Robust Adaptive Tracking Control of Manipulator Arms with Fuzzy Neural Networks

Madour Fouzia
Department of Electronics
University of Ferhat Abbas
Setif, Algeria
madour_fouzia@yahoo.fr

Nabil Khenfer
Department of Electronics
University of Ferhat Abbas
Setif, Algeria
khenfer_n@yahoo.fr

Naceur-Eddine Boukezzoula
Department of Electronics
University of Ferhat Abbas
Setif, Algeria
nasrbou@yahoo.fr

Abstract—**The learning space for executing general motions of a flexible joint manipulator is quite large and the dynamics are, in general, nonlinear, time-varying, and complex. The objective of this paper is to design a nonlinear system based on the fuzzy neural network control using supervised training, into executing reference trajectories by a flexible joint manipulator. The structure identifications of controller networks are performed by using the Adaptive Neural Fuzzy Inference System (ANFIS), with new parameters and weight coefficients automatically adapted and adjusted, in order to decrease position tracking errors. In order to adapt and reduce the number of undefined parameters in the network, a new technique is used. Reported research works use the Euler method for the resolution of the arm's dynamic function, in this paper, a more exact method was used, represented by the Fourth-Order Runge-Kutta (RK4) method. A comparative study has been carried out between these two methods in order to prove the effectiveness of the later. Finally, in order to test the robustness of the proposed approach, it was also investigated considering parameter variations. The tracking speed of the model on the system control accuracy was also analyzed. The simulation results show that the proposed approach has a good tracking effect.**

Keywords-Adaptive Neuro Fuzzy Interface System (ANFIS); Fuzzy Neural Network (FNN) control; manipulator robot; supervised training; trajectory tracking; robustness

I. INTRODUCTION

Robotic manipulators have reshaped industrial processes. Pressing requirements of improved and enhanced productivity in industrial applications have necessitated the deployment of robots to automate tasks [1]. Some of the most common tasks for which robots are designed to are: transportation and material handling [2, 3], assembly and manufacturing [4], and point and arc welding [5, 6]. Robotic manipulators can be categorized as rigid or flexible. Rigid manipulators increase tensile strength with more precision due to the materials used such as steel or aluminum frames but have high cost and weight [7]. The advancements in material technology have enabled us to acquire low cost and weight flexible joint robotic manipulators, but quite often an oscillation is produced throughout the manipulator arm, posing a great challenge in terms of manipulator's control [8]. Linear control techniques (e.g. Proportional-Integral-Derivative (PID)) are valid when a robot moves slowly and can be modeled by linear differential equations with constant coefficients [9]. The linear control fails when there is a need to tackle complex system dynamics [8], nonlinearities and parameter changes in the control laws, which require robust or adaptive control laws to handle the trajectories [9]. Many nonlinear control strategies have been developed to deal with the nonlinearities present in the system, such as robust control [10], optimal control [11], adaptive control [12], nonlinear control [13, 14], and intelligent control [15]. In [9], authors present a systematic review of control strategies for multi-Degree of Freedom (DoF) robotic manipulators, using one of the most commonly used modeling formulas, Euler Lagrange. A review of modeling of serial articulated robotic arms was recently presented in [7]. Kinematics is usually derived using Denavit-Hartenberg (D-H) parameters. The problem of robust control law design for accurate trajectory tracking of a flexible joint manipulator is addressed in [14], by considering joint flexibility and actuator dynamics. The system's model has been derived using the Euler-Lagrange approach. Authors in [16] proposed an iterative control law to regulate the set-point of a flexible robotic system that is driven electrically and is subjected to model uncertainty. Most of the reported techniques suffer from limitations related to the level of complexity resulting from iterative differentiations of nonlinear virtual functions and thus leading to complex and computationally expensive algorithms.

The arm positioning is commonly performed by a fuzzy controller. Regarding the fuzzy logic control, it incorporates in a control system the way human beings think. Through fuzzy technology, the human operator experience, which controls processes and industrial plants, is captured in order to be included in computerized controllers with the same or better performance than humans. The fuzzy control does not need mathematical modeling of the process, but the modeling of actions from the knowledge of a specialist, using linguistic terms, i.e. verbal descriptions. Moreover, fuzzy controllers also handle linear and nonlinear systems and are able to control complex multivariable systems [17]. But the fuzzy controller is unable to learn, in contrast to the neural networks. With favorable generalization ability and relatively simple structure, a neural network is not involved with complicated mathematical calculations and is characterized as the approximation to any nonlinear function with arbitrary precision [18]. Therefore, it can be combined in many aspects

Corresponding author: Madour Fouzia

of nonlinear research. From another point, the neural networks need minimum information to perform learning [19]. For that reason, we have used the hybridization of these two approaches in order to exploit their advantages. This hybridization allowed us to improve the performance of the network controller and achieve satisfactory results through online learning [20, 21]. The dynamics of a robot manipulator are, in general, nonlinear time-varying and complex [22]. Therefore, most controllers are not able of effectively controlling movements of a flexible joint manipulator under different distance, velocity, and load requirements [23], and the use of a complicated nonlinear dynamic model makes real-time implementation difficult [24]. Moreover, it is by no means an easy task to identify the model parameters accurately. In this work, two network controllers based on fuzzy neural network control using supervised training are proposed, where each network will control one joint of the arm manipulator with two DoF. Parameters and structure identifications of network are necessary in any modeling which aims to achieve a generalized model. ANFIS employs well-known parameter-identification techniques [25], where the determination of the optimal number of fuzzy rules will be determined by the identification scheme of the structure [26]. For most of fuzzy controllers, the rule composition is very difficult, it needs experts' knowledge or perspective insight of system behavior [27]. We propose a solution based on supervised training, where each synaptic weight between the third and the fourth hidden layer of each controller network represents a fuzzy rule [20].

There are several learning algorithms. They can be classified into two main categories: supervised and unsupervised. In unsupervised learning, the system learns about the pattern from the data itself without a priori knowledge. On the other hand, supervised learning is guided learning, i.e. when the network is formed it compares the input and the desired result which is represented in our case by the reference joint. However, in this case, new parameters and weight coefficients are automatically adapted and adjusted, through online error learning, in order to decrease the position tracking error. Meanwhile, the control system is analyzed. One of the major difficulties of this learning type is the local minimum error [28]. In order to overcome this problem, changes in the parameters of the learning law such as the learning step can be carried out. In the process, the resolution of the arm's dynamic function was performed by two comparative methods, RK4 and Euler method. Finally, control laws were subjected to various test inputs in a simulation to characterize the tracking performance, and some results are illustrated to show the validity of the proposed approach.

II. PROPOSED APPROACH

In this approach, the fuzzy system is used to represent the abstract controller program, and the neural network is used to manage the parameters specifying the fuzzy rules, using learning and knowledge of sampled movements [18, 29]. The fuzzy system encodes knowledge by qualitative rules rather than a precise quantitative description [30]. However, the use of fuzzy qualitative rules is more effective and tends to cover a wider learning space. The Fuzzy Neural Network (FNN) controller shown in Figure 1 modulates the motion command via sensory feedback and uses the resultant signal to move the robot manipulator. First we have a reference movement block. This movement is defined by the angle and the angular speed assigned to each articulation of each rod of the manipulator arm. These sampled data are then processed in parallel by two FNN controllers to produce two outputs, torque τ_1 and torque τ_2, applied to the manipulator arm which is simulated by its dynamic function. Finally, a sensory feedback is carried out in order to calculate the error in position and angular speed. Proper learning process is also imperative for determining the weights in the networks. Parameters specifying the fuzzy rules for governing sampled movements are stored and manipulated by the n6eural network to process a wide range of movements.

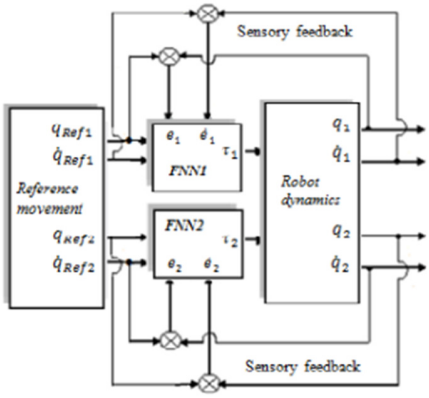

Fig. 1. Diagram block of robot learning control.

The success of the proposed scheme depends on designing the FNN to generate proper motion commands for various motions. As indicated in the block diagram in Figure 1, we use the feedback error of position (e) and error of velocity ($\dot{e}$) between the reference and actual joints, denoted as:

$$e = \frac{1}{2}\sum_{j=1}^{n}(q_{Ref} - q)^2 \quad (1)$$

$$\dot{e} = \frac{1}{2}\sum_{j=1}^{n}(\dot{q}_{Ref} - \dot{q})^2 \quad (2)$$

The performance of the trained model is evaluated while decreasing the error between the reference trajectory and the trajectory tracked by the robot manipulator where *n* represents the number of training examples.

III. CONTROLLER DESIGN

A fuzzy neural system is a fuzzy system presented in the network architecture. Parameter identification of the fuzzy system is performed by an automatic learning technique of an adaptive neural network [21, 31]. ANFIS is one of the well-known neural fuzzy approaches and it has shown good results in the modeling of complex non-linear problems [20, 21]. It must be noted that the application of ANFIS is not limited only to systems difficult for modeling, but we can also use this method to design controllers easier. Different learning techniques have been developed for the tuning of the inputs and consequent parameters in the ANFIS [32, 33]. In this study, the FNN system employed the learning, because it is the most

appropriate method for the ANFIS architecture. An initial Takagi-Sugeno (T-S) fuzzy inference system [34] is developed by defining the input and output membership functions. The determination of the optimum number of rules is very important in fuzzy modeling, it can be determined by a structure identification scheme [26], and the parameters of the initial fuzzy inference system are tuned in ANFIS architecture. Finally, the performance of the trained model is evaluated while decreasing the error between the reference trajectory and the trajectory tracked by the robot manipulator. As mentioned above, two network controllers are proposed in this study, where each FNN system is implemented as shown in Figure 2. The parameters of fuzzy reasoning are expressed by the connection weights or node functions of the neural network [35]. In Figure 2, the inputs of the FNN are the reference position and the velocity trajectories of a sampled motion, and the output is the motion command. The reason of choosing this structure is that the numbers of fuzzy rules and membership functions for input and output are pre-specified in this type of FNN. Thus, for various motions there are the same numbers of fuzzy parameters with the same attributes. Consequently, these fuzzy parameters are appropriate to be generalized by the neural network in order to deal with a motion.

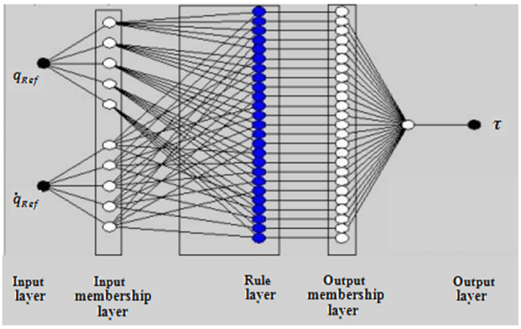

Fig. 2. The structure of the FNN.

The structure adopted of each FNN, consists of five layers of nodes, which are of the same type within the same layer. Each of the five layers performs one stage of the fuzzy inference process, as described below:

A. Layer 1

This layer is the input layer, and inputs are transmitted to the next layer directly without any calculation. In Figure 2, we can see two input nodes q_{Ref} and $\dot{q}_{Ref}$ of motions of a single DoF.

B. Layer 2

Each input node in this layer is an adaptive node which produces the membership grade of linguistic label. It is a fuzzy layer. Each node j in this layer has a node function O_j^2.

$$O_j^2 = \mu(x)$$

$$\mu(x) = exp\left[-\left(\frac{x-\theta}{\sigma}\right)^2\right]$$

$$\mu: x \to [0,1] \quad (3)$$

where μ is the degree of the Gaussian membership function, x is the input to the node j, θ is the center of membership functions, and σ the width of membership functions.

In this paper, it is chosen that q_{Ref}, $\dot{q}_{Ref}$ and the torque τ have the same kind of membership function, where τ is composed by seven fuzzy subsets, NB, NM, NS, ZE, PS, PM, PB: N represents negative, P positive, S small, M medium , B large and ZE zero. q_{Ref} and $\dot{q}_{Ref}$ are composed by five fuzzy subsets, as shown in Figure 3, which implies 5×5 = 25 nodes in the rule layer.

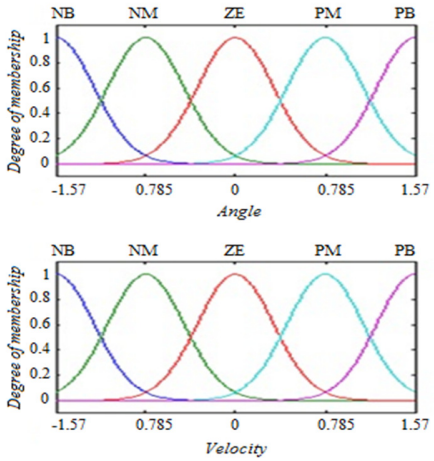

Fig. 3. Degree of membership functions.

C. Layer 3

This layer is intended for the implementation of the fuzzy rules. Each node in this layer corresponds to a rule, which is defined as a fuzzy conditional statement of the form rule [36, 37]:

$$\text{If } \boldsymbol{x} \text{ is } A \text{ and } \boldsymbol{y} \text{ is } B \text{ Then } \boldsymbol{z} \text{ is } C$$

where $\boldsymbol{x}$ and $\boldsymbol{y}$ are fuzzy sets representing the inputs q_{Ref} and $\dot{q}_{Ref}$. z represents the output τ, and A, B, and C represent linguistic variables. In this layer, the membership functions are multiplied and the output is:

$$O_j^3 = \mu(x).\mu(y) \quad (4)$$

where $\mu(x)$ and $\mu(y)$ denote the degree of membership functions. A control rule for each network is described as follows:

$$\text{If } \boldsymbol{q_{Ref2}} \text{ is NB and } \boldsymbol{\dot{q}_{Ref2}} \text{ is NB then } \boldsymbol{\tau_2} \text{ is NB.}$$

D. Layer 4

In T-S model approach, the output model is a combination of the input variables according to fuzzy rules. The use of this model allows us to obtain an output according to the input variables [17, 34]. The fuzzy rules are structured as follows:

$$R^i: \textbf{if } [\, x_1(k) \text{ is } F_1^i \,] \,\&\, \ldots \,\&\, [\, x_{nx}(k) \text{ is } F_{nx}^i \,]$$

$$\textbf{then } y^i(k) = p_0^i + p_1^i x_1(k) + p_{nm}^i x_{nm}(k)$$

$$i = 0, 1, \ldots, nr, \quad (5)$$

where R^i is the *i*-th inference rule, x_j the *j*-th input variable, F_j the fuzzy set defined on the universe of discourse of the variable x_j, used in *i*-th inference rule, y^i is the output of the *i*-th input/output (I/O) local model, nr is the number of fuzzy rules, and $\mathrm{p} = [\mathrm{a_1 a_2} \ldots \mathrm{a_{na}}\ \mathrm{b_1 b_2} \ldots \mathrm{b_{nb}}\ \mathrm{c_1 c_2} \ldots \mathrm{c_{nc}}]$ are parameters of *i*-th I/O local model. So the output of this layer is:

$$\mathrm{O_j^4 = \tau_j = O_j^3 . f_j = O_j^3 . (p_j . q_{Ref} + q_j . \dot{q}_{Ref} + r_j)} \quad (6)$$

where f_j represents the T-S function, p_j, q_j, and r_j are the consequent parameters [38].

According to several studies, the synaptic weights between the hidden layers have no physical significance. In this work, through the network architecture we can take the synaptic weights between the second and the third hidden layer in the form of the consequent parameters of the T-S function.

TABLE I. RULE BASES FOR FNN-1

$\dot{q}_{Ref1}$ / q_{Ref1}	NB	NM	ZE	PM	PB
NB	ZE	PS	NB	NB	NB
NM	PS	NS	ZE	PS	NM
ZE	PM	NB	PB	PM	PB
PM	PM	ZE	NM	NB	PS
PB	NS	NS	PB	PB	ZE

TABLE II. RULE BASES FOR FNN-2

$\dot{q}_{Ref2}$ / q_{Ref2}	NB	NM	ZE	PM	PB
NB	NB	ZE	PB	NS	NS
NM	NS	PS	PM	NB	ZE
ZE	PM	PM	PS	PB	ZE
PM	NM	NB	PS	NS	NM
PB	PB	PM	PM	ZE	PS

E. Layer 5

For given values of the input variables $x_j(k)$, the final output of the fuzzy plant model $\hat{y}$ is inferred by taking the weighted average of the local model outputs y_i [39].

$$\hat{y}(k+1) = \frac{\sum_{i=1}^{nr} K_i . \mu^i(x(k)) . y^i(k)}{\sum_{i=1}^{nr} \mu^i(x(k))} \quad (7)$$

$$\mu^i\big(x(k)\big) = \prod_{j=1}^{nx} \mu_j^i \left(x_j(k) \right) \quad (8)$$

$\mu^{\mathrm{i}}(\mathrm{x(k)})$ denotes the membership functions of the fuzzy set F_j^i, and K_i denotes the gravity centers of the output fuzzy sets associated with each rule. It means that a local model in the consequent part of each fuzzy rule describes each operation of the arm manipulator depending on the input values. In this layer only one node is needed for a single motion command τ.

$$O_j^5 = \boldsymbol{\tau} = \frac{\sum_j K_i O_j^4}{\sum_j O_j^3} = \frac{\sum_j K_i O_j^3 f_j}{\sum_j O_j^3}$$

Because the number of rules in Layer 3 is pre-specified by the number of nodes, and weights for the input and output layer are fixed, the parameters to learn in the FNN are the modifiable weights present on the input links to the rule layer and the output membership layer and the consequent parameters of the T-S function.

IV. SUPERVISED TRAINING METHOD

The purpose of learning in an FNN controller is to adapt these synaptic weights, in order to generate a control vector τ corresponding to a sampled movement. During learning, an error rate related to the resultant motion is back-propagated to adjust the parameters from layer to layer sequentially [40, 41]. The error rate can be obtained by differentiating the error between the reference motion and the actual motion relative to the motion command. The training adopted for this error is the supervised training; this method is widely applied in minimization problems, optimal control, parameter identification, neural network training, etc. [42]. The adaptation law of the weights is:

$$\omega_{ij}^k(t+1) = \omega_{ij}^k(t) - \mu . \frac{\partial e(\omega)}{\partial \omega_{ij}^k(t)} + \alpha . \left(\omega_{ij}^k(t) - \omega_{ij}^k(t-1) \right) \quad 0 \le \alpha < 1 \quad (10)$$

where $\omega_{ij}^k(t)$ is the weight between the neuron j in layer k and the neuron i in layer (k-1), t is the index of iteration, α is the momentum, and μ the learning step.

The output of the neuron *j* in layer k is given by:

$$y_j^k(t) = f\left(S_j^k(t)\right) \quad (11)$$

$$S_j^k(t) = \sum_{i=1}^{nk} \omega_{ij}^k . y_i^{k-1}(t) \quad (12)$$

The connection weights are adjusted so that the feedback error $e(w)$ is minimized:

$$e(\omega) = \frac{1}{2} \sum_{j=1}^{n} (d_j - y_j)^2 \quad (13)$$

where f is the neuron activation function, $S_j^k(t)$ the input of neuron j, nk the number of corresponding neurons, n the number of training examples, d_j the reference output, and y_j is the actual output. We use the feedback error to derive the error rate $\partial e(\omega)/\partial \omega_{ij}^k(t)$.

$$\frac{\partial e(\omega)}{\partial \omega_{ij}^k(t)} = \frac{\partial e(\omega)}{\partial y_j^k(t)} . \frac{\partial y_j^k(t)}{\partial \omega_{ij}^k(t)} \quad (14)$$

$$\frac{\partial \mathrm{y_j^k(t)}}{\partial \omega_{\mathrm{ij}}^{\mathrm{k}}(\mathrm{t})} = \frac{\partial f\left(s_j^k(t)\right)}{\partial \omega_{\mathrm{ij}}^{\mathrm{k}}(\mathrm{t})} = f\left(S_j^k(t)\right) . \frac{\partial \mathrm{s_j^k(t)}}{\partial \omega_{\mathrm{ij}}^{\mathrm{k}}(\mathrm{t})} \quad (15)$$

and

$$\frac{\partial \mathrm{s_j^k(t)}}{\partial \mathrm{w_{ij}^k(t)}} = \frac{\partial \sum_{i=1}^{nk} \omega_{ij}^k . y_i^{k-1}(t)}{\partial \omega_{ij}^k(t)} = y_i^{k-1}(t) \quad (16)$$

So:

$$\frac{\partial y_j^k(t)}{\partial \omega_{ij}^k(t)} = f\left(S_j^k(t)\right) . \ y_i^{k-1}(t) \quad (17)$$

Substituting the results obtained in (17) into (14) the derivative of the error rate is obtained as:

$$\frac{\partial e(\omega)}{\partial \omega_{ij}^k(t)} = \frac{\partial e(\omega)}{\partial y_j^k(t)} \cdot f\left(S_j^k(t)\right) \cdot y_i^{k-1}(t) \quad (18)$$

Now by simply choosing:

$$S_j^k(t) = -\frac{\partial e(\omega)}{\partial y_j^k(t)} \cdot f\left(S_j^k(t)\right) \quad (19)$$

The second term at the right side of (18) satisfies:

$$\frac{\partial e(\omega)}{\partial \omega_{ij}^k(t)} = -S_j^k(t) \cdot y_i^{k-1}(t) \quad (20)$$

Finally, we get:

$$\omega_{ij}^k(t+1) = \omega_{ij}^k(t) + \mu\, S_j^k(t) \cdot y_i^{k-1}(t) + \alpha\left(\omega_{ij}^k(t) - \omega_{ij}^k(t-1)\right) \quad (21)$$

This final function represents the adaptation law of the weights used in the FNN controller.

V. SIMULATION

To demonstrate the effectiveness of the proposed approach, a two-joint robot manipulator was used [43]. The dynamic equations for this planar manipulator of 2DoF as shown in Figure 4 are described by the joint positions, velocities, and accelerations as functions of time [44]:

$$\begin{cases} \tau_1 = H_{11}.\ddot{q}_1 + H_{12}.\ddot{q}_2 - H.(\dot{q}_2^2 + 2.\dot{q}_1.\dot{q}_2) + G_1 \\ \tau_2 = H_{21}.\ddot{q}_1 + H_{22}.\ddot{q}_2 + H.\dot{q}_1^2 + G_2 \end{cases} \quad (22)$$

where q and $\dot{q}$ are the actual position and velocity vectors and τ_1 and τ_2 are the torques.

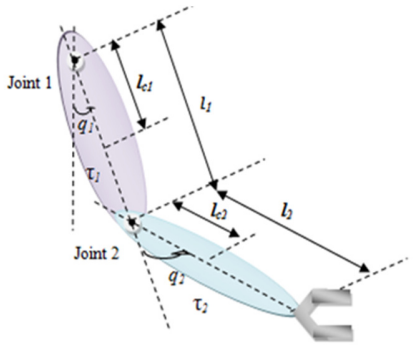

Fig. 4. The flexible robot manipulator of 2DoF.

In this simulation, the parameters for robot manipulator are given as:

$$\begin{aligned} H_{11} &= m_1.l_{c1}^2 + I_1 + m_2.(l_1^2 + l_{c2}^2 + 2.l_1.l_{C2}.cosq_2) + I_2 \\ H_{22} &= m_2.l_{c2}^2 + +I_2 \\ H_{12} &= \ m_2.l_1.l_{C2}.cosq_2 + m_2.l_{c2}^2 + \ I_2 \\ H_{21} &= H_{12} \\ H &= m_2.l_1.l_{C2}.sinq_2 \\ G_1 &= m_1.l_{c1}.g.cosq_1 + m_2.g.(l_{c2}^2. + l_1.cosq_1) \\ G_2 &= m_2.l_{c2}.g.\cos\,(q_1 + q_2) \end{aligned} \quad (23)$$

where q_1 , q_2 are the positions of links, m_1 , m_2 are the masses, l_1 , l_2 the lengths, I_1 , I_2 express lengthwise centroid inertia values, and l_{C1} , l_{C2} are the half lengths.

The resolution of the mathematical model with computed torques of the flexible robot manipulator of 2 DoF requires transforming them in the form of the following differential functions.

For the first joint:

$$\begin{cases} \frac{dq_1}{dt} = q_2 \\ \frac{dq_2}{dt} = q_3 \\ \frac{dq_3}{dt} = \frac{\tau_1 - H_{11}.q_3 + H.(q_3^2 + 2.q_2.q_3) - G_1}{H_{12}} \end{cases} \quad (24)$$

For the second joint:

$$\begin{cases} \frac{dq_1}{dt} = q_2 \\ \frac{dq_2}{dt} = q_3 \\ \frac{dq_3}{dt} = \frac{\tau_2 - H_{21}.q_3 + H.q_2^2 - G_2}{H_{22}} \end{cases} \quad (25)$$

These functions are derived by using two methods, Euler, and Runge-Kutta method [45, 46]. In many cases, differential equation systems can take the form of an ordinary first-order differential equation of the type:

$$\begin{cases} \frac{dy}{dt} = f(t, y(t)) \\ y(0) = y_0 \end{cases} \quad 0 \le t \quad (26)$$

where $y(t)$ is the function we are looking for and y_0 its initial value. We note h as a step based on the discretization of the variable t, and y_n the approximate value of $y(t_n)$ for the different instant $t_n = nh$. By integrating the differential equation between t_n and t_{n+1} we have the relation:

$$y(t_{n+1}) - y(t_n) = \int_{t_n}^{t_{n+h}} f\big(t, y(t)\big)dt \quad (27)$$

The idea is to approach this integral with more precision than the Euler method.

A. Euler Method

Coming back to Euler's method, the integral in (26) can be approached by the rectangle method on the left (explicit Euler), or by the rectangle method on the right (implicit Euler). The error produced corresponds to the area of quasi-triangular shape and of dimension $h.ph$ where p is the slope of f at the instant t_n. So the error is about: $e_E \simeq \frac{1}{2}Ph^2$.

Improvements to the Runge-Kutta method of order 2 consist in improving the integral by calculating the area of a trapezoid instead of that of a rectangle. The error is therefore related to the curvature of the function and not to its slope. In order to reduce the error of the later we used the Runge-Kutta method of order 4.

B. Runge-Kutta Method

The Runge-Kutta method of order 4 is an additional step in the refinement of the calculation of the integral (26). Instead of using the trapezoid method, we use the Simpson method [47]. This consists of replacing the integrated function with a parabola passing through the extreme points and the midpoint. We have:

$$\int_a^b f(x)dx \simeq \frac{b-a}{6}\left[f(a) + 4f\left(\frac{a+b}{2}\right) + f(b)\right] \quad (28)$$

Applied to the integral (26), this gives:

$$\int_{t_n}^{t_{n+1}} f(t, y(t))dt \simeq \frac{h}{6}\left[f(t_n, y(t_n)) + 4f\left(t_{n+1/2}, y(t_{n+1/2})\right) + f(t_{n+1}, y(t_{n+1}))\right]$$

Hence the relationship:

$$y_{n+1} = y_n + \frac{h}{6}[f(t_n, y_n) + 4f(t_{n+1/2}, y_{n+1/2}) + f(t_{n+1}, y_{n+1})] \quad (29)$$

Here, a difficulty appears because the equation presents two unknowns: $y_{n+1/2}$ and y_{n+1}. To make the diagram explicit, we must estimate $4f(t_{n+1/2}, y_{n+1/2})$ and $f(t_{n+1}, y_{n+1})$ from y_n, t_n and h. Let's start with the term $4f(t_{n+1/2}, y_{n+1/2})$ which we break down into two identical terms:

$$2f(t_{n+1/2}, y_{n+1/2}) + 2f(t_{n+1/2}, y_{n+1/2})$$

In the first, we replace $y_{n+1/2}$ by its value deduced from the explicit Euler method:

$$y^a_{n+1/2} = y_n + h/2.f(t_n, y_n).$$

In the second term, we replace $y_{n+1/2}$ by its value deduced from the implicit Euler method:

$$y^b_{n+1/2} = y_n + h/2.f(t_{n+1/2}, y_{n+1/2})$$

that we will approach $y_n + h/2.f(t_{n+1/2}, y^a_{n+1/2})$. Since the implicit and explicit Euler methods produce quasi-opposite errors, there is a hope to minimize the error on the calculation of $4f(t_{n+1/2}, y_{n+1/2})$. To summarize, we will write:

$$4f(t_{n+1/2}, y_{n+1/2}) \simeq 2k_2 + 2k_3$$

$$with \begin{cases} k_1 = f(t_n, y_n) \\ k_2 = f\left(t_n + \frac{1}{2}h, y_n + \frac{1}{2}hk_1\right) \\ k_3 = f\left(t_n + \frac{1}{2}h, y_n + \frac{1}{2}hk_2\right) \end{cases}$$

As for the term $f(t_{n+1}, y_{n+1})$, we approach it by estimating y_{n+1} by the midpoint method, i.e. by applying the rectangle method in the middle:

$$y_{n+1} \simeq y_n + hf(t_{n+1/2}, y_{n+1/2}) \simeq y_n + hf(t_{n+1/2}, y^b_{n+1/2})$$

Finally we obtain the explicit diagram of Runge-Kutta of order 4:

$$y_{n+1} = y_n + h\left[\frac{1}{6}k_1 + \frac{1}{3}k_3 + \frac{1}{3}k_3 + \frac{1}{6}k_4\right] \quad (30)$$

$$with \begin{cases} k_1 = f(t_n, y_n) \\ k_2 = f\left(t_n + \frac{1}{2}h, y_n + \frac{1}{2}hk_1\right) \\ k_3 = f\left(t_n + \frac{1}{2}h, y_n + \frac{1}{2}hk_2\right) \\ k_4 = f(t_n + h, y_n + hk_3) \end{cases}$$

Compared to the RK2 method, this numerical diagram requires the double calculations at each step and therefore longer calculation time, without mentioning the rounding errors which accumulate faster. However, this defect is compensated by a gain in precision.

Finally, the algorithm developed for this method is:

Algorithm RK4
1. Initialization of step h, of duration T.
2. Initialization of the initial conditions
3. Definition of the function $f(t, y)$
4. While $t \leq T$ Do
(a) Calculate from $k_1 = f(t, y)$.
(b) Calculate from $k_2 = f\left(t + \frac{1}{2}h, y + \frac{1}{2}hk_1\right)$.
(c) Calculate from $k_3 = f\left(t + \frac{1}{2}h, y + \frac{1}{2}hk_2\right)$.
(d) Calculate from $k_4 = f(t + h, y + hk_3)$.
(e) $y = y + \frac{h}{6}[k_1 + 2k_3 + 2k_3 + k_4]; t = t + h.$
(f) Save data.

Finally a comparative study will be carried out between Euler's method and Runge-Kutta method in order to prove the effectiveness of the later.

In each FNN controller, we have 2 nodes in layer 1, 10 nodes in layer 2, 25 nodes in layers 3 and 4, and 1 node in layer 5. Consequently, a total of 125 parameters must be adapted for each joint during training. Because between the second and the third layer we have 5.10=50 synaptic weights, and between the third and the fourth layer we have 25 synaptic weights to adapt, finally each neuron in layer 4 needs three parameters p_i, q_i and r_i where $O_i^4 = \tau_i = O_i^3(p_i q + q_i \dot{q} + r_i)$. The parameter r_i represents an amplifier of the torque τ_i, to simplify the calculation we take $r_i = 0$. Then total 50+25+50=125 parameters have to be set for each network. Moreover, it is by no means an easy task to identify the model parameters accurately [48], but as we mentioned earlier, to adapt and reduce the number of undefined parameters in the network, a new technique is used. Generally, for the adaptation of T-S function coefficients we use the Genetic Algorithm which is an optimization algorithm [32, 49]. In optimization, which is a field of mathematics, the goal is to come up with the member of a set that is the best according to some criterion [50].This method includes several steps and it takes a lot of time and calculations. In order to overcome this problem a new technique is used, with a simple idea: According to several studies, synaptic weights in the FNN controller have no physical significance, while in our work, the coefficients p_i and q_i represent the weights between the second and third hidden layer. This technique allows us to reduce the number from 125 to 75 parameters. The trajectory references tracked by joint-1 and joint-2 are expressed as follows [51]:

$$\begin{cases} q_{Ref1}(t) = \frac{1}{2}.\sin(2.\pi.t) \\ q_{Ref2}(t) = -\frac{\pi}{2} + \frac{1}{2}.\sin(2.\pi.t) \end{cases} \quad t \leq 3 \quad (31)$$

The simulation results are represented in Figures 5-11. In general, several hundred learning tests have been used to learn how to govern a sampled movement.

TABLE III. SIMULATED SYSTEM PARAMETERS

Variable	Value
L_1	0.32m
L_2	0.24m
L_{c1}	0.16m
L_{c2}	0.12m
m_1	1.8Kg
m_2	1.2Kg
I_1, I_2	0.02kg.m^2
g	9.8N/Kg

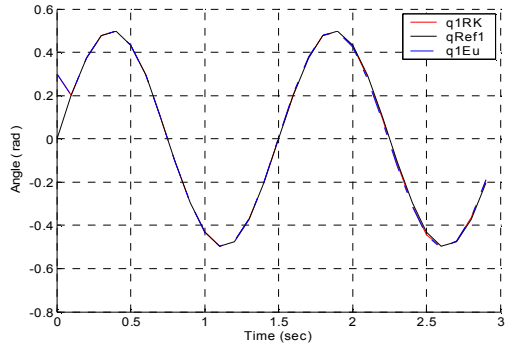

Fig. 5. Tracking position of joint-1 for the initial conditions $[q_1, \dot{q}_1, \ddot{q}_1, \tau]^T$=[0.3,0,0,0]:

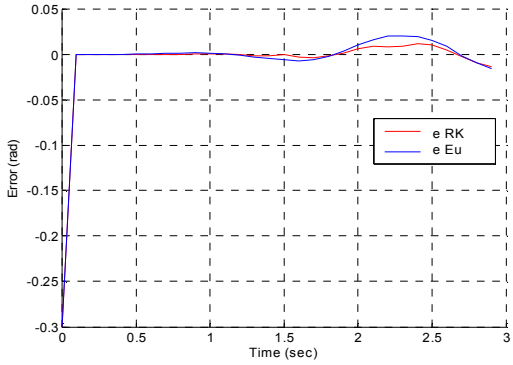

Fig. 6. Tracking error of joint angle 1.

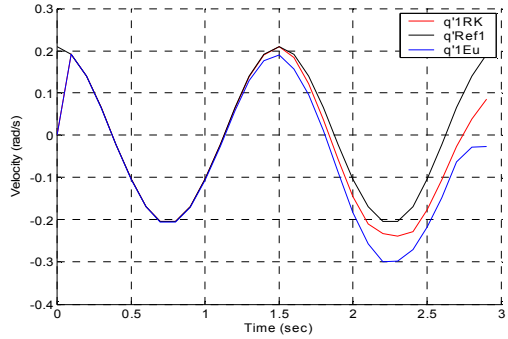

Fig. 7. Tracking angular velocity of joint-1.

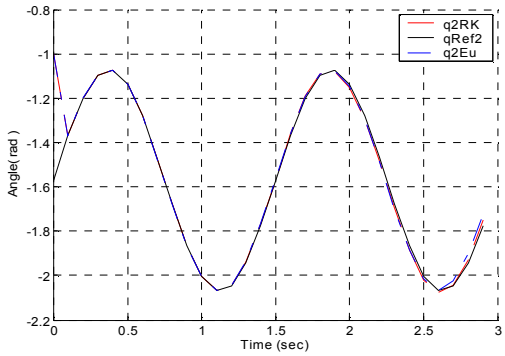

Fig. 8. Tracking position of joint-2 for the initial conditions $[q_2, \dot{q}_2, \ddot{q}_2, \tau]^T$=[-1,0,0,0].

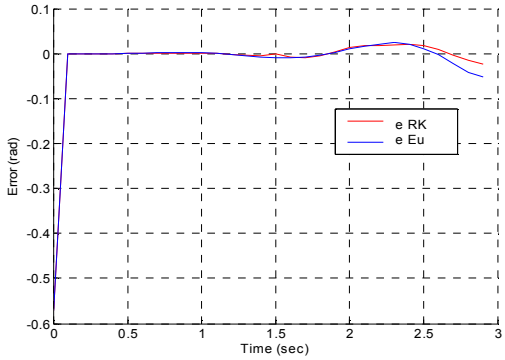

Fig. 9. Tracking error of joint angle 2.

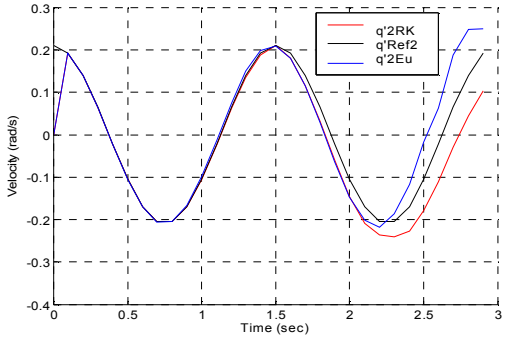

Fig. 10. Tracking angular velocity of joint-2.

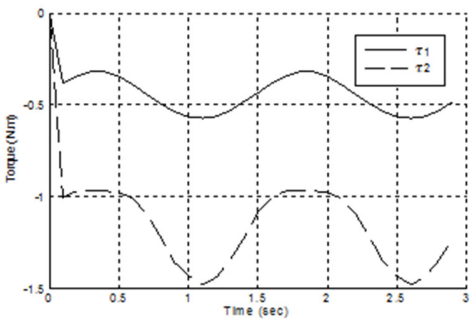

Fig. 11. FNNs outputs.

In order to solve the dynamic function, a comparative study was carried out between the two methods, the Euler method and the RK4 method. The simulation time is $t \in [0s,2.9s]$ and the sampling time is 0.01s. According to the simulation results, we can approve the efficiency of the RK4 method as can be seen from Figures 5, 7, 8, and 10 where the outputs of the first and second joint can track the reference movements and angular velocity respectively. The output is the position of the manipulator's end effector which can freely move. Due to this flexibility at the joint, oscillations are produced near the ends [8].

VI. TEST OF ROBUSTNESS

Robust control is an approach to controller design that aims to achieve robust performance and/or stability in the presence of uncertainties in the system [50]. In order to test the robustness of the proposed approach, changes of the weight and the lengths for each joint of the arm manipulator are carried out. Also changes in the initial values were affected.

TABLE IV. SIMULATED SYSTEM PARAMETERS

Variable	Value
L_1	0.40m
L_2	0.30m
L_{c1}	0.20m
L_{c2}	0.15m
m_1	2.0Kg
m_2	1.5Kg
I_1, I_2	0.028kg.m^2
g	9.8N/Kg

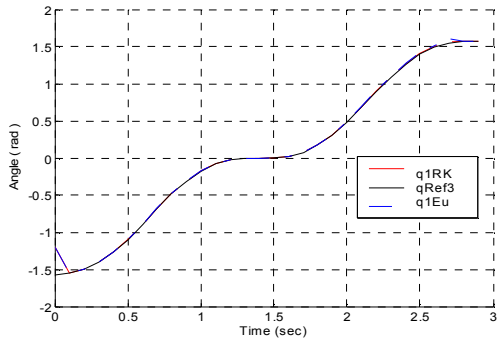

Fig. 12. Tracking position of joint-1 for the initial conditions $[q_1,q_2,q_3, \tau]^T=[-1.2,0,0,0]$.

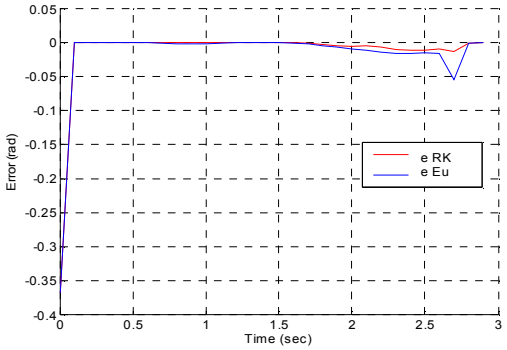

Fig. 13. Tracking error of joint angle 1.

We have chosen another motion execution described as follows [52]:

$$\begin{cases} q_{Ref3}(t) = -\frac{\pi}{2} + \frac{1}{4}.\left[2.\pi.\frac{t}{3} - \sin\left(2.\pi.\frac{t}{3}\right)\right] \\ q_{Ref4}(t) = \frac{1}{4}.\left[2.\pi.\frac{t}{3} - \sin\left(2.\pi.\frac{t}{3}\right)\right] \end{cases} t \leq 3 \quad (32)$$

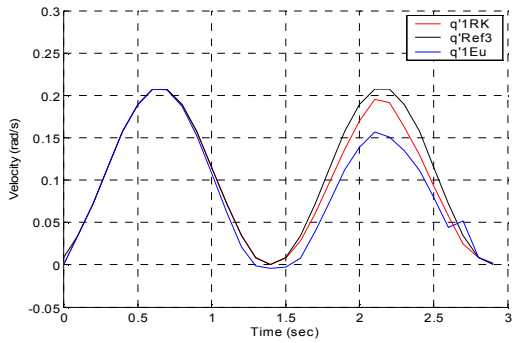

Fig. 14. Tracking angular velocity of joint-1.

TABLE V. VALUES OF TRACKING ERROR OF JOINT ANGLE 1

Time (s)	Error with RK4 method (e_{RK}) (rad)	Error with Euler method (e_{EU}) (rad)
0.1	-0.3678	-0.3678
0.2	0	0
0.3	0	-0.0001
0.4	-0.0001	-0.0002
0.5	-0.0001	-0.0002
0.6	-0.0001	-0.0003
0.7	0	-0.0005
0.8	0	-0.0012
0.9	-0.0001	-0.0017
1	-0.0002	-0.002
1.1	-0.0003	-0.0019
1.2	-0.0002	-0.0014
1.3	-0.0001	-0.0007
1.4	0	-0.0001
1.5	0	0
1.6	-0.0001	-0.0001
1.7	-0.0005	-0.0009
1.8	-0.0016	-0.0025
1.9	-0.0031	-0.0047
2	-0.0047	-0.0072
2.1	-0.0058	-0.0099
2.2	-0.0052	-0.0119
2.3	-0.0067	-0.0147
2.4	-0.0104	-0.0162
2.5	-0.0116	-0.0163
2.6	-0.0113	-0.0159
2.7	-0.0098	-0.0169
2.8	-0.0136	-0.055
2.9	-0.0012	-0.0009
3	0	-0.0001

The difference between the reference and current trajectory is an input vector to the controller that generates joint rate commands. The simulation results of the proposed approach show the responses of joint positions, and position tracking errors, tracking angular velocities, and the outputs of the FNNs, for joint 1 and 2, with different initial conditions. It can be seen from Figures 12, 14, and 17 that the outputs of the first and

second joint can track the reference movements. From Figures 6, 9, 13, and 16 can be obtained that with the adaptation of the fuzzy neural network synaptic weights, the system error is gradually reduced. According to the results shown in Table V, the average tracking error with the RK4 method is $3.10.10^{-3}$rad and with Euler's method it is $-6.33.10^{-3}$ rad. When the dynamic equations are derived by using the RK4 method, the system control effect is better, and tracking error of joint is smaller.

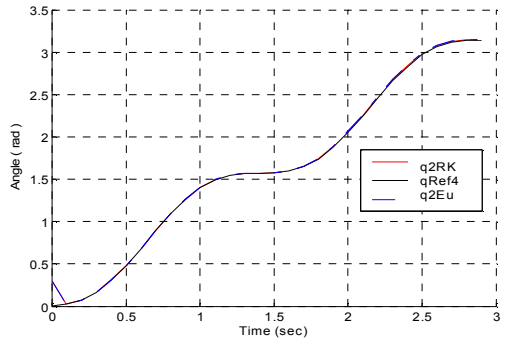

Fig. 15. Tracking position of joint-2 for the initial conditions $[q_1,q_2,q_3, \tau]^T=[0.3,0,0,0]$.

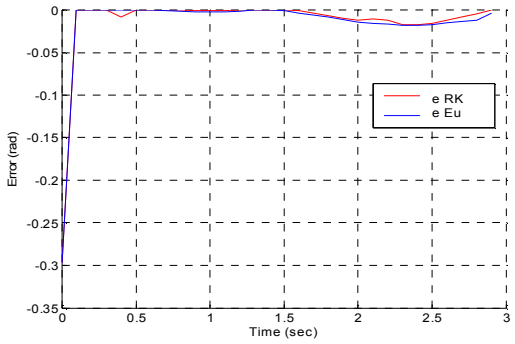

Fig. 16. Tracking error of joint angle 2.

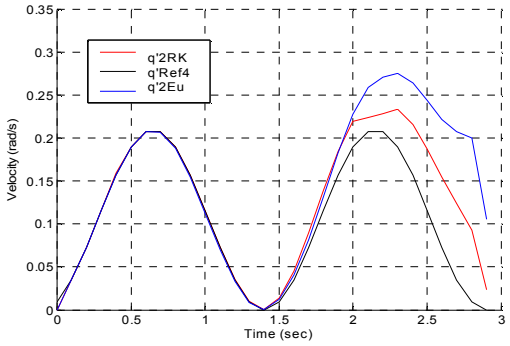

Fig. 17. Tracking angular velocity of joint-2.

Note that the proposed scheme bends the motion tracks near the ends, as shown in Figures 7, 10, 14, and 17. This error in speed produces an error in position. This phenomenon occurs because the scheme did not fully accomplish a salient braking in the end of the motion tracking and could be diminished via further learning in motion governing. Analogously, the human also demonstrates a similar behavior in performing fast movements: an overshoot in the end of the movement [53, 54]. As can be seen, the motions generated using the proposed approach capture the behaviors of the reference motions well and the simulation results verify the efficiency of the proposed solution.

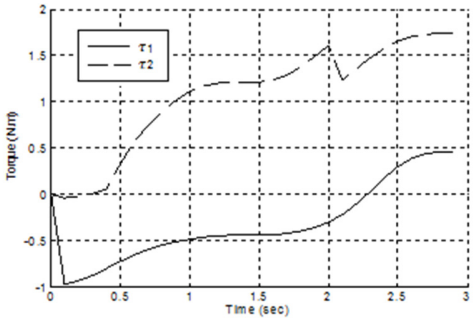

Fig. 18. FNNs outputs.

VII. CONCLUSION

This paper presents a solution for the problem of learning and controlling a 2DoF industrial manipulator. One of the major problems in applying learning controllers to govern general motions is that the dynamics of robot manipulators are, in general, nonlinear, time-varying, and complex, which makes implementation in real time difficult. To tackle this, we have developed a Fuzzy Neural Network controller by taking advantage of the merits of the T-S fuzzy system and a Neural Network using supervised training. Network parameters and structure identifications were performed by using the ANFIS system. This design allows new parameters of the controller to be adapted in order to decrease the position tracking errors. In order to solve the dynamic function, a comparative study was carried out between two methods, the Euler method and the RK4 method, and according to the simulation results, we can accept the efficiency of the later. Finally, in order to test the system robustness, the proposed approach was also investigated for parameter variations and for another motion execution. The simulation results show that the proposed approach has a good tracking effect, and verify that the proposed scheme is sufficiently accurate and robust. Regarding future research, we are going to include a task of robust control strategies on multi DoF robotic manipulators and evaluate their performance.

REFERENCES

[1] J. Iqbal, M. Ul Islam, S. Abbas, A. A. Khan, and S. Ajwad, "Automating industrial tasks through mechatronic systems – A review of robotics in industrial perspective," *Tehnicki Vjesnik*, vol. 23, no. 3, pp. 917–924, Jun. 2016, doi: 10.17559/TV-20140724220401.

[2] J. Wawerla and R. T. Vaughan, "A fast and frugal method for team-task allocation in a multi-robot transportation system," in *IEEE International Conference on Robotics and Automation*, Anchorage, AK, USA, May 2010, pp. 1432–1437, doi: 10.1109/ROBOT.2010.5509865.

[3] Mitsuru Endo *et al.*, "A car transportation system by multiple mobile robots - iCART -," in *IEEE/RSJ International Conference on Intelligent*

Robots and Systems, Nice, France, Sep. 2008, pp. 2795–2801, doi: 10.1109/IROS.2008.4651200.

[4] M. Fei, Z. Haiou, and W. Guilan, "Application of industrial robot in rapid prototype manufacturing technology," in *2nd International Conference on Industrial Mechatronics and Automation*, Wuhan, China, May 2010, vol. 1, pp. 218–220, doi: 10.1109/ICINDMA.2010.5538187.

[5] L. Yanping and L. Haijiang, "Welding multi-robot task allocation for BIW based on hill climbing genetic algorithm," in *International Technology and Innovation Conference*, Xi'an, China, Oct. 2009, pp. 1–8, doi: 10.1049/cp.2009.1485.

[6] H. B. Chen, T. Lin, S. B. Chen, J. F. Wang, J. Q. Jia, and H. Zhang, "Adaptive control on wire feeding in robot arc welding system," in *IEEE Conference on Robotics, Automation and Mechatronics*, Chengdu, China, Sep. 2008, pp. 119–122, doi: 10.1109/RAMECH.2008.4690868.

[7] S. A. Ajwad, J. Iqbal, M. I. Ullah, and R. U. Islam, *Modeling Robotic Arms – A Review and Derivation of Screw Theory Based Kinematics*. Islamabad, Pakistan: Institute of Imformation Technology, 2014.

[8] W. Alam, S. Ahmad, A. Mehmood, and J. Iqbal, "Robust Sliding Mode Control for Flexible Joint Robotic Manipulator via Disturbance Observer," *Interdisciplinary Description of Complex Systems*, vol. 17, no. 1-B, pp. 85–97, 2019.

[9] S. A. Ajwad, M. I. Ullah, K. Baizid, and J. Iqbal, "A comprehensive state-of-the-art on control of industrial articulated robots," *Journal of the Balkan Tribological Association*, vol. 20, no. 4, pp. 499–521, Dec. 2014.

[10] S. Ajwad, J. Iqbal, A. A. Khan, and A. Mehmood, "Disturbance-Observer-Based Robust Control of a Serial-link Robotic Manipulator Using SMC and PBC Techniques," *Studies in Informatics and Control*, vol. 24, no. 4, pp. 401–408, Dec. 2015, doi: 10.24846/v24i4y201504.

[11] S. Ajwad, J. Iqbal, M. U. Islam, A. Alsheikhy, A. Almeshal, and A. Mehmood, "Optimal and Robust Control of Multi DOF Robotic Manipulator: Design and Hardware Realization," *Cybernetics and Systems*, vol. 49, no. 1, pp. 77–93, Feb. 2018, doi: 10.1080/01969722.2017.1412905.

[12] Z. S. Awan, K. Ali, J. Iqbal, and A. Mehmood, "Adaptive Backstepping Based Sensor and Actuator Fault Tolerant Control of a Manipulator," *Journal of Electrical Engineering & Technology*, vol. 14, no. 6, pp. 2497–2504, Nov. 2019, doi: 10.1007/s42835-019-00277-9.

[13] J. Iqbal, "Modern Control Laws for an Articulated Robotic Arm: Modeling and Simulation," *Engineering, Technology & Applied Science Research*, vol. 9, no. 2, pp. 4057–4061, Apr. 2019.

[14] W. Alam, A. Mehmood, K. Ali, U. Javaid, S. Alharbi, and J. Iqbal, "Nonlinear Control of a Flexible Joint Robotic Manipulator with Experimental Validation," *Journal of Mechanical Engineering*, vol. 64, no. 1, pp. 47–55, Jan. 2018, doi: 10.5545/sv-jme.2017.4786.

[15] J. Iqbal, M. Ullah, A. A. Khan, and M. Irfan, "Towards Sophisticated Control of Robotic Manipulators: An Experimental Study on a Pseudo-Industrial Arm," *Journal of Mechanical Engineering*, vol. 61, no. 7, pp. 465–470, Jul. 2015, doi: 10.5545/sv-jme.2015.2511.

[16] A. Ailon, R. Lozano, and M. I. Gil', "Iterative regulation of an electrically driven flexible-joint robot with model uncertainty," *IEEE Transactions on Robotics and Automation*, vol. 16, no. 6, pp. 863–870, Dec. 2000, doi: 10.1109/70.897798.

[17] A. Benzaouia and A. El Hajjaji, *Advanced Takagi–Sugeno Fuzzy Systems*, vol. 8. New York, NY, USA: Springer, 2014.

[18] O. F. Lutfy, "Wavelet Neural Network Model Reference Adaptive Control Trained by a Modified Artificial Immune Algorithm to Control Nonlinear Systems," *Arabian Journal for Science and Engineering*, vol. 39, no. 6, pp. 4737–4751, Jun. 2014, doi: 10.1007/s13369-014-1088-5.

[19] G. Dreyfus *et al.*, *Reseaux de neurones, methodologie et application*. Paris, France: Eyrolles, 2002.

[20] M. Alizadeh, F. Jolai, M. Aminnayeri, and R. Rada, "Comparison of different input selection algorithms in neuro-fuzzy modeling," *Expert Systems with Applications*, vol. 39, no. 1, pp. 1536–1544, Jan. 2012, doi: 10.1016/j.eswa.2011.08.049.

[21] A. Subasi, "Application of adaptive neuro-fuzzy inference system for epileptic seizure detection using wavelet feature extraction," *Computers in biology and medicine*, vol. 37, no. 2, pp. 227–244, Mar. 2007, doi: 10.1016/j.compbiomed.2005.12.003.

[22] M. R. U. Islam, J. Iqbal, and Q. Khan, "Design and Comparison of Two Control Strategies for Multi-DOF Articulated Robotic Arm Manipulator," *Control Engineering and Applied Informatics*, vol. 16, no. 2, pp. 28–39, Jun. 2014.

[23] S. A. Ajwad, J. Iqbal, M. I. Ullah, and A. Mehmood, "A systematic review of current and emergent manipulator control approaches," *Frontiers of Mechanical Engineering*, vol. 10, no. 2, pp. 198–210, Jun. 2015, doi: 10.1007/s11465-015-0335-0.

[24] W. T. Miller, F. H. Glanz, and L. G. Kraft, "Application of a General Learning Algorithm to the Control of Robotic Manipulators," *The International Journal of Robotics Research*, vol. 6, no. 2, pp. 84–98, Jun. 1987, doi: 10.1177/027836498700600207.

[25] H. Taheri Shahraiyni, S. Sodoudi, A. Kerschbaumer, and U. Cubasch, "A new structure identification scheme for ANFIS and its application for the simulation of virtual air pollution monitoring stations in urban areas," *Engineering Applications of Artificial Intelligence*, vol. 41, pp. 175–182, May 2015, doi: 10.1016/j.engappai.2015.02.010.

[26] M. Panella, A. Rizzi, F. M. F. Mascioli, and G. Martinelli, "ANFIS synthesis by hyperplane clustering," in *9th IFSA World Congress and 20th NAFIPS International Conference*, Vancouver, BC, Canada, Jul. 2001, vol. 1, pp. 340–345, doi: 10.1109/NAFIPS.2001.944275.

[27] M. Dong and N. Wang, "Adaptive network-based fuzzy inference system with leave-one-out cross-validation approach for prediction of surface roughness," *Applied Mathematical Modelling*, vol. 35, no. 3, pp. 1024–1035, Mar. 2011, doi: 10.1016/j.apm.2010.07.048.

[28] J. F. Joduin, *Les reseaux de neurones. Principe et definitions*. Paris, France: Hermes, 1994.

[29] D. Dubois and H. Prade, *Fuzzy Sets and Systems: Theory and Applications*. Cambridge, USA: Academic Press, 1980.

[30] M. M. A. Elsalam, S. F. Sarayah, and F. M. Banoor, "Fuzzy Control of Multi-Link Robotic Arm with Multi-Weight Objects," *Current Science International*, vol. 6, no. 1, pp. 218–228, 2017.

[31] P. Deka and V. Chandramouli, "A fuzzy neural network model for deriving the river stage—discharge relationship," *Hydrological Sciences Journal*, vol. 48, no. 2, pp. 197–209, Apr. 2003, doi: 10.1623/hysj.48.2.197.44697.

[32] A. Tang, C. Quek, and G. Ng, "GA-TSKfnn: Parameters tuning of fuzzy neural network using genetic algorithms," *Expert Systems with Applications*, vol. 29, no. 4, pp. 769–781, Nov. 2005, doi: 10.1016/j.eswa.2005.06.001.

[33] W.-H. Ho, J.-T. Tsai, B.-T. Lin, and J.-H. Chou, "Adaptive network-based fuzzy inference system for prediction of surface roughness in end milling process using hybrid Taguchi-genetic learning algorithm," *Expert Systems with Applications*, vol. 36, no. 2, pp. 3216–3222, Mar. 2009, doi: 10.1016/j.eswa.2008.01.051.

[34] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-15, no. 1, pp. 116–132, Feb. 1985, doi: 10.1109/TSMC.1985.6313399.

[35] H. R. Berenji and P. Khedkar, "Learning and tuning fuzzy logic controllers through reinforcements," *IEEE Transactions on Neural Networks*, vol. 3, no. 5, pp. 724–740, Sep. 1992, doi: 10.1109/72.159061.

[36] H. K. Lam and S. C. Tan, "Stability analysis of fuzzy-model-based control systems: application on regulation of switching DC-DC converter," *IET Control Theory & Applications*, vol. 3, no. 8, pp. 1093–1106, Aug. 2009, doi: 10.1049/iet-cta.2008.0168.

[37] H. K. Lam and L. D. Seneviratne, "Tracking control of sampled-data fuzzy-model-based control systems," *IET Control Theory & Applications*, vol. 3, no. 1, pp. 56–67, Jan. 2009, doi: 10.1049/iet-cta:20070466.

[38] L. Xiong, A. Y. Shamseldin, and K. M. O'Connor, "A non-linear combination of the forecasts of rainfall-runoff models by the first-order Takagi–Sugeno fuzzy system," *Journal of Hydrology*, vol. 245, no. 1, pp. 196–217, May 2001, doi: 10.1016/S0022-1694(01)00349-3.

[39] C. C. Lee, "Fuzzy logic in control systems: fuzzy logic controller. I," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 20, no. 2, pp. 404–418, Apr. 1990, doi: 10.1109/21.52551.

[40] M. Samhouri, A. Al-Ghandoor, S. A. Ali, I. Hinti, and W. Massad, "An Intelligent Machine Condition Monitoring System Using Time-Based Analysis: Neuro-Fuzzy Versus Neural Network," *Jordan Journal of Mechanical and Industrial Engineering*, vol. 3, no. 4, pp. 294–305, Dec. 2009.

[41] B. S. Reddy, J. S. Kumar, and K. V. K. Reddy, "Prediction of Surface Roughness in Turning Using Adaptive Neuro-Fuzzy Inference System," *Jordan Journal of Mechanical and Industrial Engineering*, vol. 3, no. 4, pp. 252–259, Dec. 2009.

[42] Y. Shi and M. Mizumoto, "Some considerations on conventional neuro-fuzzy learning algorithms by gradient descent method," *Fuzzy Sets and Systems*, vol. 112, no. 1, pp. 51–63, May 2000, doi: 10.1016/S0165-0114(98)00056-6.

[43] W. Afzal, S. Iqbal, Z. Tahira, and M. E. Qureshi, "Gesture Control Robotic Arm Using Flex Sensor," *Applied and Computational Mathematics*, vol. 6, no. 4, pp. 171–176, Jan. 2017, doi: 10.11648/j.acm.20170604.12.

[44] C. Chavez-Olivares, F. Reyes Cortes, and E. Gonzalez-Galvan, "On Explicit Force Regulation with Active Velocity Damping for Robot Manipulators," *Automatika*, vol. 56, no. 4, pp. 478–490, Jan. 2015, doi: 10.7305/automatika.2016.01.399.

[45] J. Butcher, *Runge–Kutta methods for ordinary differential equations*. Wellington, New Zealand: The University of Auckland, 2005.

[46] B. Stout, *Methodes numeriques de resolution d'equations differentielles*. Marseille, France: Universite de Provence, 2007.

[47] E. G. Da Silva, *Methodes et Analyse Numeriques*. Grenoble, France: Institut Polytechnique de Grenoble, 2007.

[48] S. Ullah, A. Mehmood, Q. Khan, S. Rehman, and J. Iqbal, "Robust Integral Sliding Mode Control Design for Stability Enhancement of Under-actuated Quadcopter," *International Journal of Control, Automation and Systems*, vol. 18, no. 7, pp. 1671–1678, Jul. 2020, doi: 10.1007/s12555-019-0302-3.

[49] Z. Civelek, "Optimization of fuzzy logic (Takagi-Sugeno) blade pitch angle controller in wind turbines by genetic algorithm," *Engineering Science and Technology, an International Journal*, vol. 23, no. 1, pp. 1–9, Feb. 2020, doi: 10.1016/j.jestch.2019.04.010.

[50] S. A. Ajwad, A. Mehmood, M. Ullah, and J. Iqbal, "Optimal V/S robust control: A study and comparision for articulated manipulator," *Journal of the Balkan Tribological Association*, vol. 22, no. 3, pp. 2460–2466, 2016.

[51] Y. Yanling, "Model Free Adaptive Control for Robotic Manipulator Trajectory Tracking," *The Open Automation and Control Systems Journal*, vol. 7, no. 1, pp. 358–365, Apr. 2015, doi: 10.2174/1874444301507010358.

[52] H. Seraji, "Decentralized adaptive control of manipulators: theory, simulation, and experimentation," *IEEE Transactions on Robotics and Automation*, vol. 5, no. 2, pp. 183–201, Apr. 1989, doi: 10.1109/70.88039.

[53] C. H. Wu, K. Y. Young, K. S. Hwang, and S. Lehman, "Voluntary movements for robotic control," *IEEE Control Systems Magazine*, vol. 12, no. 1, pp. 8–14, Feb. 1992, doi: 10.1109/37.120444.

[54] M. B. Ayed, L. Zouari, and M. Abid, "Software In the Loop Simulation for Robot Manipulators," *Engineering, Technology & Applied Science Research*, vol. 7, no. 5, pp. 2017–2021, Oct. 2017.