eqv}(k) - \hat{u}_{eqv}(k)\right]^2 \quad (10)$$

By using the BPA, the weights of the net are adjusted such that the cost function defined in (10) is minimized. The BPA may be written briefly as:

$$W_{Net}(k+1) = W_{Net}(k) + \Delta W_{Net}(k) = W_{Net}(k) + \eta_{Net}\left(-\frac{\partial E_{eqv}(k)}{\partial W_{Net}(k)}\right) \quad (11)$$

Where η_{Net} is a constant that denotes the learning rate and W_{Net} represents the tuning weights, in this case, which are w_i^1, a_{ij}, b_{ij}, and w_{oq}^4. Subscript Net represents RFNN.

By recursive applications of the chain rule, the error term for each layer is first calculated, then the parameters in the corresponding layers are adjusted. With (9) and the cost function defined in (10), derive the update rule of w_{oq}^4:

$$w_{oq}^4(k+1) = w_{oq}^4(k) - \eta_{Net}(u_{eqv}(k) - \hat{u}_{eqv}(k))y_q^3 \quad (12)$$

Similarly, the update laws of a_{ij}, b_{ij}, w_i^1 are:

$$a_{ij}(k+1) = a_{ij}(k) - \eta_{Net}(u_{eqv}(k) - \hat{u}_{eqv}(k))w_i^1 y_q^3 \frac{y_i^1 - a_{ij}}{(b_{ij})^2} \quad (13)$$

$$b_{ij}(k+1) = b_{ij}(k) - \eta_{Net}(u_{eqv}(k) - \hat{u}_{eqv}(k))w_i^1 y_q^3 \frac{(y_i^1 - a_{ij})^2}{(b_{ij})^3} \quad (14)$$

$$w_i^1(k+1) = w_i^1(k) + \eta_{Net}\left(u_{eqv}(k) - \hat{u}_{eqv}(k)\right) w_i^1 y_q^3 \frac{y_i^1 - a_{ij}}{(b_{ij})^2} y_i^1(k-1) \quad (15)$$

Equations (12), (13), (14) and (15) contain the actual equivalent term which is unknown in reality. Hence, these equations cannot be calculated. In order to overcome this problem, the value of corrective control u_{Fuzzy} is utilized to replace $(u_{eqv}(k) - \hat{u}_{eqv}(k))$.The reason is that the characteristics of corrective control and $(u_{eqv}(k) - \hat{u}_{eqv}(k))$ are similar [6].

Remark 4: The learning rate parameter η_{Net} is determined using GA takes into account the relation (23) given in section 4.

3.4. Computation of the Corrective Control

The controller in (4) exhibits high frequency oscillations in its output, causing a problem known as the chattering phenomena. Chattering is highly undesirable because it can excite the high frequency dynamics of the system. For its elimination, a continuous FLC is used to approximate the corrective term in SMC. The design of the fuzzy controller begins with extending the crisp sliding surface $\sigma = 0$ to the fuzzy sliding surface defined by a linguistic expression [28]:

$$\tilde{\sigma} is\ ZERO \quad (16)$$

Where $\tilde{\sigma}$ is the linguistic variable for σ and $ZERO$ is one of its fuzzy set. In order to partition the universe of discourse of σ, the following fuzzy sets of σ are introduced:

$$T(\tilde{\sigma}) = \{NB, NM, ZR, PM, PB\} = \{\tilde{F}_\sigma^1, \ldots., \tilde{F}_\sigma^5\} \quad (17)$$

Where $T(\tilde{\sigma})$ is the term set of $\tilde{\sigma}$, and *NB*, *NM*, *ZR*, *PM* and *PB* are labels of fuzzy sets, which are negative big, negative medium, zero, positive medium, and positive big, respectively. For the control output u_{Fuzzy}, its term set and labels of the fuzzy sets are defined similarly by:

$$T\left(\tilde{u}_{Fuzzy}\right) = \{NB, NM, ZR, PM, PB\} = \left\{\tilde{F}_{u_{Fuzzy}}^1, \ldots., \tilde{F}_{u_{Fuzzy}}^5\right\} \quad (18)$$

The Gaussian membership functions were used for both input and output of FLC. Gaussian membership function is defined as:

$$\mu_{\tilde{F}_{u_{Fuzzy}}^{Nfs}} = exp\left(\frac{-(x-a_F)^2}{2(b_F)^2}\right) \quad (19)$$

Where subscript Nfs represents the total number of fuzzy sets, a_F and b_F are respectively, the center and the width of the Gaussian membership function, they are determined by the GA. Moreover, in Figure 3 the term ϕ indicates the boundary layer around the switch surface.

From these two term sets, we can build the following fuzzy rules [28]:

$\mathbf{R^1}$: If σ is NB, then u_{Fuzzy} is PB.
$\mathbf{R^2}$: If σ is NS, then u_{Fuzzy} is PS.
$\mathbf{R^3}$: If σ is ZR, then u_{Fuzzy} is ZR.
$\mathbf{R^4}$: If σ is PS, then u_{Fuzzy} is NS.
$\mathbf{R^5}$: If σ is PB, then u_{Fuzzy} is NB.

The result of the inference for every σ_i (general case) can be written as follows:

$$u_{Fuzzy_i} = -K_i \times sig(\sigma_i/\phi_i) \tag{20}$$

Where $sig(z) = \begin{cases} -1 & z < -1 \\ z & -1 \le z \le 1 \\ 1 & z > A \end{cases}$ (21)

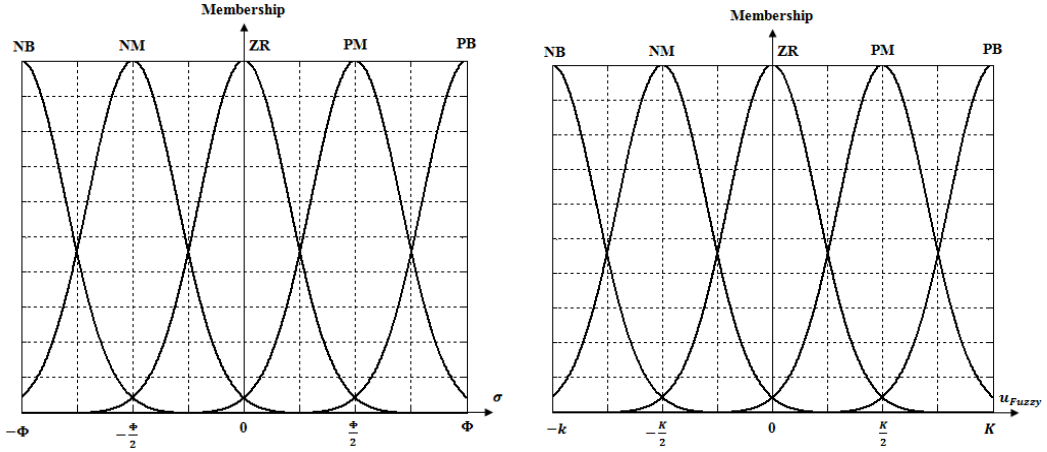


Figure 3. Fuzzy Partition of the Universe of Discourse of σ and u_{Fuzzy}

4. Stability Analysis

Define the global Lyapunov candidate function $V = \sum_{i=1}^{N} V_i$ where

$$V_i = V_{RFNN_i} + V_{FSMC_i} \Rightarrow \dot{V}_i = \dot{V}_{RFNN_i} + \dot{V}_{FSMC_i} \tag{22}$$

Where $V_{RFNN_i} = \frac{1}{2}\left(e_{RFNN_i}(k)\right)^2 = \frac{1}{2}\left(E_{eqv\,i}(k)\right)^2$ is the Lyapunov function of the RFNN_i, where $e_{RFNN_i}(k)$ represents the estimation error in the learning process and V_{FSMC_i} is the Lyapunov function of FSMC_i. For $\dot{V}_{RFNN_i}$, we assume that it can be approximated as $\dot{V}_{RFNN_i} \cong \frac{\Delta V_{RFNN_i}(k)}{\Delta T}$ where $\Delta V_{RFNN_i}(k)$ is the derivation of a discrete-time Lyapunov function, and ΔT is a sampling time.

According to [5], $\Delta V_{RFNN_i}(k)$ can be guaranteed to be negative if the learning rates are chosen using the following equation:

$$\eta_{w_{oq}^4,\,RFNN_i} \simeq \eta_{w_i^1,\,RFNN_i} = \eta_{a_{ij,\,RFNN_i}} = \eta_{b_{ij},\,RFNN_i} = \gamma \times \left[\frac{1}{\left|w_{i,RFNN_i,max}^1\right|\left(\frac{2}{b_{ij,RFNN_i,min}}\right)}\right]^2 \quad (23)$$

Where $\gamma = \frac{1}{R_{RFNN_i} \times N_{RFNN_i,max}}$ and R_{RFNN_i} and $N_{RFNN_i,max}$ denote the number of rules in the RFNN_i and the maximum of the number of fuzzy sets with respect to the input.

Proposition1. For the system (1), the designed fuzzy controller which has the fuzzy sliding surface makes the sub-system i trajectories ultimately bounded under the region $B_i(t) = \{\bar{x}_i, |\sigma_i(\bar{x}_i, t)| \le \phi_i\}$ with $\phi_i > 0$, if we choose K_i as

$$K_i \ge F_i + \sum_{k=0}^{P}\sum_{j=1}^{N} \xi_{ij}^k \left\|\bar{x}_{s_j}\right\|^k + D_i + \psi_i \quad (24)$$

Where ψ_i is a positive constant.

Proof. Define the Lyapunov function

$$\begin{aligned} V_{FSMCi} &= \frac{1}{2}\sigma_i^2 \\ \dot{V}_{FSMCi} &= \sigma_i\left(\sum_{j=1}^{n_i-1} C_{i,1}\, e_{i,(j+1)} + f_i(\bar{x}_i) + g_i(\bar{x}_i)u_i(t) + z_i(\bar{x}_s) + d_i(\bar{x}_i) - x_{id}^{(n_i)}\right) \\ &= \sigma_i\left(f_i(\bar{x}_i) + z_i(\bar{x}_s) + d_i(\bar{x}_i) - K_i \times sig(\sigma_i/\Phi_i)\right) \end{aligned} \quad (25)$$

If $|\sigma_i| > \phi_i$,

$$\begin{aligned} \dot{V}_{FSMCi} &\le \sigma_i\left(F_i + \sum_{k=0}^{P}\sum_{j=1}^{N} \xi_{ij}^k \left\|\bar{x}_{s_j}\right\|^k + D_i - K_i \times sig(\sigma_i/\Phi_i)\right) \\ &\le \sigma_i\left(F_i + \sum_{k=0}^{P}\sum_{j=1}^{N} \xi_{ij}^k \left\|\bar{x}_{s_j}\right\|^k + D_i\right) - K_i \times |\sigma_i| \\ &\le -\psi_i \times |\sigma_i| \end{aligned} \quad (26)$$

Therefore, using the value K_i of (24), we always have the bounded system trajectories under the range of $B_i(t)$.

5. Tuning of the Decentralized RFNN-FSMC Parameters by the Genetic Algorithm

In this section we present how the proposed automatic tuning of the decentralized controller is formulated by using the GA approach, where all the parameters that need to be optimized are initially randomized, and then tuned and optimized simultaneously by GA. The basic GA concept is first briefly presented.

5.1. Genetic Algorithms

GA can be viewed as a general-purpose search method, an optimization method, or a learning mechanism, based loosely on Darwinian principles of biological evolution, reproduction and ‘‘the survival of the fittest’’ [11]. The theoretical foundations of GAs were originally developed by Holland [15]. The idea of GAs is based on the proposed evolutionary process of biological organisms in nature.

At each step of evolution, natural individuals evolve according to the principles of natural selection and the "survival of the fittest." Individuals which are more successful in adapting to their environment will have a better chance of surviving and reproducing, while individuals which are less fit will be eliminated. This means that genes from the highly fit individuals (parents) will spread to an increasing number of individuals (children) in each successive generation. The combination of good characteristics from highly adapted ancestors may produce even fitter off spring. In this way, species evolve to become better and better adapted to their environment.

A GA simulates these processes by taking an initial population of individuals and applying genetic operators (selection, crossover and mutation) in each reproduction. In optimization terms, each individual in the population is encoded into a string, or chromosome, which represents a possible solution to a given problem. The fitness of each member of the population (individual) is evaluated using a fitness function. Highly fit individuals or solutions have higher probability to reproduce by exchanging pieces of their genetic information, in a mutation procedure, with other highly fit individuals. This produces new "offspring" which share some characteristics off both parents. Mutation is often applied by altering some genes in the strings. The offspring can either replace less fit individuals (steady-state approach) or replace the whole population (generational approach). The selection-crossover-mutation cycle is repeated until a satisfactory solution is found.

The computational flowchart of GA is shown in Figure 4.

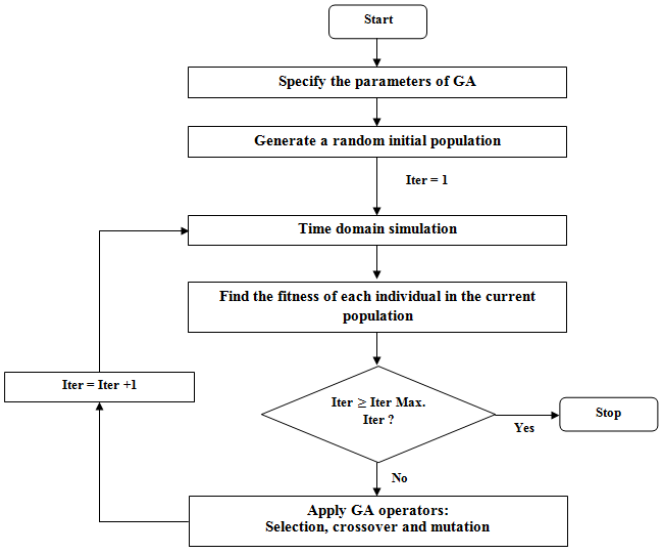


Figure 4. GA Computational Flowchart

5.2. Coding Strategy of the Decentralized RFNN-FSMC Parameters

In this paper, the GA is used to the problem of determining and optimizing some key parameters of each sub-RFNN-FSMC fixed previously. Generally, the task of defining a fitness function is usually application-specific, such that it is formulated to achieve the goal of the controller. Since the central objective of the control system in this study is to minimize the errors between the actual sub-system response $x_{i,1}$ and the desired reference state x_{id}, the following objective (or cost) function is used.

$$\mathrm{F} = \int \left(\left| e_{1,1}(t) + , \ldots . , + e_{i,1}(t) \right| \right) dt \tag{27}$$

In this study, the controller parameters that need to be tuned are the following: the learning-rate parameters η_{Neti} of the BPA used to establish the multilayer feed-forward network, the slopes $C_{i,1}, C_{i,2}, \ldots, C_{i,(n_i-1)}$ of the switching surface σ_i used as an input of the FLC_i and the membership functions parameters of the FLC_i, to improve firstly the network leaning ability and increase the speed of network convergence, to enhance the rate of convergence on the sliding surface and to boost the approximation mechanism ability of each FLC_i.

We can represent the set of all parameters to be optimized in this problem with a vector P given as follows:

$$\mathrm{P} = [C_{1,1}, \ldots, C_{i,1}, C_{1,2}, \ldots, C_{i,2}, \ldots, C_{1,(n_i-1)}, \ldots, C_{i,(n_i-1)}, \eta_{Net\,1}, \eta_{Net\,2}, \ldots, \eta_{Net\,i},$$
$$a_{input\,F_1}, b_{input\,F_1}, a_{input\,F_2}, b_{input\,F_2}, \ldots, a_{input\,F_i}, b_{input\,F_i}, a_{output\,F_1}, b_{input\,F_1},$$
$$\ldots, a_{output\,F_i}, b_{output\,F_i}] = [\mathrm{p}^{i_p}] \quad (28)$$

With $i_p = 1, \ldots, N_p$, where N_p is the total number of the RFNN-FSMC parameters to be optimized. Each sub-RFNN-FSMC_i has $\big((n_i - 1) + 5\big)$ parameters to be tuned, there are basically $N_p = N \times (n_i + 4)$ parameters.

As the GA deals with coded parameters, all the controller parameters that need to be optimized must be encoded into a finite length of string. The linear mapping method [20] is used for this purpose, and can be expressed as follows:

$$g_{i_p} = \frac{G_{i_p min} + \big(G_{i_p max} - G_{i_p min}\big) \cdot A_{i_p}}{(2^{N_b} - 1)} \quad (29)$$

Where g_{i_p} is the actual value of the $i_p th$ parameter, and A_{i_p} is the integer represented by a $N_b - bit$ string gene.

$G_{i_p max}$ and $G_{i_p min}$ are user-defined upper and lower limits of the gene. The encoded genes are concatenated to form a complete chromosome. Each of the parameters is encoded into 8-bit strings, resulting in a complete chromosome of $N_b \times N_p$ bits. The coded parameters of the controller are arranged as shown in the following representation to form the chromosome of the population:

Gene $|1|2|\ 3|\ \ldots\ldots\ldots\ldots\ N_p|$

Chromosome $|\mathrm{Sub} - \mathrm{chromosomes\ of}\ C_{11}, \ldots, C_{i,1}|$

$|\mathrm{Sub} - \mathrm{chromosomes\ of}\ C_{1,2}, \ldots, C_{i,2}\ |\mathrm{Sub} - \mathrm{chromosomes\ of}\ C_{1,(n_i-1)}, \ldots, C_{i,(n_i-1)}|$

$|\mathrm{Sub} - \mathrm{chromosomes\ of}\ \eta_{Net\,1}, \eta_{Net\,2}, ., \eta_{Net\,i}|$

$|\mathrm{Sub} - \mathrm{chromosomes\ of}\ a_{input\,F_1}, b_{input\,F_1}, a_{input\,F_2}, b_{input\,F_2}, \ldots, a_{input\,F_i}, b_{input\,F_i}|$

$|\mathrm{Sub} - \mathrm{chromosomes\ of}\ a_{output\,F_1}, b_{input\,F_1}, \ldots, a_{output\,F_i}, b_{output\,F_i}|$

It can be observed that the number of genes is equal to N_p which is the number of all controller parameters must be tuned.

5.3. GA Parameters Specification

Generally, implementation of GA requires the determination of six fundamental issues: chromosome coding strategy, selection function, the genetic operators, initialization, termination and evaluation function. However, type of GA and important parameters specifications related with GA used in this study are defined in Table 1. Additionally, in this paper we have used a non-uniform mutation strategy described as follow:
Let P_m the mutation rate given by:

$$p_m = 1 - \frac{0.1}{N_b} \times i, i = 1, \ldots\ldots, N_b \quad (30)$$

Let also $P_{m\,rand}$ a random mutation rate given by:

$$P_{m\,rand} = rand(i,j), i = 1, \ldots\ldots, N_b \; and \; j = 1, \ldots\ldots, N_p \quad (31)$$

Hence, the mutation mechanism is performed according to the following mutation condition:

$$p_m < P_{m\,rand} \quad (32)$$

Based on our past experiences (tests), this mutation strategy provides faster convergence when compared to constant probability rates.

Table 1. GA Parameter Specifications

GA properties	Parameters
Type of GA	Classical GA
Chromosomes type	Real values
Bit number in each gene	N_b
Chromosomes length	N_p
Selection type	Elitism
Crossover type	Arithmetical crossover [7]
Mutation type	Non-uniform mutation
Crossover rate P_C	0.55
Mutation rate P_m	Given by (30)

Remark 5: It should be noted that, applying GA to tune a real process plant is rather impractical, as the algorithm needs to perform a large amount of repetitive iterations and evaluation on the plant, which may be impracticable and time consuming. Moreover, the risk of instability is high during the initial stages of the GA iterations. Usually, a plant model can be used to simulate the controllability and evaluation of the controller. Also, the optimized parameters bounds are must be determined. So, in this study, theses bounds are fixed according to the stability analysis conditions (learning-rate parameter η_{Net}) and as well as on the controlled system available information.

6. Simulation Results

In this section, we present two illustrative examples to show the effectiveness of the proposed decentralized control scheme. Where, it is applied respectively to a two-link robotic manipulator and to interconnected inverted pendulums on carts; these systems have a strong interconnection between their subsystems and they can be considered as good examples to test our control method. In addition, simulation experiments using the conventional SMC had been also studied to demonstrate the effectiveness of the proposed method.

Example 1: Two-link robotic manipulator

Figure 5 shows a two-link robotic manipulator used by [30], in which m_1 and m_2 are the masses of links 1 and 2, respectively; l_1 and l_2 are the lengths of links 1 and 2, respectively. The dynamic equation of the robotic manipulator is given by

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = U \tag{33}$$

Where $q = [q_1\ q_2]^T$ is a 2×1 vector of joint position, $\dot{q} = [\dot{q}_1\ \dot{q}_2]^T$ is a 2×1 vector of joint velocity, $\ddot{q} = [\ddot{q}_1\ \ddot{q}_2]^T$ is a 2×1 vector of joint acceleration, $U = [u_1\ u_2]^T$ is a 2×1 vector of the control input torque; $M(q)$ is a 2×2 inertial matrix, $C(q,\dot{q})$ is a 2×2 of Coriolis and centrifugal forces, and $G(q)$ is a 2×1 gravity vector.

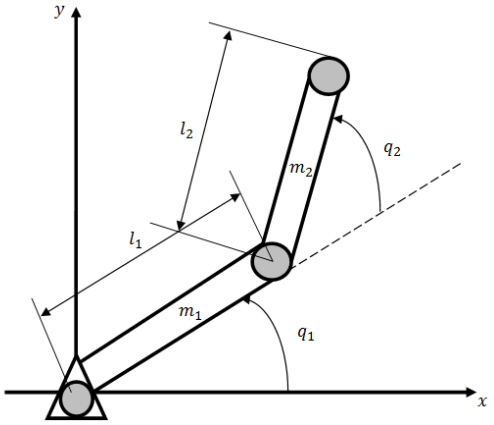


Figure 5. Two-link Robotic Manipulator

We have also from [30]

$$M(q) = \begin{bmatrix} D_{11}(q_2) & D_{12}(q_2) \\ D_{12}(q_2) & D_{22}(q_2) \end{bmatrix},$$

$$C(q,\dot{q})\dot{q} = -\begin{bmatrix} F_{12}(q_2)\dot{q}_2^2 + 2F_{12}(q_2)\dot{q}_1\dot{q}_2 \\ -F_{12}(q_2)\dot{q}_1^2 \end{bmatrix} = -\begin{bmatrix} F_1(\dot{q}_1, q_2, \dot{q}_2) \\ F_2(\dot{q}_1, q_2) \end{bmatrix},$$

$$G(q) = -\begin{bmatrix} G_1(q_1, q_2) \\ G_2(q_1, q_2) \end{bmatrix}.$$

From (33) we can write

$$\begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} = (\det(M(q)))^{-1} \begin{bmatrix} D_{22}(q_2) & -D_{12}(q_2) \\ -D_{12}(q_2) & D_{11}(q_2) \end{bmatrix} \times \left[\begin{bmatrix} F_1(\dot{q}_1, q_2, \dot{q}_2) \\ F_2(\dot{q}_1, q_2) \end{bmatrix} + \begin{bmatrix} G_1(q_1, q_2) \\ G_2(q_1, q_2) \end{bmatrix} + \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \right] \tag{34}$$

Where [30]

$$D_{11}(q_2) = (m_1 + m_2)l_1^2 + m_2 l_2^2 + 2m_2 l_1 l cos(q_2) + J_1$$
$$D_{12}(q_2) = m_2 l_2^2 + m_2 l_1 l_2 cos(q_2)$$
$$D_{22}(q_2) = m_2 l_2^2 + J_2$$

$F_{12}(q_2) = m_2 l_1 l_2 sin(q_2)$
$G_1(q_1, q_2) = -g\big((m_1 + m_2) l_1 cos(q_2) + m_2 l_2 cos(q_1 + q_2)\big)$
$G_2(q_1, q_2) = -g\big(m_2 l_2 cos(q_1 + q_2)\big)$

The parameter values used are the same as those in [30]. $l_1 = 1m$, $l_2 = 0.8\ m$, $J_1 = 5\ kg \cdot m$, $J_2 = 5\ kg \cdot m$, $m_1 = 0.5\ kg$, $m_2 = 6.25\ kg$.

Defining the state vectors $X_1 = \big[x_{1,1}\ x_{1,2}\big]^T$, and $X_2 = \big[x_{2,1}\ x_{2,2}\big]^T$, where $x_{1,1} = q_1$, $x_{1,2} = \dot{q}_1$ $x_{2,1} = q_2$, $x_{2,2} = \dot{q}_2$, we set also $A_d = (\det(M(q)))^{-1}$.

The dynamic equations of the system can be described as

1st joint (s_1)

$$\dot{x}_{1,1} = x_{1,2}$$
$$\dot{x}_{1,2} = A_d \big[D_{22}(x_{2,1})\big(F_1(x_{1,2}, x_{2,1}, x_{2,2}) + G_1(x_{1,1}, x_{2,1}) + u_1\big) - D_{12}(x_{2,1})\big(F_2(x_{1,2}, x_{2,1}) + G_2(x_{1,1}, x_{2,1}) + u_2\big)\big] \quad (35)$$

2nd joint (s_2)

$$\dot{x}_{2,1} = x_{2,2}$$
$$\dot{x}_{2,2} = A_d \big[D_{11}(x_{2,2})\big(F_2(x_{1,2}, x_{2,1}) + G_2(x_{1,1}, x_{2,1}) + u_2\big) - D_{12}(x_{2,1})\big(F_1(x_{1,2}, x_{2,1}, x_{2,2}) + G_1(x_{1,1}, x_{2,1}) + u_1\big)\big] \quad (36)$$

The robotic manipulator was controlled by the decentralized controller. So, the control objective here is to make both joints of the robotic manipulator move quickly and accurately to track a desired trajectory or to reach specified positions in two degrees of freedom. Therefore, the inputs of each $\text{RFNN}_{i=1,2}$ consisted of the desired state and actual state, as $Input_{RFNN_{1/2}} = \big[q_{1d/2d}\ q_{1/2}\big]$ each input variable has five membership functions.
The initial Gaussian membership function parameters in both $\text{RFNN}_{i=1,2}$ are chosen normally as follows:

$$a_{0s_{1/ij}} = a_{0s_{2/ij}} = 0.01 \times \begin{bmatrix} -1.5 & -0.8 & 0 & 0.8 & 1.5 \\ -1.5 & -0.8 & 0 & 0.8 & 1.5 \end{bmatrix},\ b_{0s_{1/ij}} = b_{0s_{2/ij}} = 1.21,\ i = 1,2, j = 1, \ldots, 5 \quad (37)$$

The values of the recurrent weights are initialized to be $w^1_{0s_1/i} = w^1_{0s_2/i} = 0.010, i = 1,2$ The initial values of the weights $w^4_{0s_{1/oq}} = w^4_{0s_{2/oq}} = 0.001$, $o = 1, q = 1, .., 5$.
Among the goals of the decentralized controller design is to force the tracking errors $e_1(t) = q_1(t) - q_{d1}(t)$ and $e_2(t) = q_2(t) - q_{d2}(t)$ to slide along the sliding surfaces. In this example we choose these sliding surfaces as follows:

$$\sigma_{s_1}(E_1) = C_{1,1} e_{1,1} + C_{1,2} e_{1,2} = C_{1,1} e_{1,1} + C_{1,2} \dot{e}_{1,1} \quad (38)$$
$$\sigma_{s_2}(E_2) = C_{2,1} e_{2,1} + C_{2,2} e_{2,2} = C_{2,1} e_{2,1} + C_{2,2} \dot{e}_{2,1} \quad (39)$$

In this paper, a fixed boundary layers $\phi_{i=1,2}$ around the switch surfaces σ_{s_1} and σ_{s_2} are used, which are set to 120 for the two sub-FLCs, the parameters $K_{i=1,2}$ are set to 20. In addition, the optimization problem here is to find the values of all parameters included in the following vector such that the cost function (27) is minimized.

$$P = \left[C_{i,j}, \eta_{Neti}, a_{inputF_i}, b_{inputF_i}, a_{outputF_i}, b_{outputF_i}\right] i = 1,2 \text{ and } j = 1,2 \quad (40)$$

In this example, there are $N_p = 14$ parameters in the controller structure that need to be optimized. The coefficients of the GA are chosen as in Table 1 except N_b which is set to 08. The boundaries of the search space (limit values of the parameters to be optimized) are defined in Table 2. In addition, this algorithm is run ten times for 50 iterations, at each time the algorithm converges to an acceptable solution in less than 45 iterations. The learned parameters values are given also in Table 2.

Table 2. Definition of Parameters Bounds and the Learned Parameters Values for Example 1

Parameter	Min	Max	Value
$C_{1,1}, C_{1,2}$	1,2	64,16	25.7777, 9.4580
$C_{2,1}, C_{2,2}$	1,2	64,16	38.5555, 9.7644
η_{Net1}, η_{Net2}	1.0485	3.0485	3.0030, 2.5664
$a_{inputF1}, a_{inputF2}$	10	120	59.3423, 54.7056
$b_{inputF1}, b_{inputF2}$	10	21	17.8717, 16.3685
$a_{outputF1}, a_{outputF2}$	2.5	20	10.2706, 19.8014
$b_{outputF1}, b_{outputF2}$	2	4	2.8084, 3.0614

The Figure 6 shows the evolution of the cost function values in successive iterations. The final value of the cost function which converge the optimization algorithm is $\mathrm{F}_{\mathrm{best}} = 0.1018$; this value corresponds to the learned parameters presented in Table 2.

For this example, we pick the initial displacements and velocities to be $X_1(0) = [-0.57\ 0]^T$, and $X_2(0) = [-0.57\ 0]^T$ .Simulations of tracking control with a desired reference waves given in (41) for the joint 1 and 2 are shown in Figure 7(a) and Figure 7(b).

$$x_{1d}(t) = x_{2d}(t) = 0.3\sin(0.04\pi t)\ \ (rad) \quad (41)$$

The control signals u_1 and u_2 are also illustrated in Figure 8(a) and Figure 8(b). Else, the step responses of the robotic manipulator are also illustrated in Figure 9(a), Figure 9(b), Figure 10(a) and Figure 10(b).

The system responses $q_1(t)$ and $q_2(t)$ and the output control signals u_1 and u_2 obtained using SMC with the same tracking control test used before are illustrated in Figure 11(a), Figure 11(b), Figure 12(a) and Figure 12(b).

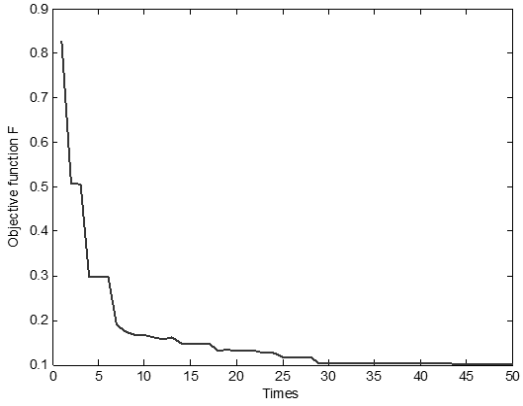
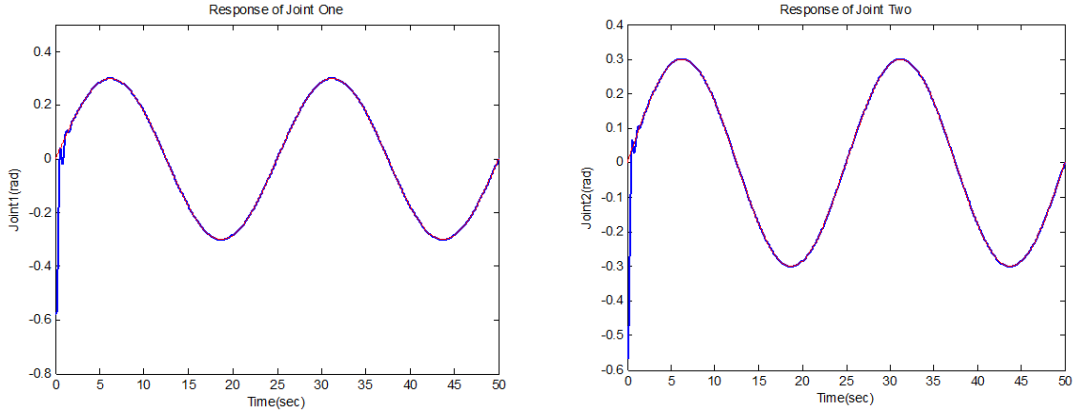
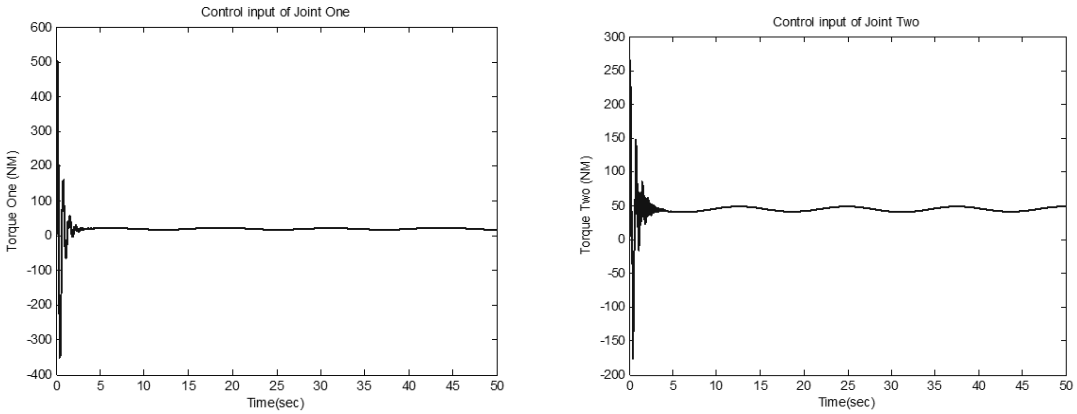


Figure 6. Evolution of Cost Function for Example 1



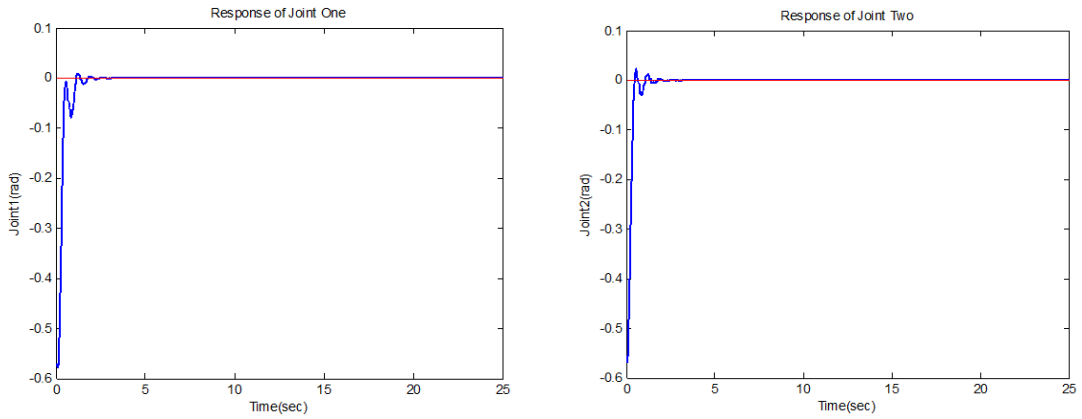
(a) **(b)**

Figure 7. Angular Tracking Responses using the Proposed Decentralized Controller for: (a) the 1st joint and (b) the 2nd joint. Desired Angular Trajectory (red line); System's Output Response (blue line)



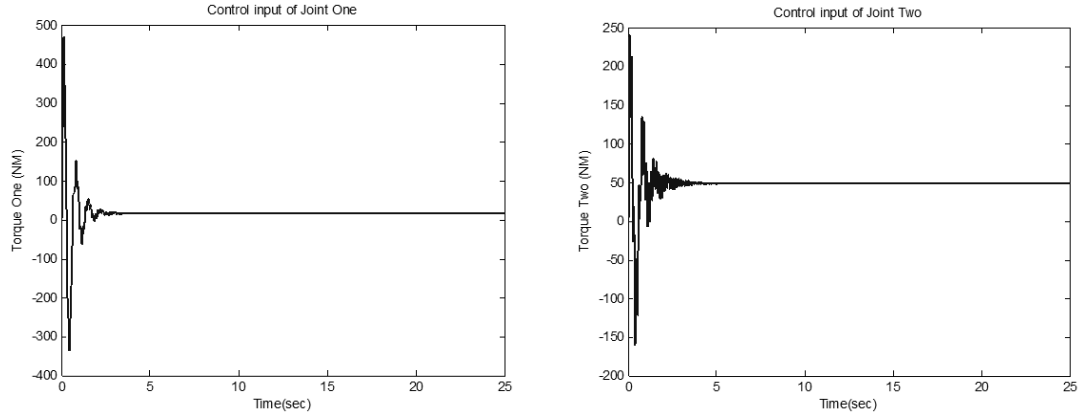
(a) **(b)**

Figure 8. Control Efforts (Angular tracking control): (a) Control Input of the 1st Joint and (b) Control Input of the 2nd Joint



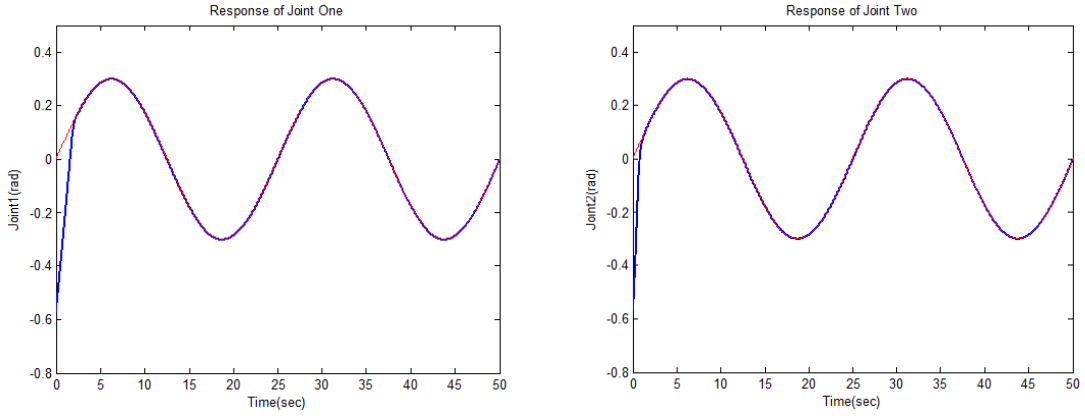
(a) **(b)**

Figure 9. Step Responses of the Robotic Manipulator using the Proposed Decentralized Controller for: (a) the 1st Joint and (b) the 2nd Joint



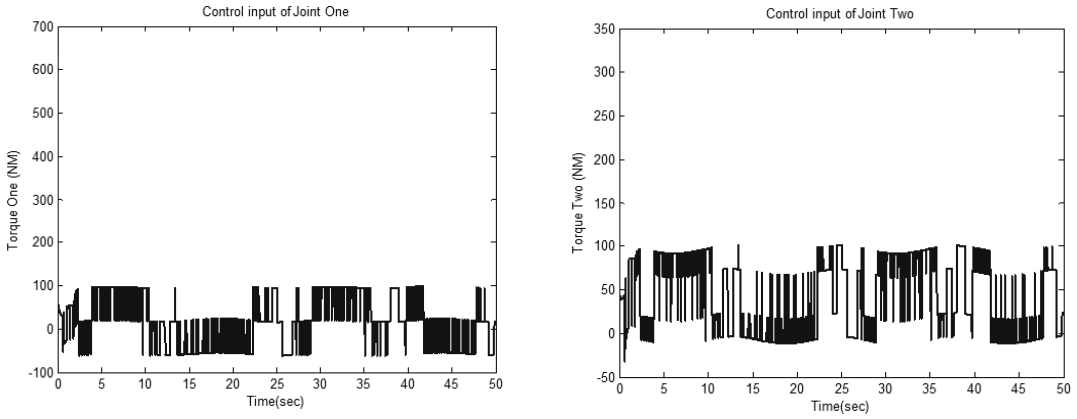
(a) **(b)**

Figure 10. Control Efforts (setpoint): (a) Control Input of the 1st Joint and (b) Control Input of the 2nd Joint



(a) **(b)**

Figure 11. Angular Tracking Responses using Conventional SMC for: (a) the 1st Joint and (b) the 2nd Joint. Desired Angular Trajectory (red line); System's Output Response (blue line)



(a) **(b)**

Figure 12. Control Efforts (Angular tracking control using conventional SMC): (a) Control Input of the 1st Joint and (b) Control Input of the 2nd Joint

It is clear from these results that the robotic manipulator responses with the designed decentralized controller are much faster, with very less settling time. Moreover, we can find that the control result of the conventional SMC produces a serious chattering phenomenon, as in Figure 12(a) and Figure 12(b). On the contrary, the chattering phenomenon of the controlled system was suppressed in the proposed decentralized controller without any degradation of the tracking performance. In addition, compared with the results obtained in [30], the position control of the decentralized RFNN-FSMC-controlled the robotic manipulator has a better performance in the sense of less overshoot and less oscillation around the desired positions, as shown in Figures 9 and Figures 10.

We can see easily that, the system trajectory goes into the region $B_i(t) = \{\bar{x}_i, |\sigma_i(\bar{x}_i, t)| \leq \phi_i = 120\}$ after around $2s$ and remains inside the region after that time.

Example 2: Two inverted pendulums on carts

We consider now two inverted pendulums connected by a moving spring mounted on two cars system (see Figure 13) used by [37, 38]. We assume that the pivot position of the moving spring is a function of time which can change along the full length of the pendulums. For this example we specify this motion of the carts as sinusoidal trajectories. The input to each pendulum is the control signals $u_i, i = 1,2$ applied at the pivot point. The objective of the decentralized controller is to control each pendulum with mass m independently so that each pendulum tracks its own desired reference trajectory while the connected spring and carts are moving.

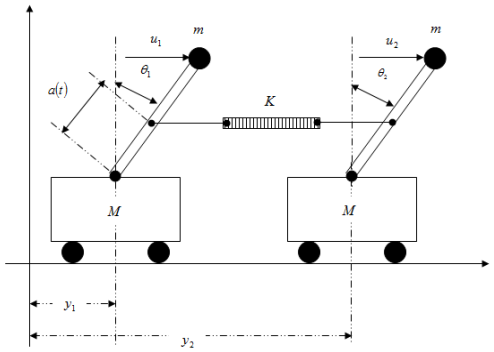


Figure 13. Two Inverted Pendulums on Carts System

Defining the state vectors $X_1 = [x_{1,1}\, x_{1,2}]^T$, and $X_2 = [x_{2,1}\, x_{2,2}]^T$, where $x_{1,1} = \theta_1$, $x_{1,2} = \dot{\theta}_1$, $x_{2,1} = \theta_2$, $x_{2,2} = \dot{\theta}_2$. The dynamic equations of the system can be described as:

1st pendulum (s_1)

$$\dot{x}_{1,1} = x_{1,2}$$
$$\dot{x}_{1,2} = \left(\frac{g}{cl} - \frac{ka(t)(a(t)-cl)}{cml^2}\right)x_{1,1} + \frac{1}{cml^2}u_1 + \frac{ka(t)(a(t)-cl)}{cml^2}x_{2,1} - B_1 x_{1,2}^2 - \frac{k(a(t)-cl)}{cml^2}(y_1 - y_2) \quad (43)$$

2nd pendulum (s_2)

$$\dot{x}_{2,1} = x_{2,2}$$
$$\dot{x}_{2,2} = \left(\frac{g}{cl} - \frac{ka(t)(a(t)-cl)}{cml^2}\right)x_{2,1} + \frac{1}{cml^2}u_2 + \frac{ka(t)(a(t)-cl)}{cml^2}x_{1,1} - B_2 x_{2,2}^2 - \frac{k(a(t)-cl)}{cml^2}(y_2 - y_1) \quad (44)$$

Where u_1 and u_2 are the control signals applied to the pendulums, $B_1 = \frac{m}{M}\sin(x_{1,1})$, $B_2 = \frac{m}{M}\sin(x_{2,1})$ and $\|B_i\| \le \frac{m}{M}$, $i = 1,2$. $c = \frac{m}{M+m}$, m, M, l, k and g are the mass of the pendulum and the car, the length of the pole, the spring constant and the gravity constant (m/s^2), respectively. To carry out this simulation, we choose m = M = 10 kg, l = 1 m, g = 9.81 m/s^2, k = 1 N/m, c = 0.5. We also let $y_1 = \sin(\omega_1 t)$, and $y_2 = L + \sin(\omega_2 t)$, where $\omega_1 \neq \omega_2$ and L is the natural length of the connecting spring. We select L =2 and set $a(t) = \sin(5t)$, $\omega_1 = 2$, $\omega_2 = 3$.

The problem here is to let the states X_1 and X_2 to track the desired sates $X_{id} = [\theta_{1d}\ \theta_{2d}]^T$. In simulation tests, the initial conditions are chosen as $X_1(0) = [0.5\ 0]^T$, and $X_2(0) = [-0.5\ 0]^T$.

The $\text{RFNN}_{i=1,2}$ configuration and their initial parameters are set as in example 1. In addition, FLCs parameters (number of input/output membership function, boundary layer,..), sliding surfaces, vector of parameters to be optimized, cost function F and GA coefficients used in this example are also all set as in example 1. Also, the boundaries of the search space and the learned parameters values for this example are set all in Table 3.

Table 3. Definition of Parameters Bounds and the Learned Parameters Values for Example 2

Parameter	Min	Max	Value
$C_{1,1}, C_{1,2}$	49,14	100,20	90.2511, 15.0018
$C_{2,1}, C_{2,2}$	49,14	100,20	92.1490, 15.0018
η_{Net1}, η_{Net2}	0.95	1.1	1.0589, 1.0782
$a_{inputF1}, a_{inputF2}$	80	120	99.3086, 113.3814
$b_{inputF1}, b_{inputF2}$	15	21	16.9749, 17.3529
$a_{outputF1}, a_{outputF2}$	15	20	17.7715, 18.1440
$b_{outputF1}, b_{outputF2}$	2	4	2.9698, 3.2428

The Figure 14 shows the evolution of the cost function values in successive iterations. The final value of the cost function which converge the optimization algorithm for this example is $F_{best} = 0.1884$. Also, the tracking system responses with the same reference trajectory wave used in example 1 and the control signals generated by each sub-RFNN-FSMC are shown in Figure 15(a), Figure 15(b), Figure 16(a) and Figure 16(b) respectively. The step responses and control inputs of each pendulum are also represented in Figure 17(a), Figure 17(b), Figure 18(a) and Figure 18(b).

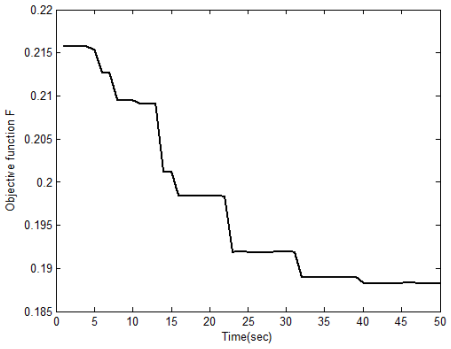
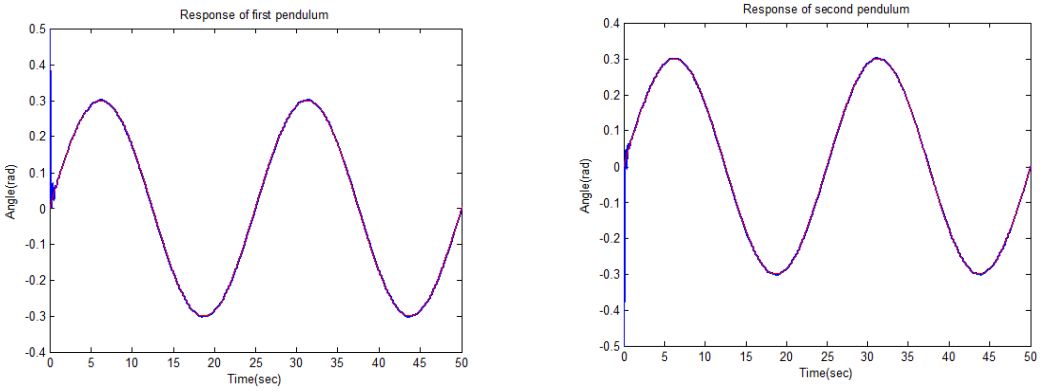
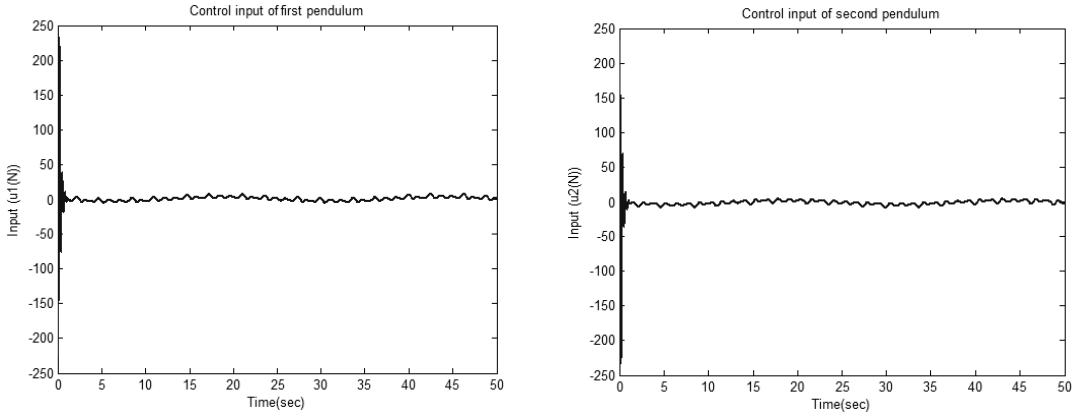


Figure 14. Evolution of Cost Function for Example 2



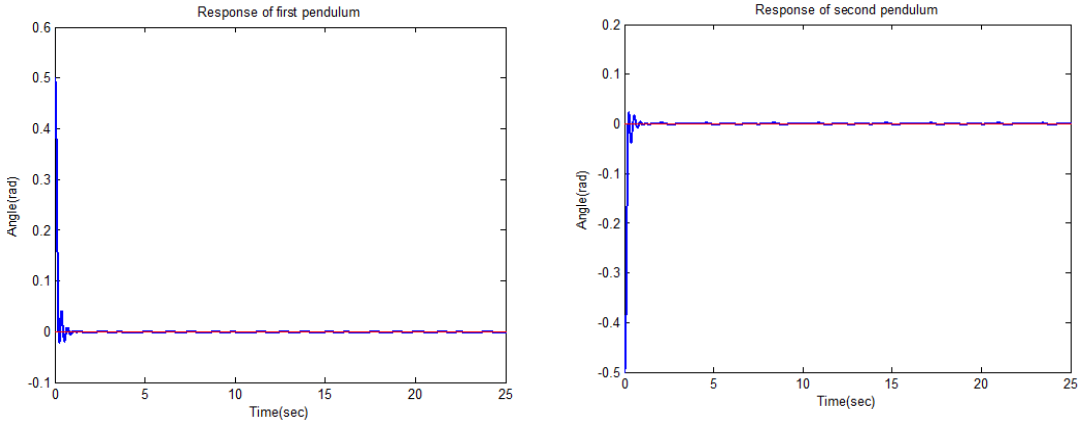
(a) **(b)**

Figure 15. Angular Tracking Responses using the Proposed Decentralized Controller for: (a) the 1st Pendulum and (b) the 2nd Pendulum. Desired Angular Trajectory (red line); System's Output Response (blue line)



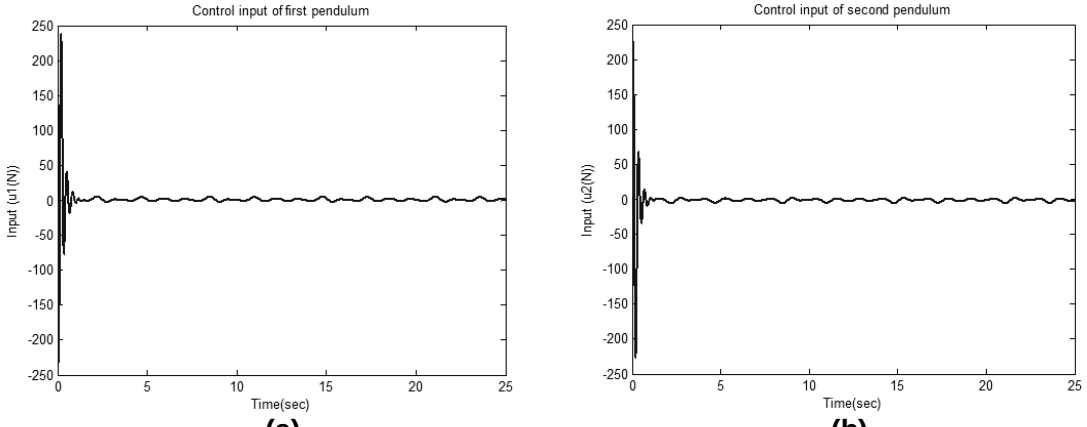
(a) (b)

Figure 16. Control Efforts (Angular tracking control): (a) Control Input of the 1st Pendulum and (b) Control Input of the 2nd Pendulum



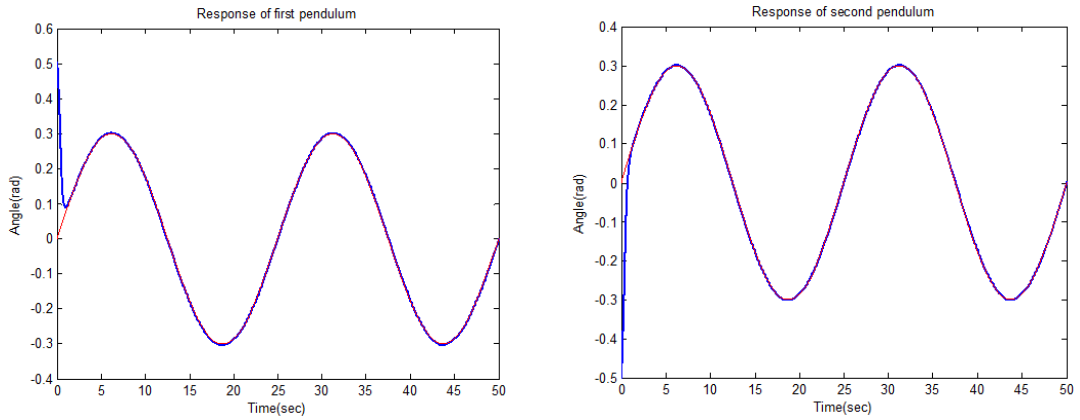
(a) (b)

Figure 17. Step Responses of the Two Inverted Pendulums on Carts System using the Proposed Decentralized Controller for: (a) the 1st Pendulum and (b) the 2nd Pendulum



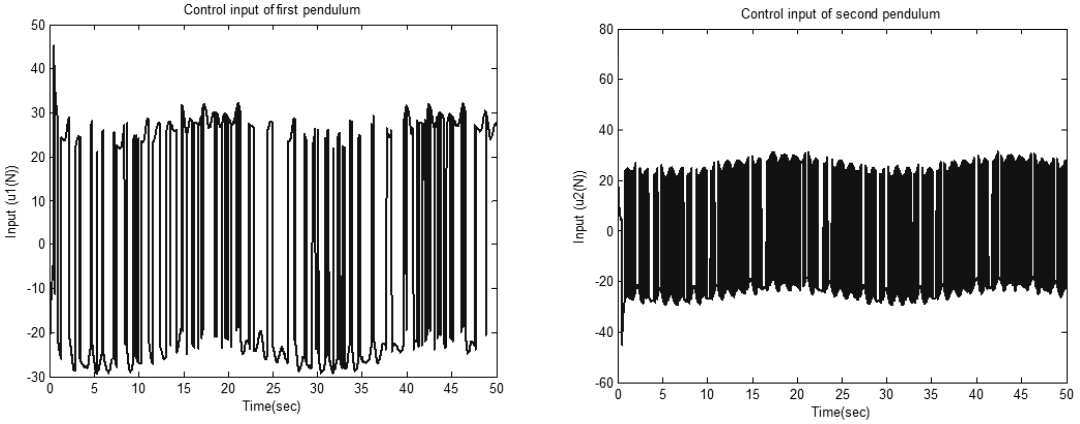
(a) (b)

Figure 18. Control Efforts (setpoint): (a) Control Input of the 1st Pendulum and (b) Control Input of the 2nd Pendulum



(a) **(b)**

Figure 19. Angular Tracking Responses using Conventional SMC for: (a) the 1st Pendulum and (b) the 2nd Pendulum. Desired Angular Trajectory (red line); System's Output Response (blue line)



(a) **(b)**

Figure 20. Control Efforts (Angular tracking control using conventional SMC): (a) control Input of the 1st Pendulum and (b) Control Input of the 2nd Pendulum

All simulation results demonstrate that the controlled system responses with the designed decentralized controller are much faster, with very less settling time compared with the results (step responses, see Figure 17) in [12, 37, 38]. Moreover, the chattering phenomenon was completely eliminated without any degradation in control performance compared to the conventional SMC applied to the same system as shown in Figure 19(a) ,Figure 19(b), Figure 20(a) and Figure 20(b).

We see also for this example that, the system trajectory goes into the region $B_i(t) = \{\bar{x}_i, |\sigma_i(\bar{x}_i, t)| \leq \phi_i = 120\}$ after around $2s$ and remains inside the region after that time.

7. Concluding Remarks

In this paper, we present a new controller using a hybrid approach for large-scale nonlinear systems with higher order interconnections and uncertainties. By using the

continuous output of the FLC to replace the discontinuous output of the SMC, the RFNN-FSMC can eliminate completely chattering phenomenon without any degradation in control performance. The structure of the RFNN that estimates the equivalent control was a standard four layer-feed-forward RFNN with the backpropagation adaptation algorithm. The corrective control was accepted as a measure of error to update the weights of the RFNN. The global stability of the system is proven by the Lyapunov function.

The developed decentralized controller is tested using two highly interconnected nonlinear systems as examples. Simulation results and performance comparisons with SMC and other control techniques used in the literature to control the same systems employed in this study, show clearly the effectiveness and efficiency of this approach. In addition, GA has been adopted to tune some key parameters of the decentralized controller.

The simulation results demonstrate the following attractive features of the control method:

1. The main contribution of the methodology presented in this paper is the proposition of a new general decentralized control structure for a class of large-scale nonlinear systems. Moreover, this method can be implemented in real time control and make the controller design easy and extendable to higher number of subsystems. On the other hand, all methods presented in references [1, 6, 22, 24] from which our method was inspired are direct applications.

1. There is no need to know the dynamical equation of each subsystem to compute the equivalent control,

2. Chattering and the excessive activity of the control signal are eliminated without any degradation in control performance,

3. The use of the GA, allows us to tune some design controller parameters in an optimal manner,

4. In the decentralized controller design, it is assumed that the bounds of interconnections and uncertainties exist but they are unknown. It further implies that the bounds can be arbitrary large. This is useful in real system, because it is difficult to estimate the bounds of interconnections and uncertainties.

References

[1] A. Boubakir, F. Boudjema and S. Labiod, "A Neuro-fuzzy sliding Mode Controller Using Nonlinear Sliding Surface Applied to the Coupled Tanks System", International Journal of Automation and Computing, vol. 6 no. 1, **(2009)**, pp. 72–80.

[2] B. M. Mirkin, E. L. Mirkin and P. O. Gutman, "Sliding mode coordinated decentralized adaptive following of nonlinear delayed plants", Proceedings of the International Workshop on Variable Structure Systems, **(2006)** June 5–7; Alghero, Italy, pp. 244–249.

[3] B. Yoo and W. Ham, "Adaptive sliding mode control of nonlinear systems", IEEE Transaction on Fuzzy systems, vol. 6, no. 2, **(1998)**, pp. 315–321.

[4] C. S. Chen and W. L. Chen, "Robust adaptive sliding mode control using fuzzy modeling for an inverted-pendulum system", IEEE Transactions Industrial Electronics, vol. 45, no. 2, **(1998)**, pp. 297–306.

[5] C. H. Lee and C. C. Teng, "Identification and control of dynamic systems using recurrent fuzzy neural networks", IEEE Transaction on Fuzzy systems, vol. 8, no. 4, **(2000)**, pp. 349–366.

[6] C. H. Tsai, H. Y. Chung and F. M. Yu, "Neural-sliding mode control with its applications in seesaw systems", IEEE Transactions on Neural Networks, vol. 15, no. 1, **(2004)**, pp. 124–134.

[7] C. J. Wu and G. Y. Lin, "Design of fuzzy logic controllers using genetic algorithms", Proceedings of IEEE International Conference on Systems, Man, and Cybernetics, vol. 6, **(1999)**, pp. 104–109.

[8] C. Mnasri and M. Gasmi, "Robust decentralized sliding mode control for large scale uncertain systems", Proceedings of the 26th Chinese Control Conference, **(2007)** 26–31 July; Hunan, China, pp. 76–81.

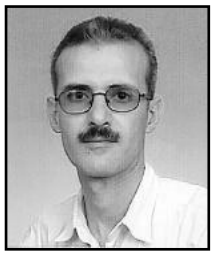
[9] C. W. Tao, M. L. Chan and T. T. Lee, "Adaptive fuzzy sliding mode controller for linear systems with mismatched time-varying uncertainties", IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, vol. 33, no. 2, **(2003)**, pp. 283–294.
[10] D. Boukhetala, F. Boudjema, T. Madani, M. S. Boucherit and N. K. M'Sirdi, "A New Decentralized Variable Structure Control for Robot Manipulators", International Journal of Robotics and Automation, vol. 18, no.1, **(2003)**, pp. 28–40.
[11] E. G. Shopova, N. G. Vaklieva and Bancheva, "BASIC–A genetic algorithm for engineering problems solution", Computers and Chemical Engineering, vol. 30, no. 8, **(2006)**, pp. 1293–1309.
[12] H. K. Lam, F. H. F. Leung and Y. S. Lee, "Design of a switching controller for nonlinear systems with unknown parameters based on a fuzzy logic approach", IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, vol. 34, no. 2, **(2004)** April, pp. 1068–1074.
[13] H. Morioka, K, Wada, A, Sabanovic and K. Jezernik, "Neural network based chattering free sliding mode control", Proceedings of the 34th SICE Annual Conference, **(1995)** July 26–28; Hokkaido, Japan, pp. 1303–1308.
[14] J. A. Burton and A. S. I. Zinober, "Continuous approximation of variable structure control", International Journal of Systems Science", vol. 17, no. 6, **(1986)**, pp. 875–885.
[15] J. H. Holland, "Adaptation in Natural and Artificial Systems", Ann Arbor, University of Michigan Press, MI, **(1975)**.
[16] J. J. Slotine and W. Li, "Applied Nonlinear Control", Englewood Cliffs, Prentice-Hall, **(1991)**.
[17] J. Wang, A. B. Rad and P. T. Chan, "Indirect adaptive fuzzy sliding mode control", Part I: Fuzzy switching, Fuzzy Sets and Systems, vol. 122, no. 1, **(2001)**, pp. 21–30.
[18] K. D. Young, V. I. Utkin and U. Ozguner, "A control engineer's guide to sliding mode control", Proceedings of the IEEE International Workshop on Variable Structure Systems, **(1996)** December 5–6:1–14; Tokyo, Japan, pp. 1–14.
[19] K. Jezernik, M. Rodič, R. Šafarič and B. Curk, "Neural network sliding mode robot control", Robotica, vol. 15, no. 1, **(1997)**, pp. 23–30.
[20] L. Davis, "Handbook of Genetic Algorithms", Van Nostrand, Reinhold, New York, **(1991)**.
[21] M. Ertugrul, O. Kaynak, A. Sabanovic and K. Ohnishi, "A Generalized Approach for Lyapunov Design of Sliding Mode Controllers for Motion Control Applications", Proceedings of the 4th IEEE International Workshop on Advanced Motion Control, **(1996)**, Mie University, Japan, pp. 407–412.
[22] M. Ertugrul, O. Kaynak, "Neuro-Sliding mode control of robotic manipulators", Mechatronics, vol. 10, **(2000)**, pp. 239–263.
[23] N. Sadati and R. Ghadami, "Adaptive multi-model sliding mode control of robotic manipulators using soft computing", Neurocomputing, vol. 71, no. 2, **(2008)**, pp. 2702–2710.
[24] P. Jiunshian, L. Jianming, M. Yasser and T. Yahaghi, "Neuro sliding mode control for magnetic levitation systems", IEEE International Symposium on Circuits and Systems, **(2005)**, pp. 5130–5133.
[25] R. Palm, "Sliding mode fuzzy control", Proceedings of IEEE Conference on Fuzzy Systems, **(1992)**, San Diego, pp. 519–526.
[26] R. J. Wai, C. M. Lin and C. F. Hsu, "Adaptive fuzzy sliding-mode control for electrical servo drive", Fuzzy Sets and Systems, vol. 143, no. 2, **(2004)**, pp. 295–310.
[27] S. Jain, F. Khorrami and B. Fardanesh, "Decentralized control of large scaled power systems with unknown interconnections", International journal of control", vol. 63, no. 3, **(1996)**, pp. 591–608.
[28] S. W. Kim and J. J. Lee, "Design of a Fuzzy Controller with Fuzzy Sliding Surface", Fuzzy Sets and Systems, vol. 71, no. 3, **(1995)**, pp. 359–367.
[29] T. L. Chern and Y. C. Wu, "Design of brushless DC position servo systems using integral variable structure approach", Proceedings of IEE Electric Power Applications, vol. 140, no. 1, **(1993)**, pp. 27–34.
[30] T. P. Leung, Q. J. Zhou and C. Y. Su, "An adaptive variable structure mode following control design for robot manipulator", IEEE Transaction on Automatic Control, vol. 36, no.3, **(1991)**, pp. 347–353.
[31] V. H. Benitez, E. N. Sanchez and A. G. Loukianov, "Decentralized adaptive recurrent neural control structure", Engineering Applications of Artificial Intelligence, vol. 20, no. 8, **(2007)**, pp. 1125–1132.
[32] V. I. Utkin, "Sliding modes in control and optimization", Berlin, Springer, **(1992)**.
[33] W. Sun and Y. Wang, "A recurrent fuzzy neural network based adaptive control and its application on robotic tracking control", Neural Information Processing Letters and Reviews, vol. 5, no. 1, **(2004)** October, pp. 19–26.
[34] X. G. Yan, C. Edwards and S. K. Spurgeon, "Decentralized robust sliding mode control for a class of nonlinear interconnected systems by static output feedback", Automatica, vol. 40, no. 4, **(2004)**, pp. 613–620.
[35] Y. Fang, T. W. S. Chow and X. D. Li, "Use of a recurrent neural network in discrete sliding-mode control", Proceedings of IEE Control Theory and Applications, vol. 146, no. 1, **(1999)**, pp. 84–90.
[36] Y. R. Hwang and M. Tomizuka, "Fuzzy smoothing algorithms for variable structure systems", IEEE Transaction on Fuzzy Systems, vol. 2, no. 4, **(1994)**, pp. 277–284.

[37] Z. M. Yeh, "Adaptive multivariable fuzzy logic controller", Fuzzy Sets and Systems, vol. 86, no. 1, **(1997)**, pp. 43–60.
[38] Z. P. Jiang, "New results in decentralized adaptive control with output-feedback", Proceedings of the 38th Conference on Decision Control, **(1999)**, Phoenix, AZ, pp. 4772–4777.

Authors

Mr. Fouad Allouani was born in EL-Aouinet, Tebessa, Algeria, on June 12, 1979. He received the Engineer degree in electronic, option industrial control from University of Tebessa, Tebessa, Algeria, in 2002, the Magister in electronic, option industrial control from University of M'sila, M'sila, Algeria, in 2006. He is currently a Ph.D. candidate in automatic control at the Ecole Nationale Polytechnique, Algiers, Algeria and an assistant professor at the department of electrical engineering, University of Khenchela, Algeria. His research interests include computational intelligence (artificial neural networks, evolutionary computation, swarm intelligence, fuzzy systems,..) in Process Control.

Prof. Djamel Boukhetala received the engineering degree in automation from the Institut National des hydrocarbures et de la chimie, Algeria, and the Magister and Doctorat d'Etat (Ph.D.) in automatic control from the Nationale Polytechnique, Algiers, Algeria in 1989, 1993, and 2002, respectively. He is currently a professor at the Department of Electrical Engineering of the Ecole Nationale Polytechnique, Algiers, Algeria and the director of the Laboratoire de commande des processus in the same school. His research interests are decentralized control, nonlinear control, fuzzy control, and artificial neural networks control applied to robotics and power systems.



Prof. Fares Boudjema was born in Algiers, Algeria, on March 28, 1962. He received the Engineer degree in Electrical Engineering from the Ecole Nationale Polytechnique, Algiers, Algeria in 1985, the DEA degree, and the Doctorat degree in Automatic Control from the University of Paul Sabatier, Toulouse, France, in 1987, and 1991, respectively. In 1991, he joined the department of Electrical Engineering at the Ecole Nationale Polytechnique, Algiers, as an Assistant Professor. He was promoted to Associate Professor in 1994, and Professor in 2000. He was the head of the control process laboratory from 2000 through 2005. Since March 2010, he has been the head of the automatic control department. His research interests include application of sliding mode control, artificial neural network control, fuzzy control, and decentralized control in the field of the Electrical machines, power systems, robotics and plus house energy.