IAES International Journal of Robotics and Automation (IJRA)
Vol. 10, No. 2, June 2021, pp. 91~103
ISSN: 2722-2586, DOI: 10.11591/ijra.v10i2.pp91-103

Switched time delay control based on neural network for fault detection and compensation in robot

Maincer Dihya[1], Mansour Moufid [2], Boudjedir Chemseddine[3], Bounabi Moussaab[4]
[1,2]Department Instrumentation and Automatic, Control Laboratory of Robots Parallelism Electroénergetic, University of Science and Technology Houari Boumediene, Algiers, Algeria
[3]Laboratory of Process and Control, Polytechnic National School, Algiers, Algeria
[4]Photovoltaic Communication and Conversion Devices Laboratory, Polytechnic National School, Algiers, Algeria

Article Info

Article history:

Received Aug 13, 2020
Revised Feb 2, 2021
Accepted Mar 5, 2021

Keywords:

Artificial neural network
Fault detection
Multi-layer perceptron
Robot manipulator (SCARA)
Sliding mode control
Time delay and control

ABSTRACT

Fault detection in robotic manipulators is necessary for their monitoring and represents an effective support to use them as independent systems. This present study investigates an enhanced method for representation of the faultless system behavior in a robot manipulator based on a multi-layer perceptron (MLP) neural network learning model which produces the same behavior as the real dynamic manipulator. The study was based on generation of residue by contrasting the actual output of the manipulator with those of the neural network; Then, a time delay control (TDC) is applied to compensate the fault, in which a typical sliding mode command is used to delete the time delay estimate produced by the belated signal in order to obtain strong performances. The results of the simulations performed on a model of the SCARA arm manipulator, showed a good trajectory tracking and fast convergence speed in the presence of faults on the sensors. In addition, the command is completely model independent, for both TDC and MLP neural network, which represents a major advantage of the proposed command.

Corresponding Author:

Maincer Dihya
Department Instrumentation and Automatic Control
University of Science and Technology Houari Boumediene
USTHB, B.P. 32, El-Alia, Bab-Ezzouar, 16 111, Algeria
Email: dmaincer@usthb.dz

1. INTRODUCTION

In various industrial processes robot manipulators have invaded the mode of technology; they are used to carry out complex and repetitive tasks quickly and efficiently. They are connected to each other by joints so that the manipulators follow the reference trajectory, where articulations must be precisely controlled. To perform these tasks, the manipulators are usually quite complex which increases their factor for fault. Thus, in order to have a good fault diagnosis on a manipulator, it is necessary to have a precise knowledge of its mathematical model. However, it is very difficult to obtain a precise of model as the modeling of dynamic robot which is not always an obvious task. For this purpose, various problems can arise such as uncertainties, external disturbances, uncertain dynamics, and measurement noises, which cause deterioration of the fault detection performance by causing false alarms [1-3]. In this respect, fault detection in a robotic manipulator is necessary for monitoring and effective support in utilize of a manipulators as independent systems [4, 5]. Methods of defects detection and isolation are generally founded on the concept of production and residual analysis of the residuals. Many techniques have been assessed in order to be

successfully applied. Taking into account the reliability which must be the most important criterion of the operation. These techniques allow reliable decisions to be made without knowledge of the mathematical system model. In this respect, artificial neural networks (ANNs) are suitable for such problems. One of this remarkable cleanliness is their ability to learn from their environment and improve their behavior from learning, in addition to the learning results in an adaptation (adjustment) of the weights and bias of the neural network [6, 7]. Ideally, after each learning step (iteration), performance improves. There are different learning approaches which differ from each other by the way of adjusting the weights and their structure depends on the architecture of the neural network and the task to be performed. Besides, neural networks have been searched and carried out in real systems [8, 9]; there are many ANN applications in data analysis, identification, and model control [10]. Amid various types of ANN, a multi-layer perceptron (MLP) is quite popular and used extensively in research. In order to achieve good fault compensation, controllers capable of effectively compensating for faults are necessary to enable them to perform their task independently and realistically. In this regard, numerous works have been developed to compensate for defects such as robust control algorithms, including synchronization control [11], ANN [12, 13], sliding mode control (SMC) [14, 15] and time delay control (TDC) [16, 17]. SMC are well known for their robustness against unknown systems dynamics.

To eliminate external perturbations and nonlinear dynamics with delay signal, handy nonlinear control strategies for unmodeled disturbances have been developed, for example TDC. This last, its principal objective is to use past observation of the system response as a control input at the present time to immediately change the control actions instead than identify the parameters or adjust the controller gain of the control system, which leads to an independent model controller i.e., compensation without any use of dynamic model [18]. On the other hand, the big disadvantage of TDC is undesirable tracking errors and time delay estimation (TDE) errors. To compensate errors for TDE, many procedures have been tested by combining commands with a TDC. An auxiliary control [16-19] has been selected to settle its gains adaptively in order to have a switching control scheme. In [20] an auxiliary control in fuzzy sliding mode has also been chattering using fuzzy rules. In general, we caused that several works were realized by combining TDC and neural networks [21].

In other words, in order to eliminate TDE errors, a SMC [22] has been consolidated to allow quick tailoring of switching gains, which improves tracking performance compared to a conventional TDC.
Especially, the use of fixed control gains from the TDC allows to ameliorate a performance of the system and the rapid adaptation of the gain [23]. TDC control combined with sliding mode [24], requires a gain adjustment. However, the TDCs adaptation law does not directly reflect current tracking errors or sliding variables, which leads to a slow convergence rate. Thus, it would be useful to develop a TDC control combined with sliding mode, which compensates for the fault with a fast convergence speed, while suppressing TDE errors and avoiding.

In this paper, a methodology has been developed in order to detect, estimate, and compensate for sensor defects in a robot manipulator, which is based on the concept of residual generation with neural network and compensation with a TDC controller. The network learning is based on nonlinear modeling and allows the approximation of the real model in the absence of defects. The principal target behind this approach is to employ neural networks to observe the robotic system to detect all modification in system dynamics owing to faults. By utilizing the approximation capabilities of neural network, the network can be used for residue generation, in which trained trajectories are considered to perform system operation training. Another trajectory is used to test the efficiency of the network; as a result, the MLP network has the same operation as the faultless system.

The difference between the output of the manipulator with fault and that of the neural network gives us an error which will then be compensated. In other words, from these compared results, a global command switch between two commands has been developed. The first command is applied to the manipulator without fault with a low TDC gain, the result, shows immediate continuation of the trajectory, then, with the same command (law gain), we introduce some default on the manipulator. The results show a divergence (failure of the tracking of the trajectory). For the second command, we use TDC with a high gain in the existence of fault on the robot manipulator. Thus, the robot manipulator successfully tracks the desired trajectory even though the measures are faults. The TDC command used in this paper is free model, that is; it does not use required model, as well as the neural network. This presents a major advantage of the developed command. The simulation results indicate the effectiveness of the proposed controller, which is capable to accurately path the reference trajectory and be robust versus uncertain dynamics and external perturbation. The rest of the paper is organized as follows: Section II describes the formulation of the problem and presentation of the ANN and TDC. Section III deals with control design and convergence analysis, section IV introduce the simulation results. In the end section V concludes the article.

2. RESEARCH METHOD

2.1. Problem formulation

Because of robots are usually involved in remote or dangerous environments; they are faced with external disturbances, uncertain dynamics, and uncertainty. This is why researchers are stimulated to work for solving these problems. For this, we will propose an effective method for detecting and compensating sensor faults, which is: Switched TDC based on neural network for fault detection and compensation in robot manipulators.

2.2. Artificial neural network

In this paper, we are interested in MLP correction of algorithm learning error, the difficult with MLP is to efficiently calculate the weight of hidden layers which give a minimal output error (or zero) as a result, where the augmentation of hidden layers increasing calculation difficulty. The MLP structure made up of three layers: input (M), hidden (N) and output (O) as it shown in Figure 1 [8, 9].

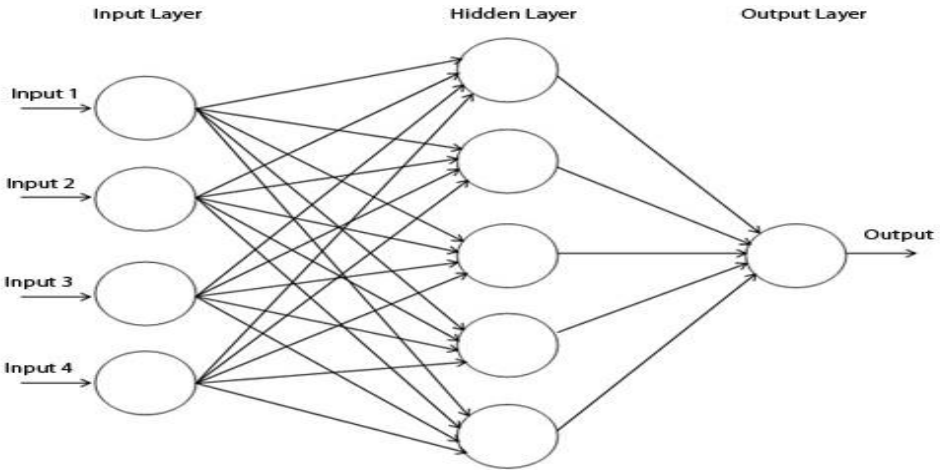


Figure 1. MLP feed-forward neural network

The input vector$X = [x_1, x_2, \dots, x_M]^T$, is changed to an intermediate vector of the hidden variables U by utilizing activation functionf_j, u_j the output of j^{th} neurons of the hidden layer is acquired as (1) [8, 9]:

$$u_j = f_1(\sum_{i=1}^{M} w_{i,j}^1 x_i + b_j^1) \quad (1)$$

where b_j^1 and $w_{i,j}^1$ represent the bias and the weight respectively, between the j^{th} neural of the hidden layer and the i^{th} neural of the input layer. The upper index 1 represents the (first) connection between the neural of the input / hidden layers. The desired output vector $y = [y_1, y_2, \dots, y_0]^T$ of the network is get from, the vector of intermediate variables U, by an activation function f_2 of output layer. For example, the output of the neuron k can be expressed as follows [8, 9]:

$$y_k = f_2(\sum_{l=1}^{N} w_{l,k}^2 + b_k^2) \quad (2)$$

where the upper index 2 denotes the (secondary) connection between neural of the hidden / output layers. There are several forms of the activation functions f_1 and f_2, such as the sigmoid function, the hyperbolic tangent, and the linear function. In MLP, the hyperbolic (tansig) and linear (purelin) tangent activation functions have been used in the hidden layer and the output layer, respectively.

2.3. Dynamic model of a robot manipulator

The dynamic equation of robot manipulator is as (3):

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \tau \quad (3)$$

where $M(q)$the inertia matrix, $C(q,\dot{q})$ is the matrix of Coriolis, and centrifuge $G(q)$ is the vector of gravitational force, $q \epsilon R^n, \dot{q} \epsilon R^n, \ddot{q} \epsilon R^n$are the angle, the angular velocity, and the angular acceleration of the

joints, respectively, and $\tau \epsilon R^n$ is the control input torque. In this paper, the dynamic model of SCARA robot has been used in the simulation [25]. We assume that there is an additive sensor's fault expressed by:

$$q_t = q + \Delta q \tag{4}$$

by replacing (4) in (3) obtains the dynamic equation of manipulator:

$$M(q_t)\ddot{q}_t + C(q_t, \dot{q}_t)\dot{q}_t + G(q_t) = \tau_t \tag{5}$$

where Δq is the sensor fault, q_t is the measured joint and q is the joint without fault. $M(q_t)$, $C(q_t, \dot{q}_t)$, $G(q_t)$ and $\tau_t$is the inertia matrix, the matrix of Coriolis and centrifuge, the vector of gravitational force and the control input torque, we are introducing the fault (additive fault).

2.3.1. Time delay control

In this paper, our objective is to synthesize a control law able to compensate the sensor defect in the manipulator based on the residuals between the real measurements and the neural network model and hence achieve good tracking performances [18]. Multiplying both part of (5) by $M^{-1}(q_t)$and resolving for $\ddot{q}_t$, we obtain:

$$\tau_t = C(q_t, \dot{q}_t)\dot{q}_t + G(q_t) + \big(M(q_t) - \bar{M}(q_t)\big)\ddot{q}_t + \bar{M}(q_t)\ddot{q}_t \tag{6}$$

where t indicates the actual sample and $\bar{M} = diag(\bar{M}_1, \bar{M}_2, \dots, \bar{M}_n)\, \epsilon R^{n\times n}$ is a diagonal positive matrix. We take up the common supposition that the robot dynamics in (5) comply with $\delta_m \leq M(q_t) \leq \delta_M$ [26], for certain positive values $\delta_m$and δ_M. This is since the inertia matrix $M(q_t)$ is voiced in terms of $\sin(q_t)$ and $\cos(q_t)$. From (6) we write a compact and simple form of $\ddot{q}_t$ as (7).

$$\ddot{q}_t = N_t + \bar{M}^{-1}\tau_t \tag{7}$$

where $N_t = -\bar{M}^{-1}[C(q_t, \dot{q}_t)\dot{q}_t + G(q_t)] - \bar{M}^{-1}[(M(q_t) - \bar{M})\ddot{q}_t] \in R^n$, the command purpose is to construct the joint angles $q_t$of a manipulator pursue the reference $q_{ref,t}$accurately, so that the tracking error $e_t = q_{ref,t} - q_t \epsilon R^n$should be as close as possible to zero. Since N_t in (7) is not available, we use its estimate $\hat{N}_t$. The TDC controller is expressed as [18]

$$\bar{\tau}_t = -\bar{M}\hat{N}_t + \bar{M}\big(\ddot{q}_{ref,t} + k_d\dot{e}_t + k_p e_t\big) \tag{8}$$

where $k_d = diag\big(k_{d_1}, k_{d_2}, \dots, k_{d_n}\big) \in R^{n\times n}$ and $k_p = diag(k_{p_1}, k_{p_2}, \dots, k_{p_n}) \in R^{n\times n}$ are positive concept matrices, $\ddot{q}_{ref,t} = [\ddot{q}_{ref_1,t}, \ddot{q}_{ref_2,t}, \dots, \ddot{q}_{ref_n,t}] \in R^n$ is the desired angular acceleration, $\dot{e}_t \epsilon R^n$ is a by-product of the tracking error,$\hat{N}_t = [\hat{N}_{1,t}, \hat{N}_{2,t}, \dots, \hat{N}_{n,t}] \in R^n$. The $\hat{N}_t$ is the estimate of N_t in (7) obtained by a delayed measurement of the sample, called TDE [18, 19]. The expression of $\hat{N}_t$ is given by:

$$\hat{N}_t = N_{t-L} = \ddot{q}_{t-L} - \bar{M}^{-1}\tau_{t-L} \tag{9}$$

where L is a sampling time period, $t - L$ is a sample passed. Replacing (9) in (8), we obtain the following recursive control:

$$\bar{\tau}_t = -\bar{M}\ddot{q}_{t-L} + \tau_{t-L} + \bar{M}(\ddot{q}_{ref,t} + k_d\dot{e}_t + k_p e_t) \tag{10}$$

which is often called TDC [18]. It should be noted that the TDE is bounded, if the control gain $\bar{M}$ is select to gratify the ensuing condition:

$$\|I - M^{-1}(q_t)\bar{M}\| < 1 \tag{11}$$

formally $t \geq 0$, also, the TDE error are limited by constant N_i^* for all $i = 1,2, \dots, n$, i.e., $\big|N_{i,t} - \hat{N}_{i,t}\big| \leq N_i^*$ [27, 28]. It implies as the command gains ought to be select to warrant the bounded ness of TDE errors. Thus, generally, little fixed command gains are utilized to gratify the inequality (11). Besides, if command gains are improperly little, the following performance degrades. Contrary, if those become improperly tall for quick response, those tend to produce a system unsteady.

2.4. Control design and convergence analysis

In this section, we will exploit the control design used in this paper together with the convergence analysis.

2.4.1. Control design

To achieve the control objectives outlined in the previous section, we must first determine the following sliding variable [29, 30]:

$$s_t = \dot{e}_t + K_d e_t \tag{12}$$

Where, $s_t = [s_{t,1}, s_{t,2}, s_{t,3}] \in R^3$, $K_d = diag(K_{d,1}, K_{d,2}, K_{d,3}) \in R^3$ and e_t=$[e_{t,1}, e_{t,2}, e_{t,3}] \in R^3$. It is renowned that K_d in (12) is a conception parameter to be specified for ensuring the stability. In terms of the sliding variable s_t in (12), we build the following command schema [31]. The proposed switching TDC based neural network algorithm is given as follows:

The main objective of this paper is to suggest a global command which switches between two controllers the first one (13) is applied to the manipulator without fault, introducing a low gain. The second controller (14) is applied to the manipulator with a fault, but a large gain has been introduced.

If

$$e_{1,t} \leq \alpha \; and \; e_{2,t} \leq \alpha \; and \; e_{3,t} \leq \alpha$$

then

$$\Gamma_{1,t} = \Gamma_{1,t-L} - M_b \ddot{q}_{t-L} + M_b(\ddot{q}_{ref,t} + K_d \dot{e}_t + K_1 sign(s_t)) \tag{13}$$

else

$$\Gamma_{2,t} = \Gamma_{2,t-L} - M_b \ddot{q}_{t-L} + M_b(\ddot{q}_{ref,t} + K_d \dot{e}_t + K_2 sign(s_t)) \tag{14}$$

And where $sign(s_t) = [sign(s_{1,t}), sign(s_{2,t}), \dots, sign(s_{n,t}) \epsilon R^n$ is defined as

$$sign(s_{i,t}) = \begin{cases} 1 \; if \; s_{i,t} \geq 0 \\ -1 \; if \; s_{i,t} < 0 \end{cases}$$

$K_1 = diag(K_{1,1}, K_{2,1}, K_{3,1}) \epsilon R^3$, $K_2 = diag(K_{1,2}, K_{2,2}, K_{3,2}) \epsilon R^3$ are a positive constant switching gain matrix for guaranteeing stability, $\lambda_{min}(K_2) > \lambda_{max}(K_1)$, λ indicates the proper value of a matrix, $M_b = diag(M_{1,b}, M_{2,b}, M_{3,b}) \in R^3$ is a control gain to be updated in-line according to the adaptive law, and $\alpha$is the threshold not to be exceeded.

It should be noted that the results of the neural network outputs compared to those of the real system outputs generate residuals which are processed as to be compared. Our proposed controller switches between two control schemes. The first one (13) is applied on a system without fault, where the TDC uses a small gain, in order to achieve good tracking performances with a smooth control signal. The second control scheme (14) is developed on a faulty system. In this case we introduce a large gain, in order to compensate the fault and external disturbances effects. The switching between the two schemas based on the residual's values.

2.4.2. Convergence analysis

Consider the dynamic model of the robot manipulator described as follows:

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) + d = \tau \tag{15}$$

where d is the external disturbance. Let us consider that we have a sensors fault expressed in (4). Hence, the dynamic model with sensors fault is given:

$$M(q_t)\ddot{q}_t + C(q_t, \dot{q}_t)\dot{q}_t + G(q_t) + d_1 = \tau \tag{16}$$

where d_1 is the lumped disturbance which is expressed by:

$$d_1 = M(q)\ddot{q} + C(q, \dot{q}) + G(q) - M(q_t)\ddot{q}_t - \left(C(q_t, \dot{q}_t)\dot{q}_t + G(q_t)\right) + d \tag{17}$$

Therefore $N_t$is given by:

$$N_t = -\bar{M}^{-1}(C(q_t, \dot{q}_t)\dot{q}_t + G(q_t) + d_1) - \bar{M}^{-1}[(M - \bar{M})\ddot{q}] \tag{18}$$

Then the TDE is used to estimate N_t, which allows us to evaluate the dynamics model without requiring any prior knowledge neither on the system nor on the sensors fault. Furthermore, TDC can use this estimation to compensate the effect of the lumped disturbances and achieve a good tracking performance.

Theorem: Consider the dynamic model of robot (3) subjected to external disturbances and sensors fault. The control law given by (13) and (14), guarantees the convergence asymptotically of the tracking error and the tracking error rate.

Proof: In order to prove above theorem, we use the Lyapunov function applicant as specified:

$$v = \frac{1}{2} s_t^T s_t = \frac{1}{2} \sum_{i=1}^{3} (s_{t(i)}^2) \tag{19}$$

Its time derivative is given by:

$$\dot{v} = s_t^T \dot{s}_t \tag{20}$$

Using (12) and (20) lead to:

$$\dot{v} = {s_t}^T (\ddot{e}_t + K_d \dot{e}_t) = s_t^T (\ddot{q}_{ref,t} - \ddot{q}_t + K_d \dot{e}_t) \tag{21}$$

Replacing in (21) $\ddot{q}_t$by its expression deduced from (7), it comes to:

$$\dot{v} = {s_t}^T (\ddot{q}_{ref,t} + K_d \dot{e}_t - \bar{M}^{-1} \tau_t + N_t) \tag{22}$$

By applying the control law given by (13) and (14) we obtain:

$$\dot{v} = {s_t}^T (\ddot{q}_{ref,t} + K_d \dot{e}_t + N_t - \hat{N} - (\ddot{q}_{ref,t} + K_d \dot{e}_t + K_2 sign(s_t))) \tag{23}$$

We obtain:

$$\dot{v} = {s_t}^T (N_t - \hat{N} - K_2 sign(s_t)) \tag{24}$$

Since $||N_t - \hat{N}||_\infty \leq N^*$ as demonstrated in [27, 28]; hence

$$\dot{v} \leq \sum_1^3 |s_i| \, (N^* - K_{2,i}) \tag{25}$$

where $K_{2,i}$ is ith value of the diagonal matrix K_2. Hence, if we choose $k_{2,i} > N^*$ then $\dot{V}$ becomes defined negative. Consequently, from the definition of v we can conclude that s_t converge asymptotically to zero:

$$s_t = \dot{e}_t + K_d e_t \to 0, as\ t\ \to \infty \tag{26}$$

As a result, the tracking error and the tracking error rate converge asymptotically to zero:

$$e_t \to 0, \dot{e}_t \to 0\ as\ t\ \to \infty \tag{27}$$

3. SIMULATIONS AND RESULTS

We considered a manipulator arm with 3 degrees of freedom, called SCARA as described in section 2 of this paper. The command purpose here is to force the robot to follow a reference trajectory created in advance. The robot's equations and parameters as in (1) are given in [25]. The manipulator's parameters are $m_1 = 0.5, m_2 = 0.3, m_3 = 0.1\ (Kg)$. The length of the links is set to be $L = 1m$, the moments of inertia are $I_1 = 0.02,\ I_2 = 0.03, I_3 = 0.05\ (rad/s)$ and the sampling time is equal to $t_s = 0.01\ s$. In the following two sections, we will present the simulations and results of the proposed method for the detection and compensation of faults in the robot arm. These methods were presented in the previous sections of this paper.

3.1. Simulation results of the fault detection

In order to detect the faults in our system (the SCARA robot arm), we have chosen to use the following neural network design: One input layer that contains 9 neural (three articular positions, three articular velocities and three articular couples measured in t), one hidden layer, with 15 neural, and two

output layers with six neural (three articular positions and three articular velocities at (t+Δt)). The MLP was designed with a back propagation algorithm. The learning set is made by simulating 4 flawless spherical trajectories, introducing the input/output of each trajectory. After the training, the model obtained is then tested and validated using a fifth spherical trajectory by introducing only the input. As a result, the trajectory was immediately followed, as the MLP produced a copy of the dynamic manipulator behavior, in which the neural network has the same functionality as the manipulator. The simulations results are given in Figures 2-5.

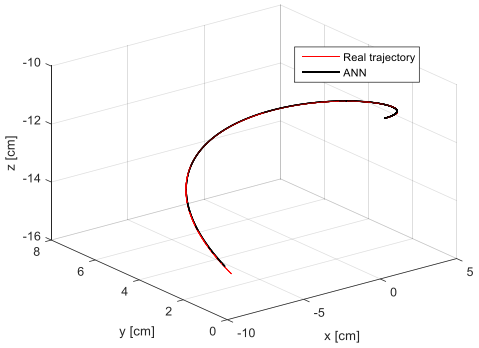


Figure 2. Tracking trajectory in task space

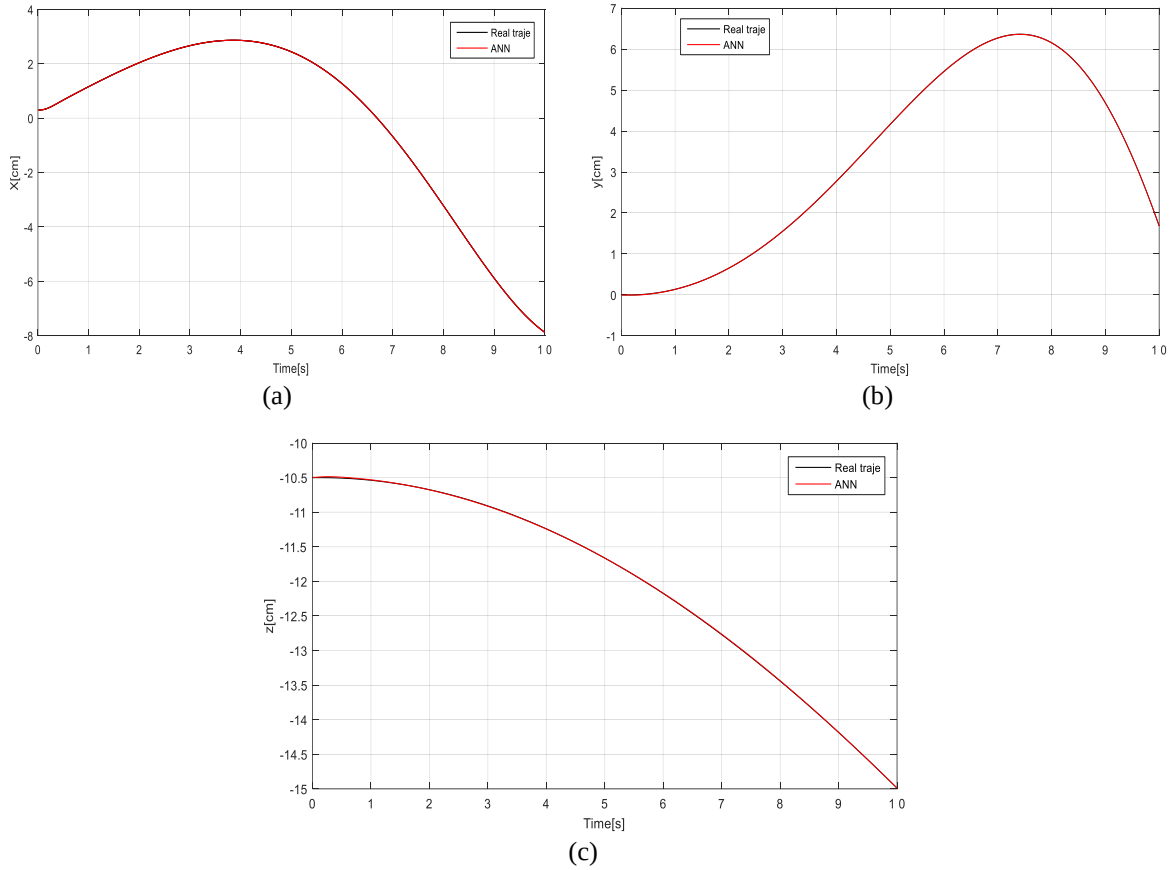


Figure 3. Tracking trajectory for **(**a) the x-axis, (b) the y-axis, and (c) the z-axis

Where Figure 2 represents the set trajectory to be followed together with the neural network output trajectory, it is noted that the neural network output trajectory follows exactly that of the manipulators. Figure 3 (a, b, c) represents the real and neural network trajectories following the x-axis, y-axis, and z-axis, respectively. It is clear again, that the neural network output trajectory is the same of that of the real robot

manipulator. The MLP is set to generate accurate models of the system under normal (without fault) operating conditions. The comparison between the output of the network $y_1(j,i)$ and the output of the system $q(j,i)$ (with fault) gives the error vector $e_j(i) = y_i(j,i) - q(j,i)$ with $j = 1,\dots,3, i = 1,\dots,n, n = 1000$.

The suggested procedure is executed on-line so that faults can be detected right away. In order to simulate a sensor fault, we add $\Delta q = 0.001°$as a failure on a sensor which leads to a significant reduction or rise in the torque, resulting in anomalous variations of the residue. This residue is compared to some set values, and if the comparison is positive, it conducts to the detection of the fault. Once the fault detection has been carried out, one proceeds to the fault compensation using the TDC which is presented in the following section.

3.2. Simulation result of compensation

Three sets of simulations have been carried out in order to assess the TDC method.

- **Case 1 (fault-free)**

This case is considered the nominal one, where the system is not subjected to any faults (Δq=0). We will see how our method behaves when no faults are present on our system. The TDC controller's gains are chosen as follows $\alpha = 0.2$, $M_b = \text{diag}[0.03,0.03,0.03]$, $K_d = [1.9,1.9,1.9]$, $K_1 = \text{diag}[1.5,1.5,1.5]$and $K_2 = \text{diag}[6.1,6.1,6.1]$. The simulations results are shown in Figures 4 and 5. Figure 4 presents the ser trajectory to be followed and the robot's real trajectory (in 3D). It is shown that both trajectories are superimposed, which shows that the TDC method has not intervened negatively in the fault-free case. Figure 5 also show that we have obtained good tracking performance (on the x, y, and z axis respectively) when using the TDC method, while the torques stay in admissible values Figure 6.

- **Case 2 (faulty case)**

In order to evaluate the performances of the controller (13) in faulty environment, we have introduced sensor faults as follows: Δq=0.01° in joints 1 and 2. The results of the simulation of the suggested command are illustrated in Figures 7 to 9.

From Figures 6-9, it is clear that the controller given by (13) does not give acceptable results, as the robot does not track well the proposed trajectory. We observe that there is an error between the set trajectory and the robot's trajectory. Therefore, we precede a change in the controller's parameters as given in (14). This change in parameters has been applied in the following section.

- **Case 3 (faulty case, with parameter change)**

In this case, the proposed switched TDC based on neural network is introduced to overcome the drawbacks of controller (13). The proposed scheme uses the controller (12) in the scenario where there were no sensors faults, and switches to the controller (14) in the case of detecting important errors.

The simulations results for the proposed controller are demonstrated in Figures 10 to 12. We can clearly see that at the time of the introduction of the fault, the robot's trajectory changes Figure 10, to be corrected soon after by the proposed TDC method. This shows that the proposed controller is working well. Indeed, soon after introducing a defect, we switch to the controller (14) which is programmed with a large gain. The latter provides an important help to improve tracking performance. We can also see that the controller is effective in eliminating chattering (according to the Figures 10 to 12). In addition, it is obvious that the trajectory tracking has been improved by comparison to the controller (13).

The main goal of this paper is to develop a method which allows detecting and compensating of sensor defect that often affects the manipulator joints. So, we proposed an MLP neural network, which allows to learn the functioning of the system (manipulator) without faults, as it was shown in Figure 2 and 3 perfect learning of the system function. Once he has learned how to operate the system without faults, we injected it with disturbances at the manipulator outputs (position and speed) in order to compensate them later. Then, we developed two controllers (13 and 14) which allow us to compensate this defect. Controller 13 is applied to the system (without fault) as it was shown in Figure 4 and 5; where it is noted a divergence when the fault is introduced, continuation of the trajectory is lost Figure 7 and 8. This result allowed us to think of developing another controller with a significant gain in order to compensate for the defect. The controller 14 which allows us to have a rapid compensation for the continuation of the trajectory even in the presence of a fault as it was shown in Figure 10. The principle of this step is to switch between the controllers 13 and 14, in presence or absence fault based on neural networks.

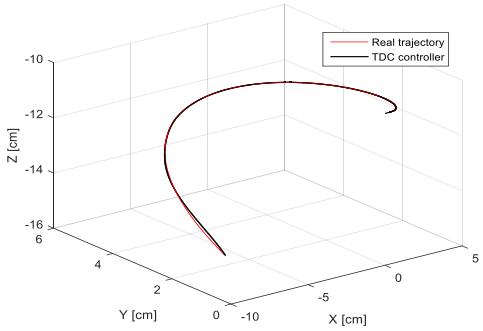


Figure 4. Tracking trajectory in task space of Case 1

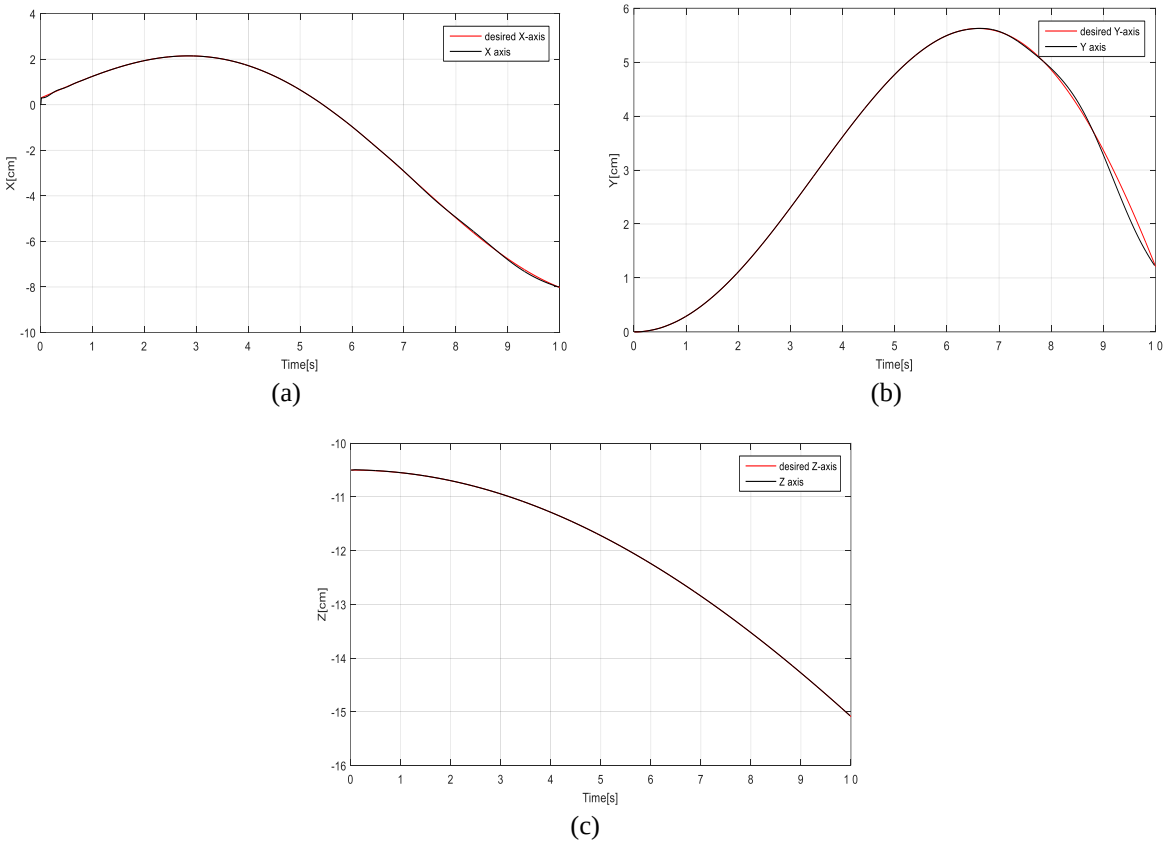


Figure 5. Tracking trajectory for (a) x-axis, (b) y-axis, and (c) z-axis of Case 1

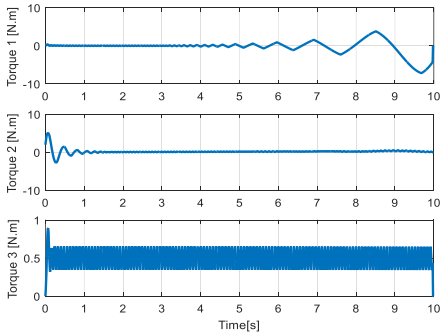


Figure 6. Required torque for Case 2

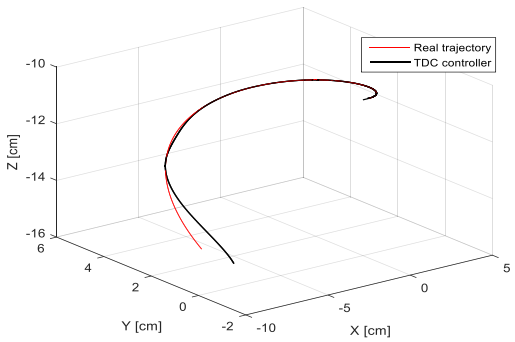
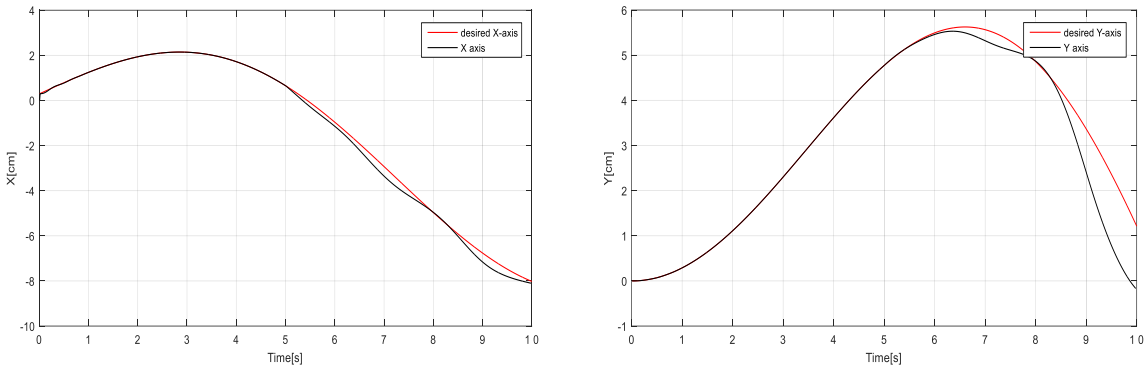
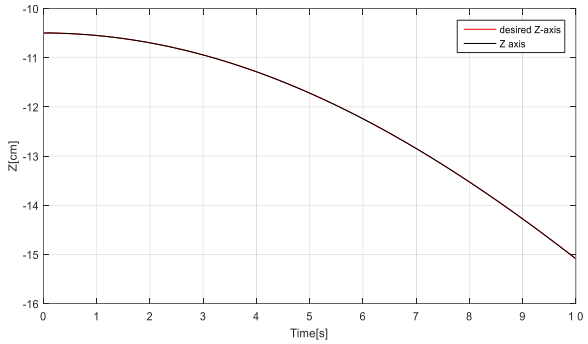


Figure 7. Tracking trajectory in task space in Case 2



(a) (b)



(c)

Figure 8. Tracking trajectory for (a) x-axis, (b) y-axis, and (c) z-axis in Case 2

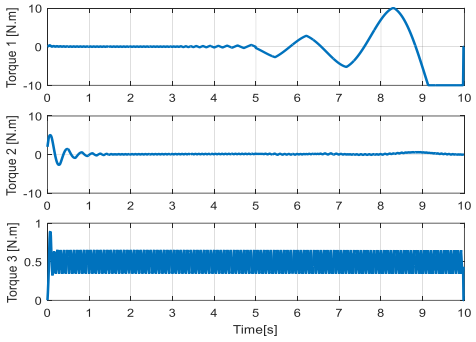


Figure 9. Require torque in Case 3

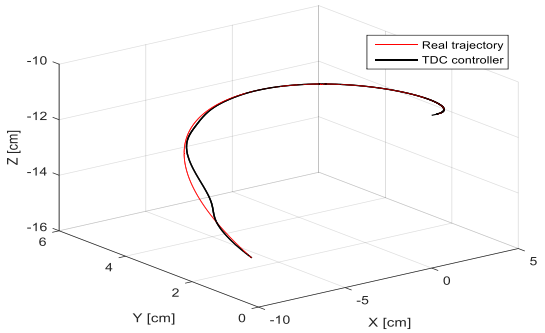
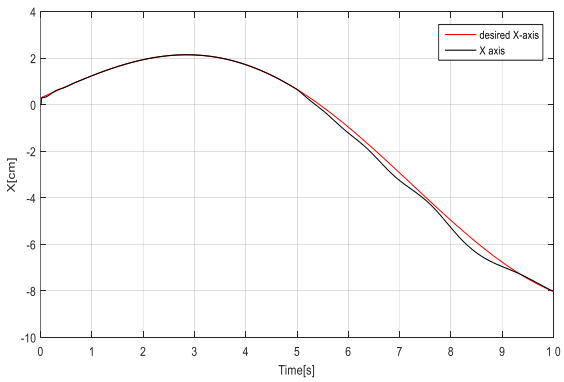
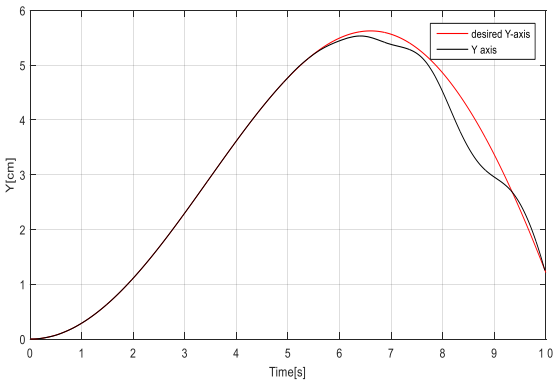


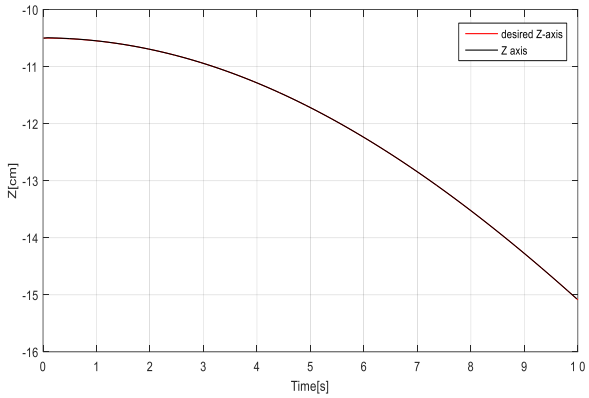
Figure 10. Tracking trajectory in task space in Case 3



(a)



(b)



(c)

Figure 11. Tracking trajectory for (a) x-axis, (b) y-axis, and (c) z-axis in Case 3

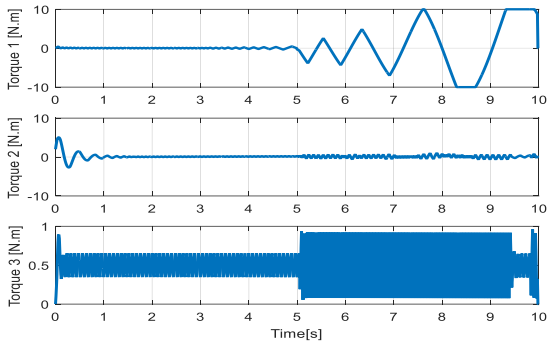


Figure 12. Require torque

4. CONCLUSION

In this paper, a new concept of fault detection and compensation based on neural networks and TDC, on a SCARA robot arm has been developed. It consists of a reliable detection and robust compensation in the presence of faults on the sensors. It is important to note that the combination of neural networks and TDC is very important to diagnose in a system. For this purpose, neural network modeling was used for the generation of residues, in which several trajectories were driven in order to model the different functioning cases of the system. Another trajectory was used to test the efficiency of the network and validate it. This results in MLP model that has the same behavior of the faultless system. By making the difference between the output of the robot with faulty sensors and that of the network mode, we obtain an error which should be compensated. From these errors, a global control switch (TDC) between two commands has been developed. The first command (TDC) is applied to the faultless robot with a low gain, which showed a good tracking of the trajectory. Then, with the same command (low gain), we introduce a fault on the robot, which shows the divergence (failure to follow the trajectory). A second control was developed, with a high gain in the presence of a fault. The results of the simulations show that the robot arm followed the desired trajectory successfully. The TDC command used in this paper is model free, i.e., it does not use a specific model, which represents a major advantage of the proposed command. This study shows that the developed approach can produce good detection and compensation results for a nonlinear system exposed to faults. Moreover, the controller conception is capable of stabilizing the error trajectories of the system in a finite time. The robustness, stability, and perturbation rejection of the proposed controller have also been demonstrated.

REFERENCES

[1] J. Wünnenberg and P. M. Frank, "Dynamic model based incipient fault detection concept for robots," in *IFAC Proceedings Volumes*, vol. 23, no. 8, pp. 61-66, 1990.

[2] K. S. Gaeid, A. F. Nashee, I. A. Ahmed, and M. H. Dekheel, "Robot control and kinematic analysis with 6DoF manipulator using direct kinematic method," *Bulletin of Electrical Engineering and Informatics*, vol. 10, no. 1, pp. 70-78, 2020.

[3] A. M. Al-Ghaili, *et al*., "A review of anomaly detection techniques in advanced metering infrastructure," *Bulletin of Electrical Engineering and Informatics*, vol. 10, no. 1, 2021.
[4] M. PDRS. MAL 2102, "Payload Deployment and Retrieval System Malfunction Workbook," *NASA Technical manual, NASA Johnson Space Center*, Houston, TX, Apr, 433-9, 1988.
[5] F. El Hammouchi, L. El Menzhi, and A. Saad, "Diagnosis Methods for Wind Turbine Doubly Fed Induction Generator under Grid Defects," *International Journal of Information Science and Technology*, vol.3, no. 3, pp. 46-55, 2019.
[6] J. B. Mbede, W. Wei, and Q. Zhang, "Fuzzy and recurrent neural network motion control among dynamic obstacles for robot manipulators," *Journal of Intelligent and Robotic Systems*, vol. 30, no. 2, pp. 155-177, 2001.
[7] W. Setiawan, M. I. Utoyo, and R. Rulaningtyas, "Reconfiguration Layers of Convolutional Neural Network for Fundus Patches Classification," *Bulletin of Electrical Engineering and Informatics*, vol.10, no. 1, PP. 383-389, 2021.
[8] K. P. Sridhar, B. Vignesh, S. Saravanan, M. Lavanya, and V. Vaithiyanathan, "Design and Implementation of Neural Network Based circuits for VLSI testing," *World Applied Sciences Journal*, vol. 29, pp. 113-117, 2014.
[9] S. Ibrahim, N. A. Zulkifli, N. Sabri, A. A. Shari, and M. R. M. Noordin, "Rice grain classification using multi-class support vector machine (SVM)," *IAES International Journal of Artificial Intelligence*, vol. 8, no. 3, pp. 215-220, 2019.
[10] M. A. Ahmad, H. Ishak, A. N. K. Nasir, and N. Abd Ghani, "Data-based PID control of flexible joint robot using adaptive safe experimentation dynamics algorithm," *Bulletin of Electrical Engineering and Informatics*, vol. 10, no. 1, pp. 79-85, 2021.
[11] S. J. Cho, M. Jin, T. Y. Kuc, and J. S. Lee, "Stability guaranteed auto-tuning algorithm of a time-delay controller using a modified Nussbaum function," *International Journal of Control*, vol. 87, no. 9, pp. 1926-1935, 2014.
[12] A. Abe, "Trajectory planning for flexible Cartesian robot manipulator by using artificial neural network: numerical simulation and experimental," *Robotica*, vol. 29, no. 5, pp. 797-804. 2011.
[13] N. Nikdel, P. Nikdel, M. A. Badamchizadeh, and I. Hassanzadeh, "Using Neural Network Model Predictive Control for Controlling Shape Memory Alloy-Based Manipulator," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 3, pp. 1394-1401, 2014.
[14] S. Islam and X. P. Liu, "Robust Sliding Mode Control for Robot Manipulators," *IEEE Transactions on Industrial Electronics*, vol. 58, no. 6, pp. 2444-2453, 2011.
[15] J. Baek and M. Jin, S. Han, "A New Adaptive Sliding-Mode Control Scheme for Application to Robot Manipulators," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 6, pp. 3628-3637, 2016.
[16] S. Yu, X. Yu, B. Shirinzadeh, and Z. Man, "Continuous finite-time control for robotic manipulators with terminal sliding mode," *Automatica*, vol. 41, no. 11, pp. 1957-1964, 2005.
[17] M. Galicki, "Finite-time control of robotic manipulators," *Automatica*, vol. 51, pp. 49-54, 2015.
[18] K. Youcef-Toumi and O. Ito, "A time delay controller for systems with unknown dynamics," *ASME The American Society of Mechanical Engineers*, vol. 112, no. 1, pp. 133-142, 1990.
[19] S. J. Cho, M. Jin, T. Y. Kuc, and J. S. Lee, "Control and synchronization of chaos systems using time-delay estimation and supervising switching control," *Nonlinear Dynamics*, vol. 75, no. 3, pp.549-560, 2014.
[20] H. J. Bae, M. Jin, J. Suh, J. Y. Lee, P. H. Chang, and D. S. Ahn, "Control of robot manipulators using time-delay estimation and fuzzy logic systems," *Journal of Electrical Engineering & Technology*, vol. 12, no. 3, pp. 1271-1279, 2017.
[21] A. Wu and Z. Zeng, "Exponential Stabilization of Memristive Neural Networks with Time Delays," in *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 12, pp. 1919-1929, 2012.
[22] J. Baek, M. Jin, and S. Han, "A New Adaptive Sliding-Mode Control Scheme for Application to Robot Manipulators," in *IEEE Transactions on Industrial Electronics*, vol. 63, no. 6, pp. 3628-3637, 2016.
[23] S. J. Cho, M. Jin, T. Y. Kuc, and J. S. Lee, "Stability guaranteed auto-tuning algorithm of a time-delay controller using a modified Nussbaum function," *International Journal of Control*, vol. 87, no. 9, pp. 1926-1935, 2014.
[24] M. Jin, J. Lee, and N. G. Tsagarakis, "Model-Free Robust Adaptive Control of Humanoid Robots with Flexible Joints," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 2, pp. 1706-1715, Feb. 2017.
[25] A. Nagchaudhuri, S. Kuruganty, and A. Shakur, "Introduction of mechatronics concepts in a robotics course using an industrial SCARA robot equipped with a vision sensor," *Mechatronics*, vol. 12, no. 2, pp.183-193, 2002.
[26] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modeling and Control*. Hoboken, NJ, USA: Wiley, 2005.
[27] T. C. Hsia and L. S. Gao, "Robot manipulator control using decentralized linear time-invariant time-delayed joint controllers," in *Proceedings, IEEE International Conference on Robotics and Automation*, Cincinnati, OH, USA, vol. 3, pp. 2070-2075, 1990.
[28] J. Baek, S. Cho, and S. Han, "Practical Time-Delay Control with Adaptive Gains for Trajectory Tracking of Robot Manipulators," in *IEEE Transactions on Industrial Electronics*, vol. 65, no. 7, pp. 5682-5692, 2018.
[29] V. Utkin, J. Guldner, J. Shi, *Sliding mode control in electro-mechanical systems*. CRC press, 2017.
[30] M. Jin, J. Lee, P. H. Chang, and C. Choi, "Practical Nonsingular Terminal Sliding-Mode Control of Robot Manipulators for High-Accuracy Tracking Control," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 9, pp. 3593-3601, Sept. 2009.
[31] S. J. Cho, M. Jin, T. Y. Kuc, and J. S. Lee, "Control and synchronization of chaos systems using time-delay estimation and supervising switching control," *Nonlinear Dynamics*, vol. 75, no. 3, pp.549-560, 2014.

BIOGRAPHIES OF AUTHORS

Maincer Dihya was born in Tizi Ouzou, Algeria, on January 13, 1988. Currently a PhD student on control engineering 5th inscription at the University of Science and Technology Houari Boumediene, Algiers, Algeria under the supervision of Professor Moufid Mansour. A member in Instrumentation and Automatic Control Department, Laboratory of Robots Parallelism Electroénergetic University of Science and Technology Houari Boumediene, Algiers, Algeria since 2015. Research interests include conducted on iterative learning control, sliding mode control, time-delay control, and robotics. The actual research subject is about "Detection and diagnosis of sensor and actuator faults for a robotic system".

Moufid Mansour, currently professor at the University of Science and Technology Houari Boumediene, Algiers, Algeria, Faculty of Electronics and Computer Science, Department of Instrumentation and Control. He is affiliated at the Laboratory of Robots Parallelism Electroénergetic University of Science and Technology Houari Boumediene, Algiers, Algeria. His research field interests on MATLAB simulation, electrical and electronics engineering, automation, control and instrumentation, control theory, automation and robotics, system modeling, robotics, system identification and system dynamics modeling.

Chems Eddine Boudjedir, PhD. in Control Engineering at the polytechnics national school of Algiers (ENP), Algeria, affiliated at Laboratory of Process and Control, Polytechnic National School, Algiers, Algeria. He is interested on centered on iterative learning control, sliding mode control, time-delay control, and robotics.

Bounabi Moussaab, PhD on Modelling and management photovoltaic systems since 2019 from the polytechnics national school of Algiers (ENP), Algeria. He is a member of Photovoltaic Communication and Conversion Devices Laboratory, Polytechnic National School, Algiers, Algeria. His research interests include power electronics, power generation, power converters, inverters, control instrumentation, PLC programming, power conversion, harmonics, grid integration, microgrids optimization, PIC programming.