Iranian Journal of Fuzzy Systems

Volume 20, Number 5, (2023), pp. 89-107

Original paper

University of Sistan and Baluchestan

Robust controller design for a class of MIMO nonlinear systems using TOPSIS function-link fuzzy cerebellar model articulation controller and interval type-2 fuzzy compensator

S. Sepahvand[1], N. Amiri[2], M. Pourgholi[3] and V. Fakhari[4]

[1,3] *Faculty of Electrical Engineering, Shahid Beheshti University, Tehran, Iran*
[2,4] *Faculty of Mechanical and Energy Engineering, Shahid Beheshti University, Tehran, Iran*

shayan.sp@torontomu.ca, ni.amiri@torontomu.ca, m_pourgholi@sbu.ac.ir, v_fakhari@sbu.ac.ir

Abstract

This research suggests a novel controller capable of achieving satisfactory tracking control performance for a class of multiple-input multiple-output (MIMO) systems. It aims to approximate an impracticable ideal controller to design directly due to insufficient knowledge about the plant or lack of data. The primary control unit is a function-link fuzzy cerebellar model articulation controller (FLFCMAC), applying the technique for order of preference by similarity to the ideal solution (TOPSIS) to eliminate less dominant rules in the fuzzy inference engine. A function-link network (FLN) is added to this controller to enhance the tracking performance. Moreover, an interval type-2 fuzzy logic controller (IT2FLC) is proposed to improve the closed-loop system performance. Due to the caterpillar robot's highly nonlinear dynamics and undesirable factors such as external disturbance and parametric uncertainties, online tuning laws are offered. In other words, the control system is designed to be robust. A Lyapunov stability approach is presented to show the stability of the controlled nonlinear plant. Furthermore, the adaptation laws are derived through this approach. Finally, numerical simulations are employed to validate the effectiveness and robustness of the controller for the caterpillar robot and the inverted double pendulum.

Keywords: Caterpillar robot, fuzzy cerebellar model articulation controller, interval type-2 fuzzy logic controller, function-link network, fuzzy inference engine, online learning control system.

1 Introduction

Many bio-inspired robots are nonlinear and sensitive to various uncertainties and disturbances. These undesirable factors contribute to the unsatisfactory performance of many control systems and even may lead to instability. For example, assume that the goal is the precise positioning of a caterpillar robot where a very slight displacement of its bars is required in each step. In addition to precision, this robot is expected to perform for a prolonged period, and its parameters consequently vary on account of overheating; therefore, adopting a robust controller seems inevitable.

Turning to possible solutions, for example, neural networks and fuzzy systems may be combined with sliding mode control (SMC) [1, 16]. This complex controller not only develops robustness but also increases convergence speed, mainly when applied to chaotic systems. In other words, it benefits from robustness against parameter variations, external disturbance rejection ability, and rapid stabilization of highly nonlinear dynamics. Likewise, this manuscript investigates a method that benefits from a complicated neural network whose central unit is a CMAC.

Lin defines a CMAC as a network with a partially connected perceptron-like associative memory network [12], whose structure contains blocks and layers. These blocks are activated whenever a quantized input stimulates them. Many researchers state that CMAC benefits from specific qualities such as fast learning and generalization (the ability to

Corresponding Author: V. Fakhari
Received: September 2022; Revised: May 2023; Accepted: June 2023.
https://doi.org/10.22111/IJFS.2023.43250.7618

generate similar outputs for similar inputs) [2, 26]. To increase the flexibility of this network, it is often combined with other networks to perform a better estimation of nonlinear functions in control or identification tasks [5, 20]. Not only have many scholars combined it with other networks, but they also conveyed studies concerning altering its learning methods, e.g., [26]. Recently, several self-organizing structures of CMAC have been introduced in [9, 12, 13] to avoid the significant issue of out-of-range inputs. In other words, if the initial ranges of blocks are not chosen precisely, no block will be activated. This issue will lead to the instability of closed-loop systems in control applications and a weak estimation in identification problems.

As was discussed, a CMAC is usually used as the main part of a control system, so it is either combined or modified. One of these modifications is the fuzzy CMAC (FCMAC) network that was first introduced in [20] and contains four layers. The first layer is called the input space layer. In control applications, it is often a tracking error vector like [4, 15] or a sliding manifold and its derivatives aggregated in a vector [7, 17]. The next layer is the association memory space that distinguishes FCMAC from CMAC. Instead of binary blocks, fuzzy membership functions (MFs) are utilized in this layer. In the related literature, Gaussian or wavelet MFs are applied. Other types of MFs can also build this layer; however, these two specific types are established because it is more straightforward to compute their derivatives. It is worth noting that employing overlapping [13] or recurrent MFs [10] is also possible. This contributes to the enhanced robustness of closed-loop nonlinear dynamics against various types of uncertainties and external disturbances. Moreover, fuzzy type-2 MFs are suggested in [14] to mitigate the undesirable effects of unwanted sensor noise; nonetheless, choosing fuzzy type-2 MFs affects the structure of the subsequent layers. The third layer, the receptive-field (or hypercube space), removes the input dimension from the calculations by multiplying activated MFs (blocks) in layers for every single input. Lin and Huynh [13] conveyed a study concerning the modification of inputs to this layer. More specifically, they sought some solutions to omit the MFs with firing strength lower than a threshold. Their investigations motivated others to extend their idea [13, 16]. To explain more, the number of exploited blocks was reduced, leading to the reduction of fuzzy rules; in fact, these methods facilitate decreasing the computational burden and avoiding microprocessor crashes. Then the weight memory space comprises an updating weight vector adapting to stabilize the tracking error dynamics. It can be combined with the FLN to enhance the estimation task. Eventually, the output layer performs an algebraic sum of activated functions in the receptive-field layer.

There is a rich literature on how CMAC can address nonlinear control problems. In [11], an intelligent controller is designed using interval type-2 fuzzy systems and cerebellar model articulation networks (CMAN). Zhao et al. studied the design of an optical tweezers manipulation control system based on a self-organizing FCMAC [28]. Another exciting application of CMAC is proposed in [19], where the authors used CMAC for high-resolution soil moisture mapping using sentinel-2 imagery coupled with geophysical properties. Tsai et al. [27] developed a control system based on CMAC to estimate the speed of a sensorless induction motor. In 2012, a Takagi–Sugeno–Kang (TSK) FCMAC robust adaptive controller was presented in [15]. This controller was combined with a robust controller to dispel the remaining tracking error. It outperformed the conventional CMAC due to variable output weights. Sepahvand and Pourgholi [25] designed a supervisory CMAC controller to synchronize two chaotic systems. The supervisory controller is designed to avoid undesirable control system performance as the tracking error exceeds a certain value. Lin and Chen [12] suggested a self-organizing CMAC (SOCMAC) for a class of MIMO nonlinear uncertain systems where gradient-descent is established to update the parameters of CMAC. In another similar work, authors in [7] proposed an adaptive CMAC neural control (ACNC) system with a PI-type learning algorithm and applied it to control the chaotic systems. In 2012, a CMAC-based adaptive backstepping control (CABC) system was proposed [15]. This method combines backstepping and adaptive CMAC (ACMAC) and is applied to synchronize two Genesio chaotic systems. Chiu [4] designed an adaptive output recurrent CMAC (AORCMAC) to control wheeled inverted pendulums. This research aimed to derive a controller capable of balancing the pendulums. In order to control the position of a magnetic levitation system, a dynamic Petri FCMAC (DPFCMAC) was developed in [16]. To enhance the computational efficiency and reduction of firing rules, Lin et al. [13] proposed a function-link CMAC (FLFCMAC) that utilizes an approach called TOPSIS. A similar research was done in 2018 for a MIMO system, and the tracking performance was more satisfactory than other types of controllers [8]. A sliding mode interval type-2 wavelet FCMAC (T2WFCMAC)-based control system is designed for the DC–DC converters in [14]. The suggested control strategy was reported to be robust against input voltage variations and load resistance variations. Recent studies modified the CMAC-based controllers and offered an interval type-2 fuzzy brain emotional structure to improve the response and performance for synchronization. For example, in [10], the control system combines a novel wavelet interval type-2 fuzzy quad-function-link brain emotional controller and a robust controller. In 2021, another similar work was done by Lin et al. [17]. This research also offered a new controller, the self-organizing double function-link brain emotional learning controller (SO-DFL-BELC), which applies to MIMO nonlinear systems. The self-organizing mechanism decreases the number of neurons, contributes to the robustness against sharp changes in the input, and reduces the computation time. We chose a caterpillar robot to test our method like the above studies. A brief description of control strategies implemented on this robot is reported

in the next paragraph.

The caterpillar robot was first introduced by a set of nonlinear MIMO equations [6]. Different controllers have been designed to control the motion of this robot. In [22], an optimized PD torque controller was designed using Multi-Layer Perceptron. High trajectory tracking was achieved due to learning ability. Rahmani et al. conducted a combination of an adaptive Fractional Order PID (FOPID) and SMC to create a highly performed controller to deal with chattering. Bat Algorithm optimized parameters of this controller (BA) [24]. A Fraction Integral Terminal SMC (FITSMC) was designed in [23]. This controller is effective in reducing chattering and can control disturbances and uncertainties. In another work, FITSMC was combined with Super-Twisting Control (STC) [21]. This effort led to increasing the robustness of the system. Amiri et al. [3] designed an optimized SMC and Feedback Linearization Controller (FLC) using a genetic algorithm to investigate if employing an optimized SMC would result in high control effort compared with an optimized FLC or not. These controllers were compared in the presence of sensor noise, actuator disturbance, and parameter uncertainty. Despite lots of efforts, controlling this robot has remained a challenge, especially in terms of disturbance and uncertainty.

Our research aims to develop a TOPSIS Function-Link FCMAC (TFLFCMAC) controller for MIMO nonlinear systems that undergo external disturbance and uncertainties. An interval fuzzy type-2 controller is added to the second control effort to enhance the system's robustness against uncertainties. Each time step adapts both controllers' parameters. A Lyapunov candidate is chosen to prove the stability of the closed-loop system and derive the adaptation rules. Finally, the proposed control system is utilized to control the motion of a caterpillar robot. The suggested method's superiority is verified by comparing its root mean square error of tracking to some recent control strategies. It is worth highlighting that all the numerical simulations were inspected by considering external disturbances and uncertainties.

The second section is dedicated to formulating the general form of inspected nonlinear open-loop system that undergoes external disturbances and uncertainties. Besides, an ideal controller is suggested. In the next section, the comprising part of TFLFCMAC and IT2FLC will be introduced in detail. Moreover, proof of the stability of the closed-loop system is presented. Following this, the adaptation laws that update both controller parameters are derived. The numerical simulations are investigated on a caterpillar system. The last section is dedicated to reporting the concluding remarks.

2 Problem formulation

A nonlinear MIMO state-space equation can be assumed as:

$$\boldsymbol{x}^{(n)}(t) = \boldsymbol{A}(\boldsymbol{x}(t)) + \boldsymbol{B}(\boldsymbol{x}(t))\boldsymbol{u}(t) + \boldsymbol{d}(t), \tag{1}$$

where $\boldsymbol{x}(t) = \begin{bmatrix} x_1(t) & x_2(t) & \cdots & x_m(t) \end{bmatrix}^T \in \boldsymbol{R}^m$ is the vector of outputs with a length of m and the input vectors is $\boldsymbol{u}(t) = \begin{bmatrix} u_1(t) & u_2(t) & \cdots & u_m(t) \end{bmatrix}^T \in \mathbb{R}^m$; the external disturbance is denoted by $\boldsymbol{d}(t) = \begin{bmatrix} d_1(t) & d_2(t) & \cdots d_m(t) \end{bmatrix}^T$. Also, the number of inputs and outputs is equal to m. $\boldsymbol{A}(\boldsymbol{x}(t)) \in \mathbb{R}^m$ is an unknown vector and $\boldsymbol{B}(\boldsymbol{x}(t)) \in \mathbb{R}^{m\times m}$ is considered as some unknown square matrix. If this nonlinear system undergoes uncertainties, then the state-space equation (1) will be as follows:

$$\boldsymbol{x}^{(n)}(t) = \boldsymbol{A_n}(\boldsymbol{x}(t)) + \Delta\boldsymbol{A_n}(\boldsymbol{x}(t)) + \boldsymbol{B_n}(\boldsymbol{x}(t))\boldsymbol{u}(t)) + \Delta\boldsymbol{B_n}(\boldsymbol{x}(t))\boldsymbol{u}(t)) + \boldsymbol{d}(t), \tag{2}$$

where $\Delta\boldsymbol{B}_n(\boldsymbol{x}(t))$ and $\Delta\boldsymbol{A}_n(\boldsymbol{x}(t))$ are the variation of nominal values $\boldsymbol{B}_n(\boldsymbol{x}(t))$ and $\boldsymbol{A}_n(\boldsymbol{x}(t))$, respectively due to some uncertainties in (2).

Assumption 1: The value of every element in $\boldsymbol{d}(t)$ is bounded. Besides, $\boldsymbol{A}(\boldsymbol{x}(t))$ and $\boldsymbol{B}(\boldsymbol{x}(t))$ are continuous and limited functions. Without loss of generality, $\boldsymbol{B}(\boldsymbol{x}(t))$ is a diagonal matrix.

Let's consider a desired reference trajectory vector $\boldsymbol{r}(t)$ that the system trajectory $\boldsymbol{x}(t)$ should track. An appropriate control law is required to be designed to reach a satisfactory tracking performance of the closed-loop system. Tracking error can be defined as (3):

$$\boldsymbol{e}(t) \triangleq \boldsymbol{r}(t) - \boldsymbol{x}(t) \in \mathbb{R}^m, \tag{3}$$

where

$$\boldsymbol{e} = \begin{bmatrix} e_1 & e_2 & \cdots & e_m \end{bmatrix}^T, \tag{4}$$

and vector of system tracking error is expressed as:

$$\boldsymbol{E}(t) = \begin{bmatrix} \boldsymbol{e} & \dot{\boldsymbol{e}} & \cdots & \boldsymbol{e}^{(n)} \end{bmatrix} \in \mathbb{R}^{mn}, \tag{5}$$

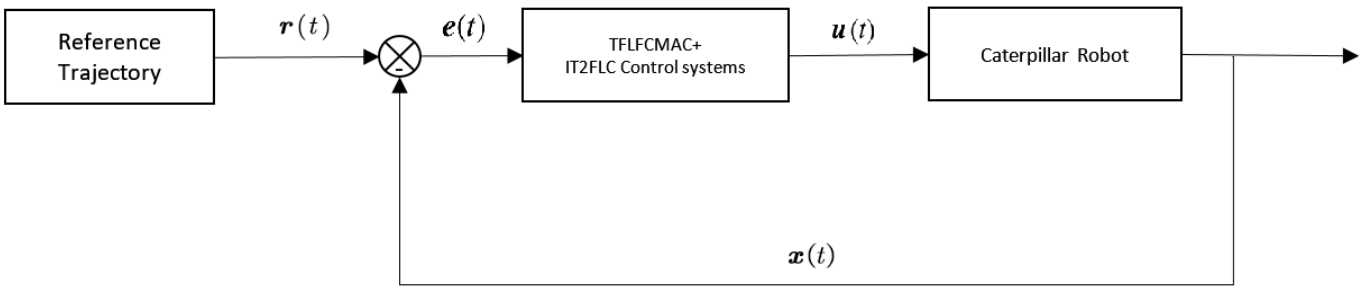


Figure 1: Block diagram of the closed-loop system

the aforementioned vector (5) consists of m multiplied by n elements. It is worth noting that there exists an ideal controller (6) that is capable of achieving the desired tracking performance:

$$\boldsymbol{u}_{ideal}(t) = \boldsymbol{B}^{-1}(\boldsymbol{x}(t))[\boldsymbol{r}^{(n)}(t) - \boldsymbol{A}_n(\boldsymbol{x}(t)) - \Delta\boldsymbol{A}_n(\boldsymbol{x}(t)) - \Delta\boldsymbol{B}_n(\boldsymbol{x}(t))\boldsymbol{u}_{ideal}(t - t_s) - \boldsymbol{d}(t) - \boldsymbol{K}\boldsymbol{E}(t)], \tag{6}$$

where t_s is the step size of the solver, and $\boldsymbol{K} \in \mathbb{R}^{m\times mn}$ is the feedback gain matrix and is expressed by (7)

$$\boldsymbol{K} = \begin{bmatrix} k_{n1} & 0 & 0 & \cdots & k_{i1} & 0 & 0 & \cdots & k_{11} & 0 & 0 \\ 0 & \ddots & 0 & \cdots & 0 & \ddots & 0 & \cdots & 0 & \ddots & 0 \\ 0 & 0 & k_{nm} & \cdots & 0 & 0 & k_{im} & \cdots & 0 & 0 & k_{1m} \end{bmatrix}, \tag{7}$$

elements of the gain feedback matrix should be selected as nonzero positive numbers. If the ideal controller (6) is adopted in (1), then the dynamics of error will be resulted as:

$$\boldsymbol{e}^{(n)}(t) + \boldsymbol{K}\boldsymbol{e}(t) = 0. \tag{8}$$

The error dynamics (8) can reach a zero vector by selecting suitable values of feedback gain matrix resulting in the tracking of the desired trajectory; nevertheless, the ideal controller (6) is not straightforward to reach in many practical situations due to the effects of unknown external disturbance and uncertainties. Thus, an alternative control law is required to remedy the effects of these undesirable signals. The TFLFCMAC+IT2FLC control system schematics is illustrated in Fig. 1. As it can be seen, the total control effort is designed as (9)

$$\boldsymbol{u}(t) = \boldsymbol{u}_{TFLFCAMC}(t) + \boldsymbol{u}_{IT2FLC}(t). \tag{9}$$

Where in $\boldsymbol{u}_{TFLFCAMC}$ is the TOPSIS function-link CMAC controller output, and $\boldsymbol{u}_{IT2FLC}$ denotes the interval type-2 fuzzy robust controller. In order to achieve satisfactory tracking, a controller is needed to mimic the ideal controller (6), which is accomplished by the first control input, and the second one is designed to eliminate the estimation error.

3 Control system design

The proposed closed-loop system is comprised of a TFLCMAC, an IT2FL controller, a function link network, and sliding surface blocks. The block diagram of the closed-loop system and proposed control system for the caterpillar robot are illustrated in Fig. 1 and Fig. 2, respectively. These blocks will be discussed in this section.

3.1 Function-link network

In order that the function estimation may be ameliorated, an FLN is approved. The activation functions in the input layer can be trigonometric, Gaussian, or a set of orthogonal polynomials. No matter the type of these activation functions, they need to be linearly independent. The output of the first layer is multiplied by a connecting weight. Next, the second layer performs an algebraic sum of the production results. The formulation of the FLN may be expressed as follows:

$$\boldsymbol{w}(:, j) = \boldsymbol{P}(:, :, j)^T\boldsymbol{\phi} \qquad \text{for } j = 1, \ldots, m, \tag{10}$$

with

$$\boldsymbol{s} = \begin{bmatrix} s_1 & \cdots & s_m \end{bmatrix}^T \in \mathbb{R}^m, \tag{11}$$

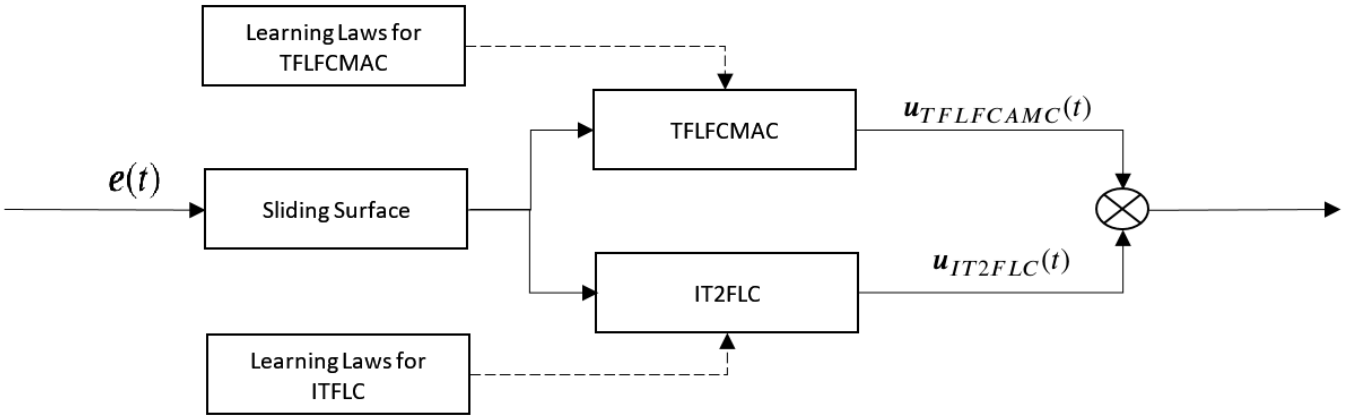


Figure 2: Block diagram of the suggested control system for the caterpillar robot

$$\boldsymbol{\Phi} = \begin{bmatrix} \varphi_1(\boldsymbol{s}) & \cdots & \varphi_{n_m}(\boldsymbol{s}) \end{bmatrix}^T \in \mathbb{R}^{n_m}. \tag{12}$$

$$\boldsymbol{w} = \begin{bmatrix} w_{11} & \dots & w_{1j} & \dots & w_{1m} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ w_{l1} & \dots & w_{lj} & \dots & w_{lm} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ w_{n_l 1} & \dots & w_{n_l j} & \dots & w_{n_l m} \end{bmatrix}. \tag{13}$$

$$\boldsymbol{p}(:,:,j) = \begin{bmatrix} w_{11} & \dots & w_{1j} & \dots & w_{1m} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ w_{lk1} & \dots & w_{lkj} & \dots & w_{lm} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ w_{n_l n_k 1} & \dots & w_{n_l n_k j} & \dots & w_{n_l m} \end{bmatrix}^T, \tag{14}$$

where $\boldsymbol{s}$ is the input vector, $\boldsymbol{\Phi}$ denotes the expanded space vector, $\boldsymbol{w} \in \mathbb{R}^{n_i n_j \times m = n_l \times m}$ is the output matrix of the FLN, and $\boldsymbol{P} \in \mathbb{R}^{n_m \times n_i n_j n_k \times m}$ is a three-dimensional matrix of the connecting weights. Furthermore, $\boldsymbol{w}(:,j)$ is the j-th column of $\boldsymbol{w}$ and $\boldsymbol{P}(:,:,j) \in \mathbb{R}^{n_m \times n_i n_j n_k}$ is a 2-dimensional matrix related to the j-th output. Different kinds of basis functions can be used in function expansion such as trigonometric and triangular. The latter is employed in this research. For instance, for a 3-input 3-output system, expanded space is possible to be selected as (15):

$$\boldsymbol{\Phi} = \begin{bmatrix} 1 \\ s_1 \\ \sin(\pi s_1) \\ \cos(\pi s_1) \\ s_2 \\ \sin(\pi s_2) \\ \cos(\pi s_3) \\ s_3 \\ \sin(\pi s_3) \\ \cos(\pi s_3) \\ s_1 s_2 s_3 \end{bmatrix} \in \mathbb{R}^{11}. \tag{15}$$

3.2 Description of the proposed MIMO TFLFCMAC

The FCMAC network is composed of seven layers. An input layer, an association memory layer, a TOPSIS transition layer, a receptive-field layer, a weight memory layer, and an output layer. Generally, it can be explained as a MIMO

fuzzy inference engine.

$$R^l : \text{If } s_1 \text{ is } \mu_{1jk} \text{ and } s_2 \text{ is } \mu_{2jk} \text{ and } , \cdots , s_{n_i} \text{ is } \mu_{n_ijk} \text{then } \boldsymbol{u} \text{ is } \boldsymbol{w}_{jk}. \tag{16}$$

The number of inputs is denoted by n_i, and each layer is comprised of n_k blocks. Thus, there exists $n_l = n_k n_i$ rules in this fuzzy system. The consequent part is included a singleton weight. However, it seems complicated to demonstrate this system for more than 2 inputs, which is our case in the numerical simulations section.

3.2.1 Input layer

Tracking error, sliding surface, etc. can be introduced as the input of the network (all having the same dimension as output vector). A sliding surface for each output is adopted in this research. Furthermore, this input vector is possible to be normalized. Normalization is achievable for each input variable by using the following formula:

$$s_{in} = \frac{(s_i - s_{i_{min}})}{s_{i_{max}} - s_{i_{min}}}, \tag{17}$$

where s_{in} is the normalized input, s_i is the i-th element of the input vector, $s_{i_{max}}$ and $s_{i_{min}}$ are the maximum and minimum value of s_i, respectively. Still, it requires prior knowledge of the closed-loop system and reference trajectory to determine these boundaries for each input element.

3.2.2 Association memory space (membership function)

Each layer of FCMAC can be considered as a separate look-up table. For each layer, there exist several blocks of the same kind. Depending on the input position, some of these blocks may be more activated than others and outputs numeric values closer to one. In addition, MFs that are not in the vicinity of the input lead to infinitestimal values. The critical difference between type-1 fuzzy and FCMAC systems can be figured out more straightforwardly in the look-up table scheme. In fuzzy type-1 systems, all rules that are possible to be fired are used. In other words, the number of rules for each table is $n_k^{n_i}$; therefore, the total number of fuzzy rules is $n_l = n_j n_k^{n_i}$, whereas the number of fuzzy rules in an FCMAC system is less than fuzzy type-1 systems. For each look-up table, it is only possible to multiply the blocks with the same index for each input, which means the number of rules for each look-up table is equal to n_k, leading to an overall number of $n_l = n_j n_k$ rules. In fact, constructing fewer fuzzy rules leads to faster computations and eases of implementation on hardware. As a result, the tracking performance may be weaker than fuzzy type-1 or type-2 systems. Unfortunately, authors of [13] calculated the number of fuzzy rules for a fuzzy type-1 system and reported this figure for an FCMAC system.

The number of parameters is specified depending on the type of MFs utilized in this layer. Gaussian or wavelet-type MFs are adopted in many resources since the derivative of such functions is easy to calculate; hence, Gaussian-type MFs are practiced in this study. Specifically, there are two parameters, mean and variance, for each block to be updated in each time step, which is 10 microseconds. Consequently, there are $2n_l = 2n_j n_k$ parameters in the antecedent part of the rules needing to change appropriately. Indeed, employing MFs in this layer can contribute to producing a smooth output, which is robust against variation of input vectors.

3.2.3 TOPSIS layer

This paper employs the TOPSIS layer formulated in [13]. This approach is applied to delete less important MFs, contributing to reducing the number of rules and enhancing the computation speed. A threshold value is generated by utilizing seven steps. For briefness, these steps are not included here, and only the threshold is presented here.

$$d_{th} = \max(C^*_{m_x}). \tag{18}$$

Readers may find detailed explanations about (18) in [13].

3.2.4 Receptive-field space (hypercube space)

After calculating the MFs values in association memory space, an intersection operator (e.g., product, minimum, etc.), should be used. If the product is applied to each rule, the output of this layer will be:

$$r_{jk} = \prod_{i=1}^{n_i} \mu_{ijk}(s_i) = \prod_{i=1}^{n_i} exp\left[\left(\frac{s_i - m_{ijk}}{v_{ijk}}\right)^2\right]. \tag{19}$$

Also, let's define:

$$\boldsymbol{r} = \begin{bmatrix} r_{11} & \cdots & r_{1n_k} & r_{21} & \cdots & r_{2n_k} & \cdots & r_{n_j1} & \cdots & r_{n_jn_k} \end{bmatrix}^T \in \mathbb{R}^{n_jn_k}, \tag{20}$$

or,

$$\boldsymbol{r} = \begin{bmatrix} r_1 & r_2 & \cdots & r_l & \cdots & r_{n_l} \end{bmatrix}^T \in \mathbb{R}^{n_l}. \tag{21}$$

3.2.5 Weight memory layer

The consequent part of each rule consists of adjustable singleton MFs. FLN outputs contribute to building these weights. The vector of weights for each output is the output of FLN that is calculated by (10).

3.2.6 Output space

It was mentioned that only some of the MFs in the association layer are activated for each input space. The output layer performs an algebraic weighted sum of these MFs. It is possible to express:

$$\boldsymbol{u} = \boldsymbol{w}^T \boldsymbol{r}. \tag{22}$$

3.3 Deriving adaptation laws for TFLFCAMC

Two controllers were proposed in previous sections. The ideal controller can achieve a desirable tracking performance; nevertheless, it is impossible to obtain it due to different problems such as uncertainties, disturbance, and sometimes unavailability of the exact nonlinear state-space equation. Turning to possible alternatives, a controller similar to the ideal one is required to be designed. It is feasible to update means, variances, and output weights in each time sample, resulting in a stable closed-loop nonlinear system. A Lyapunov candidate needs to be introduced, but some modifications must be made to the matrices leading to control input. There exists an optimal controller $\boldsymbol{u}^*_{TFLFCMAC}$ such that:

$$\boldsymbol{u}_{ideal} = \boldsymbol{u}^*_{TFLFCMAC} + \boldsymbol{\varepsilon}. \tag{23}$$

Let's consider

$$\boldsymbol{u}^*_{TFLFCMAC}(j) = (\boldsymbol{P}^*(:,:,j)^T \boldsymbol{\phi})\boldsymbol{r}^*, \tag{24}$$

where the minimum reconstructed error is denoted by $\boldsymbol{P}^*$, $\boldsymbol{v}^*$, $\boldsymbol{m}^*$, and $\boldsymbol{r}^*$ are in respect the optimally desired parameter vectors of $\boldsymbol{P}$, $\boldsymbol{v}$, $\boldsymbol{m}$, and $\boldsymbol{r}$ in the suitable controller. However, this should be expressed that these optimal parameters are imaginary because there is no way to calculate them exactly. Thus, an approximation of these parameters needs to be available to construct the controller.

$$\boldsymbol{u} = \hat{\boldsymbol{u}}_{TFLFCMAC} + \boldsymbol{u}_{IT2FLC}. \tag{25}$$

By subtracting (23) from (24), the error of control input estimation will be revealed as:

$$\widetilde{\boldsymbol{u}}(j) = \boldsymbol{u}^*(j) - \boldsymbol{u}(j) = \boldsymbol{\phi}^T \widetilde{\boldsymbol{P}}(:,:,j)\boldsymbol{r}^* + \boldsymbol{\phi}^T \hat{\boldsymbol{P}}(:,:,j)\widetilde{\boldsymbol{r}} + \boldsymbol{\varepsilon} - \boldsymbol{u}_{IT2FLC}. \tag{26}$$

Moreover, the estimation error of the following parameters can be expressed as:

$$\widetilde{\boldsymbol{P}} = \boldsymbol{P}^* - \hat{\boldsymbol{P}}. \tag{27}$$

$$\widetilde{\boldsymbol{r}} = \boldsymbol{r}^* - \hat{\boldsymbol{r}}, \tag{28}$$

it is possible to report the Taylor expansion of $\widetilde{\boldsymbol{r}}$ as:

$$\widetilde{\boldsymbol{r}} = \boldsymbol{r}_m^T \widetilde{\boldsymbol{m}} + \boldsymbol{r}_v^T \widetilde{\boldsymbol{v}} + \boldsymbol{O}_t, \tag{29}$$

where $\boldsymbol{O}_t$ is the vector of higher-order terms. Also, $\boldsymbol{r}_m$ and $\boldsymbol{r}_v$ may be described as follows:

$$\boldsymbol{r}_m = \text{Jacobian}(r, m). \tag{30}$$

$$\boldsymbol{m} = \begin{bmatrix} m_{111} & m_{211} & \cdots & m_{ijk} & \cdots & m_{n_i n_j n_k} \end{bmatrix}^T \in \mathbb{R}^{n_i n_j n_k}, \tag{31}$$

$$\boldsymbol{r}_m = \begin{bmatrix} \frac{\partial r_{11}}{\partial m_{111}} & \cdots & \frac{\partial r_{11}}{\partial m_{ijk}} & \cdots & \frac{\partial r_{11}}{\partial m_{n_i n_j n_k}} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \frac{\partial r_{jk}}{\partial m_{111}} & \cdots & \frac{\partial r_{jk}}{\partial m_{ijk}} & \cdots & \frac{\partial r_{jk}}{\partial m_{n_i n_j n_k}} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \frac{\partial r_{n_j n_k}}{\partial m_{111}} & \cdots & \frac{\partial r_{n_j n_k}}{\partial m_{ijk}} & \cdots & \frac{\partial r_{n_j n_k}}{\partial m_{n_i n_j n_k}} \end{bmatrix} \in \mathbb{R}^{n_i n_j \times n_i n_j n_k}, \tag{32}$$

$$\boldsymbol{r}_v = \text{Jacobian}(r, v), \tag{33}$$

$$\boldsymbol{v} = \begin{bmatrix} v_{111} & v_{211} & \cdots & v_{ijk} & \cdots & v_{n_i n_j n_k} \end{bmatrix}^T \in \mathbb{R}^{n_i n_j n_k}, \tag{34}$$

$$\boldsymbol{r}_v = \begin{bmatrix} \frac{\partial r_{11}}{\partial v_{111}} & \cdots & \frac{\partial r_{11}}{\partial v_{ijk}} & \cdots & \frac{\partial r_{11}}{\partial v_{n_i n_j n_k}} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \frac{\partial r_{jk}}{\partial v_{111}} & \cdots & \frac{\partial r_{jk}}{\partial v_{ijk}} & \cdots & \frac{\partial r_{jk}}{\partial v_{n_i n_j n_k}} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \frac{\partial r_{n_j n_k}}{\partial v_{111}} & \cdots & \frac{\partial r_{n_j n_k}}{\partial v_{ijk}} & \cdots & \frac{\partial r_{n_j n_k}}{\partial v_{n_i n_j n_k}} \end{bmatrix} \in \mathbb{R}^{n_i n_j \times n_i n_j n_k}. \tag{35}$$

The optimal receptive field vector is:

$$\boldsymbol{r}^* = \hat{\boldsymbol{r}} + \boldsymbol{r}_m^T \widetilde{\boldsymbol{m}} + \boldsymbol{r}_v^T \widetilde{\boldsymbol{v}} + \boldsymbol{O}_t. \tag{36}$$

Substituting (29) and (36) into (26) yields:

$$\widetilde{\boldsymbol{u}}(j) = \boldsymbol{\phi}^T \widetilde{\boldsymbol{P}}(:,:,j)\hat{\boldsymbol{r}} + \boldsymbol{\phi}^T \hat{\boldsymbol{P}}(:,:,j)\left(\boldsymbol{r}_m^T \widetilde{\boldsymbol{m}} + \boldsymbol{r}_v^T \widetilde{\boldsymbol{v}} + \boldsymbol{O}_t\right) + \boldsymbol{\xi}(t) - \boldsymbol{u}_{IT2FLC}. \tag{37}$$

$$\boldsymbol{\xi}(t) = \boldsymbol{\phi}^T \widetilde{\boldsymbol{P}}(:,:,j)\boldsymbol{r}_m^T \widetilde{\boldsymbol{m}} + \boldsymbol{\phi}^T \widetilde{\boldsymbol{P}}(:,:,j)\boldsymbol{r}_v^T \widetilde{\boldsymbol{v}} + \boldsymbol{\phi}^T \boldsymbol{P}^*(:,:,j)\boldsymbol{O}_t + \boldsymbol{\varepsilon}. \tag{38}$$

Let's define the derivate of the sliding mode vector as below:

$$\dot{\boldsymbol{s}} \triangleq \ddot{\boldsymbol{e}}(t) + \boldsymbol{\theta}_1 \dot{\boldsymbol{e}}(t) + \boldsymbol{\theta}_2 \boldsymbol{e}(t) = \boldsymbol{u}_{ideal} - \boldsymbol{u}. \tag{39}$$

In order to guarantee the stability of the closed-loop system and derive adaptation rules, a Lyapunov candidate is chosen as:

$$V_1 = \frac{1}{2}\boldsymbol{s}^2(t) + \frac{1}{2\mu_p} tr(\widetilde{\boldsymbol{P}}^T(:,:,j)\boldsymbol{P}(:,:,j)) + \frac{1}{2\mu_m}\widetilde{\boldsymbol{m}}^T\widetilde{\boldsymbol{m}} + \frac{1}{2\mu_m}\widetilde{\boldsymbol{v}}^T\boldsymbol{v}. \tag{40}$$

where $tr(.)$ denotes the trace of some matrix. Differentiating (40) with respect to time yields:

$$\dot{V}_1 = \boldsymbol{s}(t)\dot{\boldsymbol{s}}(t) + \frac{1}{\mu_p} tr(\widetilde{\boldsymbol{P}}^T(:,:,j)\boldsymbol{P}(:,:,j)) + \frac{1}{\mu_m}\widetilde{\boldsymbol{m}}^T\widetilde{\boldsymbol{m}} + \frac{1}{\mu_m}\widetilde{\boldsymbol{v}}^T\boldsymbol{v}. \tag{41}$$

$$\begin{aligned} \dot{V}_1 = s(t)[\boldsymbol{\phi}^T \widetilde{\boldsymbol{P}}(:,:,j)\hat{\boldsymbol{r}} + \boldsymbol{\phi}^T \hat{\boldsymbol{P}}(:,:,j)(\boldsymbol{r}_m^T \widetilde{\boldsymbol{m}} + \boldsymbol{r}_v^T \widetilde{\boldsymbol{v}}) + \xi(t) - u_{IT2FLC}(j)] \\ + \frac{1}{\mu_p} tr(\widetilde{\boldsymbol{P}}^T(:,:,j)\dot{\widetilde{\boldsymbol{P}}}(:,:,j)) + \frac{1}{\mu_m}\widetilde{\boldsymbol{m}}^T\dot{\widetilde{\boldsymbol{m}}} + \frac{1}{\mu_v}\widetilde{\boldsymbol{v}}^T\dot{\widetilde{\boldsymbol{v}}}. \end{aligned} \tag{42}$$

$$\dot{V}_1 = [\sum_{j=1}^{n_j} \widetilde{\boldsymbol{P}}^T(:,:,j)(\boldsymbol{s}(t)\boldsymbol{r}\boldsymbol{\phi} - \frac{\dot{\hat{\boldsymbol{P}}}(:,:,j)}{\mu_p}] + \boldsymbol{m}^T[\boldsymbol{s}(t)\boldsymbol{r}_m\hat{\boldsymbol{P}}^T(:,:,j)\boldsymbol{\phi} - \frac{\dot{\hat{\boldsymbol{m}}}}{\mu_m}] + \boldsymbol{v}^T[\boldsymbol{s}(t)\boldsymbol{r}_v\hat{\boldsymbol{P}}^T(:,:,j)\boldsymbol{\phi} - \frac{\dot{\hat{\boldsymbol{v}}}}{\mu_v}] + \boldsymbol{s}(t)\boldsymbol{\xi}(t). \tag{43}$$

the following adaptation rules can be derived:

$$\dot{\hat{\boldsymbol{P}}}(:,:,j) = \mu_p \boldsymbol{s}(j)\boldsymbol{r}\boldsymbol{\phi}^T. \tag{44}$$

$$\dot{\hat{\boldsymbol{m}}} = \sum_{j=1}^{m} \mu_m \boldsymbol{s}(j)\boldsymbol{r}_m \hat{\boldsymbol{P}}^T(:,:,j)\boldsymbol{\phi}. \tag{45}$$

$$\dot{\hat{\boldsymbol{v}}} = \sum_{j=1}^{m} \mu_v \boldsymbol{s}(j)\boldsymbol{r}_v \hat{\boldsymbol{P}}^T(:,:,j)\boldsymbol{\phi}. \tag{46}$$

If these rules are substituted in (44)-(46), then (43) can be rewritten as:

$$\dot{V}_1 = \boldsymbol{s}(t)\boldsymbol{\xi}(t). \tag{47}$$

Nevertheless, the last term in (43) remains and needs to be compensated by another controller. Thus, that motivates us to adopt an adaptive interval fuzzy type-2 controller to achieve closed-loop stability. In addition, the approximation error term $\boldsymbol{\xi}(t)$ is assumed to be bounded; otherwise, $\dot{V}_1$ will become positive as stated in (47).

3.4 Adaptive interval type-2 fuzzy logic controller

An interval type-2 fuzzy logic controller (IT2FLC) is adopted to compensate for the approximation error as a secondary controller. Different choices can be made to specify or design an IT2FLC. Using the step-by-step procedure introduced by Mendel in [18], an IT2 FS, whose properties are illustrated in Table 1, is utilized.

Property	Selected Property
Rules Type	TSK (A2-C0)
Fuzzifier	IT2 Non-singleton
t-norm	Type-reduction + D
Number of input variables	p
Names of terms per input	Low, Medium, High
Number of rules	M
Number of output variables	1
Type of UMFs	Primary Gaussian
Type of LMFs	Primary Gaussian
FOU parameters	Pre-specified

Table 1: selected IT2FLC and its properties

Consequently, by selecting the properties as mentioned earlier, the structure of the l-th rule for TSK IT2FLC is:

$$\widetilde{R}^l_{TSK}: \text{ If } x_1(t) \text{ is } \widetilde{F}^l_1 \text{ and} \dots \ x_p(t) \text{ is } \widetilde{F}^l_p, \text{ then } u_l \text{ is } \widetilde{g}^l(x_1,\dots,x_p) \text{ for } l = 1,\dots,M, \tag{48}$$

if the consequent parts are crisp numbers (48) will be transformed into:

$$\widetilde{R}^l_{TSK}: \text{ If } x_1(t) \text{ is } \widetilde{F}^l_1 \text{ and} \dots \ x_p(t) \text{ is } \widetilde{F}^l_p, \text{ then } y \text{ is } g_l = c^l_0 + \ c^l_1 x_1(t) + \dots + c^l_p x_p(t) \text{ for } l = 1,\dots,M, \tag{49}$$

which are often called 'A2C0' type IT2 TSK rules. In (49), $x_i(t)$ denotes the i-th sliding surface of the dynamical system under control, and the consequent is the produced control effort which is the weighted sum of the sliding mode variables. Primary Gaussian-type MFs with uncertain standard deviations, whose means are located at $x_i(t)$ are exploited as IT2 non-singleton fuzzifier. By applying the unnormalized type reduction + defuzzification, the output of the IT2FLC will be:

$$u^U_{TSK} = \sum_{l=1}^{M} g^l(\boldsymbol{x}') \left[\frac{\underline{f}^l(\boldsymbol{x}') + \overline{f}^l(\boldsymbol{x}')}{2} \right], \tag{50}$$

where,

$$g^l(\boldsymbol{x}') = c^l_0 + \ c^l_1 x_1(t) + \dots + c^l_p x_p(t), \tag{51}$$

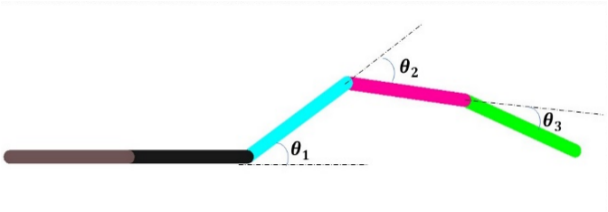


Figure 3: schematic of the caterpillar robot

assuming $c_0^l = 0$ then (51) can be rewritten in vector form:

$$g^l(\boldsymbol{x}') = (\boldsymbol{C}^l)^T \boldsymbol{X}, \tag{52}$$

where $\boldsymbol{C}^l = \begin{bmatrix} c_1^l & c_2^l & \cdots & c_p^l \end{bmatrix}$ is the output weight vector and $\boldsymbol{X} = \begin{bmatrix} x_1(t) & x_2(t) & \cdots & x_p(t) \end{bmatrix}$ is the input vector. Moreover, $\underline{f}^l(\boldsymbol{x}')$ and $\overline{f}^l(\boldsymbol{x}')$ denote the endpoints of the firing interval. To compute these values, it is possible to use the following formulas:

$$\underline{f}^l(\boldsymbol{x}') = \prod_{i=1}^{p} \underline{f}^l(x_i(t)). \tag{53}$$

$$\overline{f}^l(\boldsymbol{x}') = \prod_{i=1}^{p} \underline{f}^l(x_i(t)). \tag{54}$$

The output vector weight of each rule needs to change in a way that reduces the approximation error. To achieve this, first, consider the following Lyapunov candidate:

$$V_2 = \frac{1}{2} S^2 + \frac{1}{2k_c} \boldsymbol{C}^T \boldsymbol{C}. \tag{55}$$

By taking the time derivative of (55), the following inequality is used to provide the system with closed-loop stability:

$$\dot{V}_2 = \boldsymbol{S}\left[\boldsymbol{\xi} - \boldsymbol{u}_{FC}\right] + \frac{1}{k_c} \boldsymbol{C}\dot{\boldsymbol{C}} \leq 0. \tag{56}$$

As it is assumed that $\boldsymbol{\xi}$ is bounded, that is $|\boldsymbol{\xi}| \leq \boldsymbol{\xi}_m$, the right hand side of (56) will transform to an equality if $\boldsymbol{\xi} = \boldsymbol{\xi}_m$. Thus:

$$\dot{V}_2 = \boldsymbol{S}[\boldsymbol{\xi}_m - C] + \frac{1}{k_c} \boldsymbol{C}\dot{\boldsymbol{C}} = 0. \tag{57}$$

Considering the worst-case scenario, the following adaptation rule, (58), will always guarantee the closed-loop stability.

$$\dot{\boldsymbol{C}} = -k_c \boldsymbol{C}^{-1} \boldsymbol{S}\left[\boldsymbol{\xi}_m - \boldsymbol{C}\right]. \tag{58}$$

4 Simulation results and discussion

4.1 Example 1

In order to illustrate the effectiveness of the proposed control system, it is applied to a caterpillar robot [3]. This robot comprises five bars, and four sequential gaits develop its motion. Inputs are joints' torque generated by servo motors. A schematic of this robot is shown in Fig. 3. The following MIMO nonlinear space-state equations can express it.

$$\ddot{\boldsymbol{X}} = \boldsymbol{M}^{-1}(\boldsymbol{X})\left(-\boldsymbol{C}(\boldsymbol{X})\dot{\boldsymbol{X}}^2 - \boldsymbol{G}(\boldsymbol{X}) + \boldsymbol{D}\boldsymbol{u} + F_R \boldsymbol{L}_A\right) + \boldsymbol{d}(t), \tag{59}$$

where,

$$\boldsymbol{X}(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix}, \ \boldsymbol{u}(t) = \begin{bmatrix} u_1(t) \\ u_2(t) \\ u_3(t) \end{bmatrix}, \boldsymbol{d}(t) = \begin{bmatrix} \cos(50\pi t) \\ \cos(30\pi t) \\ \sin(25\pi t) \end{bmatrix}, \tag{60}$$

are the state, input, and disturbance vectors, respectively. In addition, the system matrices are as follows:

$$\boldsymbol{M}(\boldsymbol{X}) = \frac{ml^2}{2}\begin{bmatrix} 14 & 9cos\ (x_1 - x_2) & 3\cos(x_1 - x_3) \\ 6\cos(x_1 - x_2) & 8 & 3\cos(x_2 - x_3) \\ 3\cos(x_1 - x_3) & -3\cos(x_2 - x_3) & 2 \end{bmatrix}. \tag{61}$$

$$\boldsymbol{C}(\boldsymbol{X}) = \frac{ml^2}{6}\begin{bmatrix} 0 & 9\sin(x_1 - x_2) & 3\sin(x_1 - x_3) \\ -9\sin(x_1 - x_2) & 0 & 3\sin(x_2 - x_3) \\ -3\sin(x_1 - x_3) & -3\sin(x_2 - x_3) & 0 \end{bmatrix}. \tag{62}$$

$$\boldsymbol{G}(\boldsymbol{X}) = \frac{mlg}{6}\begin{bmatrix} 5\cos(x_1) \\ 3\cos(x_2) \\ \cos(x_3) \end{bmatrix}. \tag{63}$$

$$\boldsymbol{D} = \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \end{bmatrix}. \tag{64}$$

$$\boldsymbol{L}_a = l\begin{bmatrix} \cos(x_1) + \mu\sin(x_1) \\ \cos(x_2) + \mu\sin(x_2) \\ \cos(x_3) + \mu\sin(x_3) \end{bmatrix}. \tag{65}$$

The system has uncertain mass (m) and length (l), which vary according to:

$$m = 1.1m_0, \tag{66}$$

$$l = 1.1l_0, \tag{67}$$

with the nominal values of $l_0 = 0.14$ (m) and $m_0 = 0.015$ (kg). Also, $F_R = 20$ (N) is the contact force, $\mu = 0.4$ is the friction coefficient, and $g = 9.81\frac{m}{s^2}$ denotes the gravitational constant. The initial system state and its derivatives are in respect $\boldsymbol{X}(0) = \begin{bmatrix}0.4 & 0.1 & -0.35\end{bmatrix}^T$ and $\dot{\boldsymbol{X}}(0) = \begin{bmatrix}0 & 0 & 0\end{bmatrix}^T$. The sliding surface (S) parameters are chosen as $k_p = 20$, $k_i = 0.1$, and $k_d = 1$. The numbers of layers and blocks for TFLFCMAC are $n_k = 3$ and $n_j = 1$, respectively. The learning rate parameter was chosen as $\mu_p = 0.5$, $\mu_m = 0.1$, and $\mu_v = 0.1$. The initial mean and standard deviation and vectors are respectively set as:

$$\begin{aligned} m(0) = [&-4.4\ -1.2\ 2\ -3.6\ -0.4\ 2.8\ -2.8\ 0.4\ 3.6\ -2\ 1.2\ 4.4 \\ &-4.4\ -1.2\ 2\ -3.6\ -0.4\ 2.8\ -2.8\ 0.4\ 3.6\ -2\ 1.2\ 4.4 \\ &-4.4\ -1.2\ 2\ -3.6\ -0.4\ 2.8\ -2.8\ 0.4\ 3.6\ -2\ 1.2\ 4.4]^T. \end{aligned} \tag{68}$$

$$v(0) = 0.8. * ones(36, 1). \tag{69}$$

$$P(0) = ones(12, 11, 3). \tag{70}$$

Besides, the upper and lower primary Gaussian MFs are selected with the following parameters:

$$m_u = \begin{bmatrix}-0.2 & 0 & 0.2\end{bmatrix} = m_l, \tag{71}$$

$$v_u = \begin{bmatrix}0.8 & 0.8 & 0.8\end{bmatrix}, \tag{72}$$

$$v_l = \begin{bmatrix}0.5 & 0.5 & 0.5\end{bmatrix}. \tag{73}$$

The learning rate constant is set as $\mu_C = 0.01$. All initial parameters are selected by trial and error; afterward, they are adjusted online. Tracking responses, tracking error, and control effort are illustrated in Fig. 4 – Fig. 7, respectively. The simulation results show that the proposed control system is robust against uncertainties and external disturbances. It achieves rapid convergence, and there exists no chattering phenomenon. Fig. 4 shows the system and desired trajectory, respectively, denoted by solid and dashed lines. These states approximately reach their set points

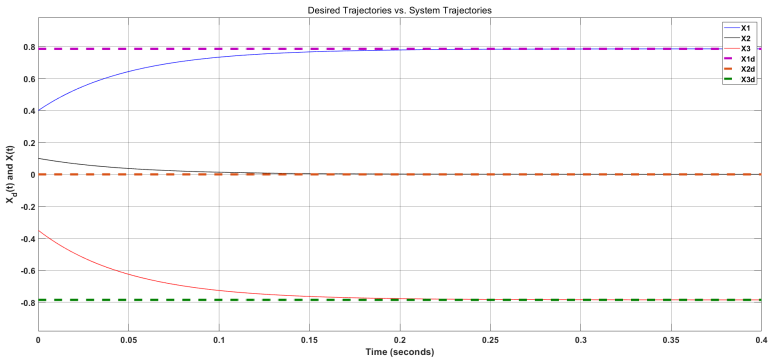


Figure 4: Tracking Responses for the First Set of Initial Conditions

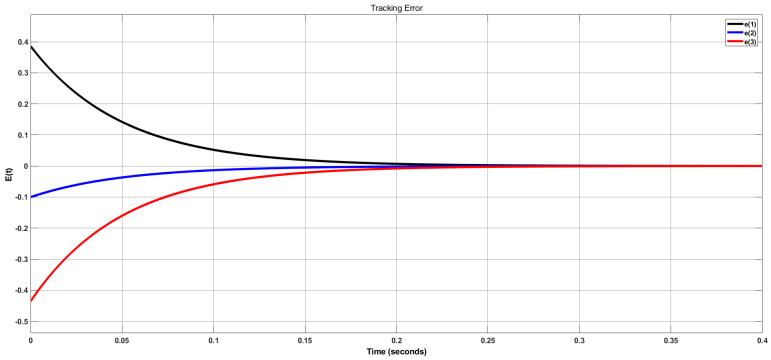


Figure 5: Generated Control IT2FLC Input for the First Set of Initial Conditions

in less than 0.2 seconds. Besides, there is no sign of overshoot and steady-state error. Tracking errors are depicted in Fig. 5 that converge to zero. The generated control input of TFLFCMAC and IT2FLC are illustrated in Fig. 6 and 7, respectively. At the first initial moments, the control inputs jump to a large value, which can be interpreted as the learning moments of the control system. After passing this learning phase, approximately until 0.2 seconds, a smooth signal is generated. In this interval, the controller stabilizes the robot, but in the next seconds, fluctuations are not noticeable. The number of fired rules in the learning phase changes due to changes in the adaptive threshold and reduces when the threshold reaches a constant value of 0.93. The comparisons of these simulation results with other controllers are listed in table 2. The test is repeated for a different set of initial conditions to assert the robustness against initial conditions. (see results in Fig. 8 - Fig. 11)

As indicated in table 2, the root mean square error (RMSE) is smaller than that for computed torque controller (CTC) [23], neural network computed torque controller (NNCTC) [23], and Neural Network Fraction Integral Terminal Sliding Mode Control (NN-FITSMC) [21]. Moreover, its stabilizing time, the time that states reach their set points (in MIMO systems, the maximum of several time values is considered), is less than CTC and NNCTC but 0.1 second higher

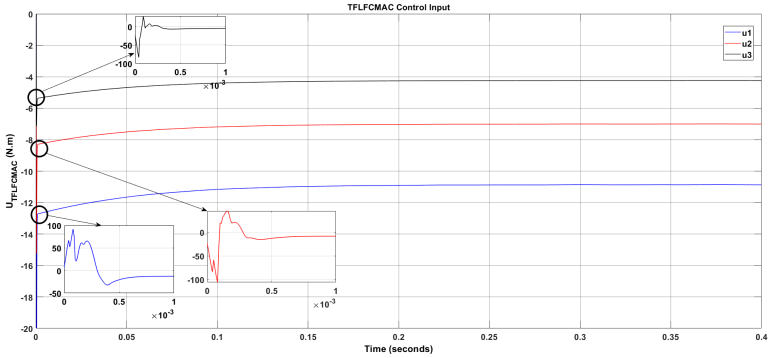


Figure 6: Tracking Errors for the First Set of Initial Conditions

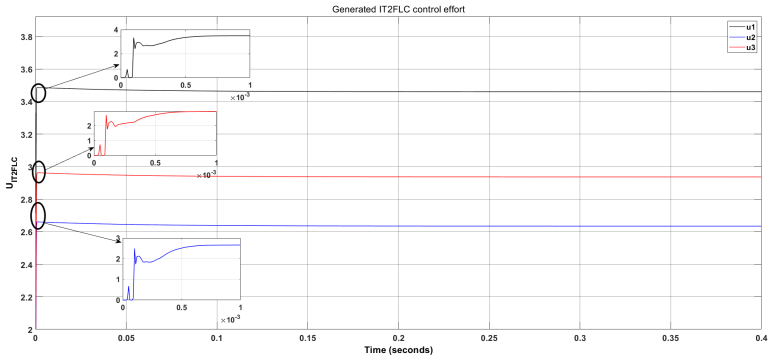


Figure 7: Generated Control TFLFCMAC Input for the First Set of Initial Conditions

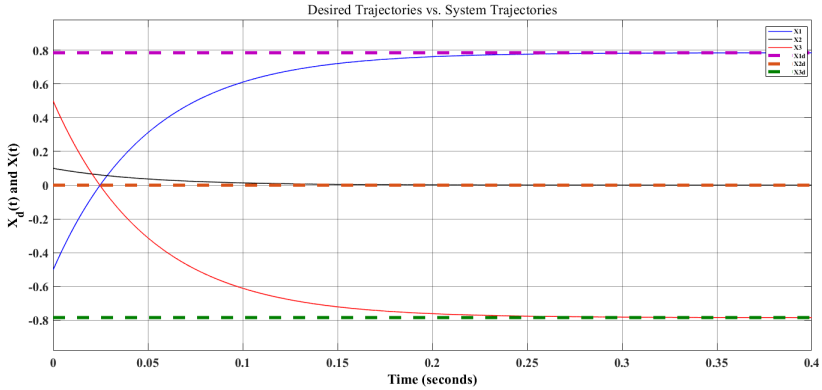


Figure 8: Tracking Responses for the Second Set of Initial Conditions

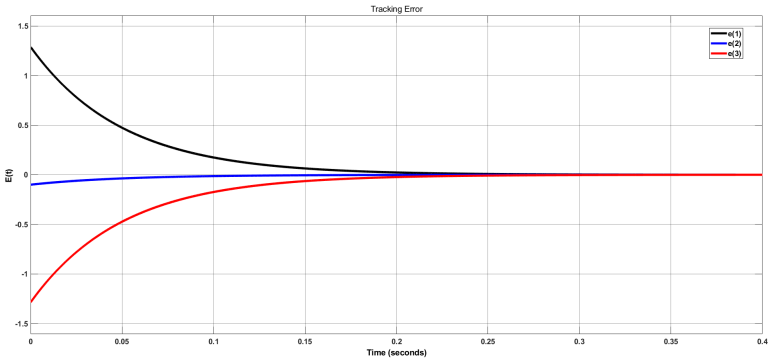


Figure 9: Tracking Errors for the Second Set of Initial Conditions

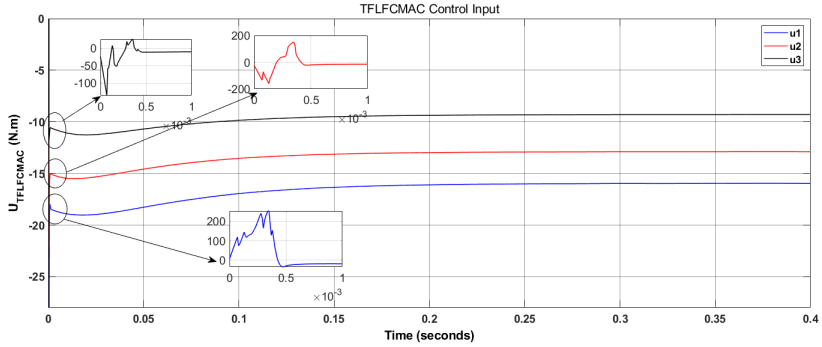


Figure 10: Generated Control TFLFCMAC Input for the Second Set of Initial Conditions

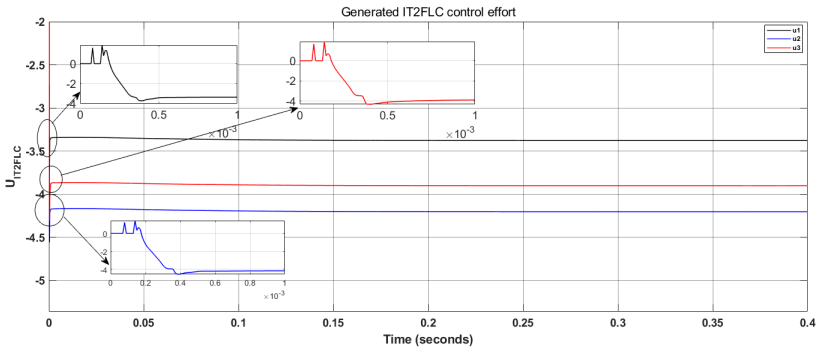


Figure 11: Generated Control IT2FLC Input for the Second Set of Initial Conditions

Control System	Total RMSE	Stabilizing Time (Sec)	Network Structure	Needs Prior Knowledge
CTC [23]	0.286	2	Offline	Yes
NNCTC [23]	0.2743	1.5	Offline	Yes
NN-FITSMC [21]	N/A	0.1	Offline	Yes
TFLFCMAC+IT2FLC	0.07243	0.2	Online	No

Table 2: A comparison between the proposed controller and previous methods

than NN-FITSMC. Overall, it outperforms the mentioned controller as it is not dependent on prior knowledge of the robot. NN-FITSMC can be achieved when the nonlinear state-space form of the system is known to the designer, while the online learning property of the proposed controller can deal with the lack of knowledge of the robot's dynamics.

Regarding transient and steady-state response criteria, table 3 indicates rise time, peak time, settling time, transient time, and steady-state error for the first, second, and third states, respectively. These values are reported for the first set of initial conditions.

Furthermore, to assess the robustness of the closed-loop system, the masses and link lengths are selected randomly from a uniform probability distribution function between 80% and 120% of their initial values. As these values are chosen randomly, the mean of the resulting tracking error and control effort signals are reported in Fig. 12 and 13. Two cost functions are introduced here to analyze the system's robustness. Two simple quadratic cost functions are defined for the control effort and tracking error (see (74) and (75)), and the test is repeated ten times for various values of random uncertainty.

$$P_e = \frac{\sum_{n=1}^{3} e_i^T e_i}{\frac{1}{N}\sum_{j=1}^{N}\sum_{n=1}^{3} e_{ij}}, \tag{74}$$

$$P_u = \sum_{n=1}^{3} u_i^T u_i, \tag{75}$$

where e_{ij} denotes the i-th element of error for the j-th test, and N is the number of tests.

Fig. 14 demonstrates how P_u changes for each repetition, which indicates that the control signals remain bounded. The standard deviation of P_e is about 0.002, while this value is significantly higher for P_u. This indicates that the control system can adapt to parameter variations in a way that keeps the tracking error close to zero, while the control input changes significantly, verifying the system's robustness as the error cost function exhibits only infinitesimal variations.

State	Rise Time(s)	Settling Time(s)	Steady-state Error(rad)
x(1)	0.0796	0.1142	0
x(2)	0.1152	0.1497	0
x(3)	0.0856	0.1202	0

Table 3: Transient and steady-state response criteria

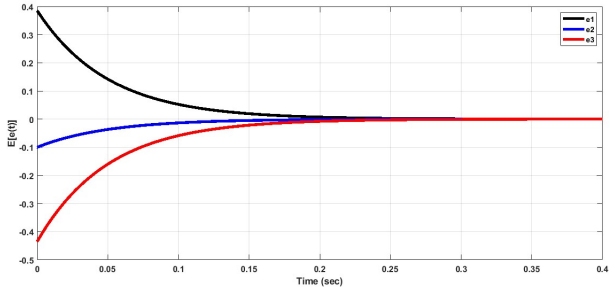


Figure 12: Mean of tracking error signal for different uncertainy values in mass and length

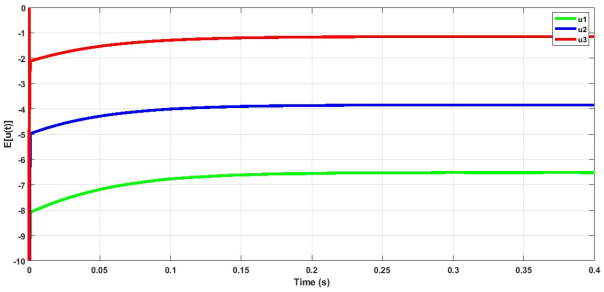


Figure 13: Mean of control input signal for different uncertainy values in mass and length

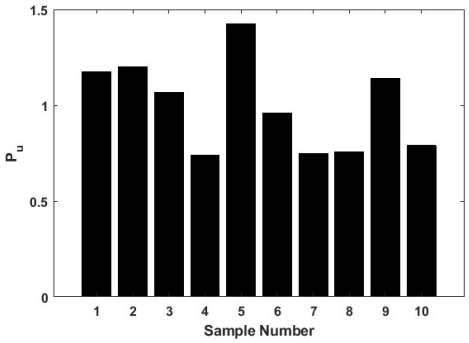


Figure 14: The value of P_u for different levels of parametric uncertainty

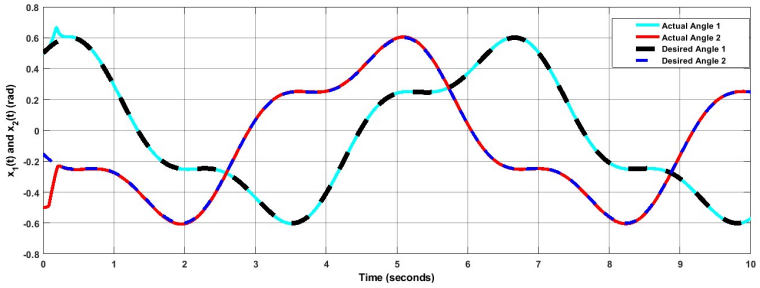


Figure 15: The Angles

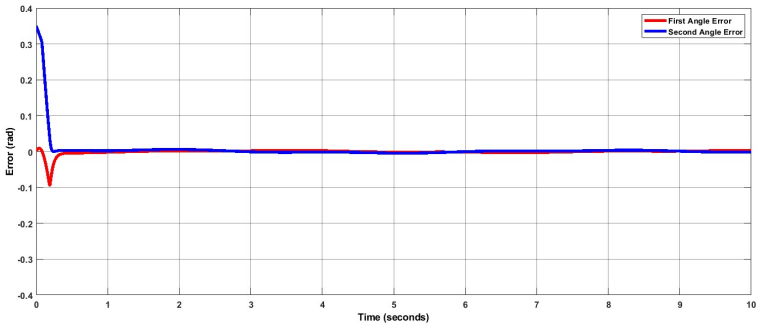


Figure 16: Tracking Error

4.2 Example 2

To further confirm the efficacy of the suggested approach, the control system is evaluated using an inverted pendulum system as a test case. The governing nonlinear equation of the inverted pendulum system is as follows:

$$\ddot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{G}(\mathbf{x})\mathbf{u}(t) + \mathbf{d}(t), \tag{76}$$

where $\mathbf{f}$, $\mathbf{G}$, and $\mathbf{d}$ are:

$$\mathbf{f}(\mathbf{x}) = \frac{g_a}{p_m L_p}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} \frac{K(a-p_m L_p)}{(mp_m Lp^2)}(-\alpha x_1 + \alpha x_2 - \delta_1 + \delta_2) \\ \frac{K(a-p_m L_p)}{(mp_m Lp^2)}(-\alpha x_2 + \alpha x_1 - \delta_2 + \delta_2) \end{bmatrix}. \tag{77}$$

$$\mathbf{G}(\mathbf{x}) = \frac{1}{mp_m L_p^2}\begin{bmatrix} \frac{1}{mp_m L_p^2} & 0 \\ 0 & \frac{1}{mp_m L_p^2} \end{bmatrix}. \tag{78}$$

$$\mathbf{d}(\mathbf{x}) = \begin{bmatrix} -\frac{m}{M}\dot{x}_1^2 \sin(x_1) \\ -\frac{m}{M}\dot{x}_2^2 \sin(x_2) \end{bmatrix}, \tag{79}$$

where x_1, x_2 are the angle and angular velocity of pendulums, $\mathbf{u} = [u_1(t), u_2(t)]^T$ are the control inputs of the pendulums. $\mathbf{d}(t)$ is the system uncertainties vector; k and g_r show the spring and the gravity constant, respectively; m_a and m_b denotes the masses of the carts and the pendulums, respectively; l is the length of the pendulums, and L is the natural length of the spring; d_1 and d_2 are the displacements of the moving carts; $\alpha = \sin(5t), c_m = \frac{m_b}{m_b+m_a}$. The numeric values of the system are as follows. $m_a = m_b = 50kg$; $k = 1$; $g_r = 1$; $l = 1$; $L = 2$, $\delta_1 = \sin(2t)$, $\delta_2 = \sin(2t) + L$; The tracking reference signals are $x_{d1} = \frac{\pi}{6.28}[\cos(t) + 0.3\sin(3t)]$ and $x_{d2} = -\frac{\pi}{6.28}[\sin(t) + 0.3\cos(3t)]$. The initial conditions of the inverted double pendulums system are $\mathbf{x} = [0.5, -0.5]^T$ and $\dot{\mathbf{x}} = [0, 0]^T$. Fig 15 - 18 illustrates the generated responses of the closed-loop system. The suggested control system generates a desirable tracking response with significantly minimized errors.

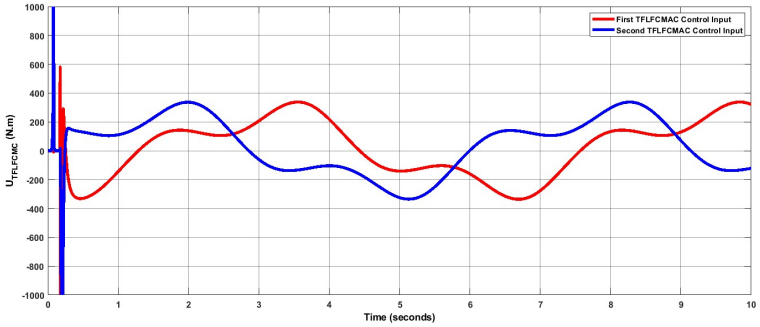


Figure 17: The TFLFCMAC control effort

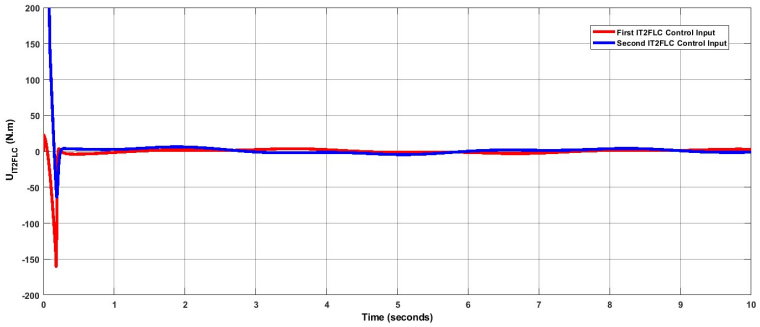


Figure 18: The IT2FLC control effort

5 Conclusions

This research has suggested a novel control structure that combines a current and highly robust control system TFLFCMC and a TSK IT2FLC. This control system mimics the ideal controller for a nonlinear MIMO system. The primary controller approximates the ideal controller; nonetheless, some estimation error remains due to different contributing factors. To remedy this, many researchers design a second controller, often called the compensator controller, which is the TSK IT2FLC in this brief. TFLFCMAC is a fast-learning and highly robust controller. It is an FCMAC that is combined with an FLN and TOPSIS layer. The FCMAC is robust due to its generalization quality. The TOPSIS layer omits unimportant rules and improves system learning speed. Furthermore, it reduces the computational burden. The FLN is added to this structure to enhance the approximation task. Both controllers contain rules that update with each time step. In the central controller, the antecedent and consequent parts of each law adapt, while in the compensation controller, only the consequent part changes. The closed-loop stability is guaranteed by establishing a Lyapunov method. Furthermore, numerical studies are conveyed on the nonlinear caterpillar robot. System trajectories reach the setpoint in less than a second. The robustness of the control system is verified by adding external disturbance and parametric uncertainties. Even though the sliding surface variable is used, the response is free of chattering phenomena.

References

[1] M. Aghaseyedabdollah, M. Abedi, M. Pourgholi, *Supervisory adaptive interval type-2 fuzzy sliding mode control for planar cable-driven parallel robots using grasshopper optimization*, Iranian Journal of Fuzzy Systems, **19**(5) (2022), 111-129.

[2] J. S. Albus, *A new approach to manipulator control: The cerebellar model articulation controller (cmac)*, Journal of Dynamic Systems, Measurement, and Control, **97**(3) (1975), 220-227.

[3] N. Amiri, V. Fakhari, S. Sepahvand, *Motion control of a caterpillar robot using optimized feedback linearization and sliding mode controllers*, International Journal of Dynamics and Control, **9**(3) (2021), 1107-1116.

[4] C. H. Chiu, *The design and implementation of a wheeled inverted pendulum using an adaptive output recurrent cerebellar model articulation controller*, IEEE Transactions on Industrial Electronics, **57**(5) (2010), 1814-1822.

[5] Z. J. Geng, C. L. McCullough, *Missile control using fuzzy cerebellar model arithmetic computer neural networks*, Journal of Guidance, Control, and Dynamics, **20**(3) (1997), 557-565.

[6] A. Ghanbari, A. Rostami, S. M. R. S. Noorani, M. M. S. Fakhrabadi, *Modeling and simulation of inchworm mode locomotion*, Intelligent Robotics and Applications, (2008), 617-624.

[7] C. F. Hsu, C. M. Chung, C. M. Lin, C. Y. Hsu, *Adaptive cmac neural control of chaotic systems with a pi-type learning algorithm*, Expert Systems with Applications, **36**(9) (2009), 11836-11843.

[8] T. T. Huynh, T. L. Le, C. M. Lin, *A topsis multi-criteria decision method-based intelligent recurrent wavelet cmac control system design for mimo uncertain nonlinear systems*, Neural Computing and Applications, **32**(8) (2020), 4025-4043.

[9] T. T. Huynh, C. M. Lin, T. L. Le, H. Y. Cho, T. T. T. Pham, N. Q. K. Le, F. Chao, *A new self-organizing fuzzy cerebellar model articulation controller for uncertain nonlinear systems using overlapped gaussian membership functions*, IEEE Transactions on Industrial Electronics, **67**(11) (2020), 9671-9682.

[10] T. T. Huynh, C. M. Lin, T. L. Le, N. P. Nguyen, S. K. Hong, F. Chao, *Wavelet interval type-2 fuzzy quad-function-link brain emotional control algorithm for the synchronization of 3d nonlinear chaotic systems*, International Journal of Fuzzy Systems, **22**(8) (2020), 2546-2564.

[11] T. L. Le, *Design of intelligent controller using type-2 fuzzy cerebellar model articulation and 3d membership functions*, International Journal of Fuzzy Systems, **25** (2022), 966-979.

[12] C. M. Lin, T. Y. Chen, *Self-organizing cmac control for a class of mimo uncertain nonlinear systems*, IEEE Transactions on Neural Networks, **20**(9) (2009), 1377-1384.

[13] C. M. Lin, T. T. Huynh, *Function-link fuzzy cerebellar model articulation controller design for nonlinear chaotic systems using topsis multiple attribute decision-making method*, International Journal of Fuzzy Systems, **20**(6) (2018), 1839-1856.

[14] C. M. Lin, V. H. La, T. L. Le, *Dc–dc converters design using a type-2 wavelet fuzzy cerebellar model articulation controller*, Neural Computing and Applications, **32**(7) (2020), 2217-2229.

[15] C. M. Lin, H. Y. Li, *Tsk fuzzy cmac-based robust adaptive backstepping control for uncertain nonlinear systems*, IEEE Transactions on Fuzzy Systems, **20**(6) (2012), 1147-1154.

[16] C. M. Lin, H. Y. Li, *Dynamic petri fuzzy cerebellar model articulation controller design for a magnetic levitation system and a two-axis linear piezoelectric ceramic motor drive system*, IEEE Transactions on Control Systems Technology, **23**(2) (2015), 693-699.

[17] C. M. Lin, H. B. Nguyen, T. T. Huynh, *A new self-organizing double function-link brain emotional learning controller for mimo nonlinear systems using sliding surface*, IEEE Access, **9** (2021), 73826-73842.

[18] J. M. Mendel, *Uncertain rule-based fuzzy systems: Introduction and new directions*, Springer Cham, 2018.

[19] V. Moosavi, A. Karami, R. Aliramaee, *High-resolution soil moisture mapping using pso-based optimized cerebellar model articulation controller (cmac)*, Science of The Total Environment, **857** (2023). DOI: 10.1016/j.scitotenv.2022.159493.

[20] J. Ozawa, I. Hayashi, N. Wakami, *Formulation of cmac-fuzzy system*, In [1992 Proceedings] IEEE International Conference on Fuzzy Systems, (1992), 1179-1186.

[21] M. Rahmani, *Control of a caterpillar robot manipulator using hybrid control*, Microsystem Technologies, **25**(7) (2019), 2841-2854.

[22] M. Rahmani, A. Ghanbari, *Computed torque control of a caterpillar robot manipulator using neural network*, Advanced Engineering Forum, **15**(3) (2016), 106-118.

[23] M. Rahmani, A. Ghanbari, M. M. Ettefagh, *Hybrid neural network fraction integral terminal sliding mode control of an inchworm robot manipulator*, Mechanical Systems and Signal Processing, **80** (2016), 117-136.

[24] M. Rahmani, A. Ghanbari, M. M. Ettefagh, *Robust adaptive control of a bio-inspired robot manipulator using bat algorithm*, Expert Systems with Applications, **56** (2016), 164-176.

[25] S. Sepahvand, M. Pourgholi, *An adaptive fuzzy cerebellar model articulation controller for synchronization of chaotic systems*, In 2019 6th International Conference on Control, Instrumentation and Automation (ICCIA), (2019), 1-6.

[26] S. F. Su, T. Tao, T. H. Hung, *Credit assigned cmac and its application to online learning robust controllers*, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), **33**(2) (2003), 202-213.

[27] C. H. Tsai, M. F. Yeh, *Application of cmac neural network to the control of induction motor drives*, Applied Soft Computing, **9**(4) (2009), 1187-1196.

[28] J. Zhao, H. Hou, Q. Y. Huang, X. G. Zhong, P. S. Zheng, *Design of optical tweezers manipulation control system based on novel self-organizing fuzzy cerebellar model neural network*, Applied Sciences, **12**(19) (2022). DOI:10.3390/app12199655.