Type-2 Fuzzy Hybrid Controller Network for Robotic Systems

Chao, Fei; Zhou, Dajun; Lin, Chih-Min; Yang, Longzhi ; Zhou, Changle ; Shang, Changjing

Published in:
IEEE Transactions on Cybernetics

DOI:
10.1109/TCYB.2019.2919128

Publication date:
2020

Citation for published version (APA):
Chao, F., Zhou, D., Lin, C.-M., Yang, L., Zhou, C., & Shang, C. (2020). Type-2 Fuzzy Hybrid Controller Network for Robotic Systems. *IEEE Transactions on Cybernetics*, *50*(8), 3778-3792. Article 8754689.
https://doi.org/10.1109/TCYB.2019.2919128

Type-2 Fuzzy Hybrid Controller Network for Robotic Systems

Fei Chao, *Member, IEEE,* Dajun Zhou, Chih-Min Lin, *Fellow, IEEE,* Longzhi Yang, *Senior Member, IEEE,* Changle Zhou, and Changjing Shang

***Abstract*—Dynamic control, including robotic control, faces both the theoretical challenge of obtaining accurate system models and the practical difficulty of defining uncertain system bounds. To facilitate such challenges, this paper proposes a control system consisting of a novel type of fuzzy neural network and a robust compensator controller. The new fuzzy neural network is implemented by integrating a number of key components embedded in a Type-2 fuzzy Cerebellar Model Articulation Controller (CMAC) and a brain emotional learning controller network (BELC), thereby mimicking an ideal sliding mode controller. The system inputs are fed into the neural network through a Type-2 fuzzy inference system (T2FIS), with the results subsequently piped into sensory and emotional channels which jointly produce the final outputs of the network. That is, the proposed network estimates the non-linear equations representing the ideal sliding mode controllers using a powerful compensator controller with the support of T2FIS and BELC, guaranteeing robust tracking of the dynamics of the controlled systems. The adaptive dynamic tuning laws of the network are developed by exploiting the popular brain emotional learning rule and the Lyapunov function. The proposed system was applied to a robot manipulator and a mobile robot, demonstrating its efficacy and potential; and a comparative study with alternatives indicates a significant improvement by the proposed system in performing intelligent dynamic control.**

***Index Terms*—Adaptive control, robot dynamic control, Type-2 inference system, brain emotion learning controller network.**

I. Introduction

DYNAMIC control of robots is required to handle complex uncertain situations [1], [2]. In particular, robot actuator dynamics, such as those of robot manipulators or driven wheels, determine the entire robot's dynamic features and system stability. Model-based adaptive control is a popular strategy to solve robot dynamic problems [3]. All model-based control systems, such as the dynamic sliding mode control (SMC) method, were developed on the establishment of precise mathematical models of the controlled systems [4]. However, the difficulties of achieving precise and accurate models often result in unsatisfactory performance of SMC controllers [5]. To address this important issue, attempts have been made to take the advantages of the learning ability of artificial neural networks (ANN) to compensate the inefficiencies of the SMC method regarding the uncertainties in building reliable mathematical models, in an effort to successfully mimic ideal SMC controllers [6].

The embrace of ANN in robot dynamic control invokes two major challenges. First, the ANN in robot controllers should ensure sufficient non-linear learning abilities, so as to effectively approximate ideal controllers using on-line learning laws. A Cerebellar Model Articulation Controller (CMAC) is able to address non-linear problems, which has been adopted in a wide variety of applications due to its rapid learning convergence and simple structure [7], [8]. Adaptive neural network controllers provide another solution, which have been applied in a number tracking control problems of mobile robots [3]. However, these studies only took the errors from the outputs of the neural network-based controller as the learning assessments for network weights updating. Yet, the overall performance of the robot should also be considered during the process of control parameter adjustment for better system performance.

Second, neural network controllers must contain sufficient adjustable parameters to deal with unexpected disturbances in the dynamics of robotic systems under uncertain environment. To enable the handling of such uncertainty, recent studies on intelligent control suggested the direct incorporation of human expertise into neural networks [7], [9]. Fuzzy inference systems have been employed as adaptive controllers for robots [10]–[14], showing one of the most successful applications of fuzzy logic systems [15]–[20]. Naturally, neural networks have been fuzzified in various ways to address the presence of uncertainty [7], [21], [22], with a number of successful applications in uncertain environments [23]. However, the limited adjustable parameters in conventional fuzzy systems restrict the degree-of-freedoms in system design and hence restrain the controller performance [24], which leads to the requirement of a more desirable and effective solution to handling complex control tasks.

The present work aims to address both challenges. The first is tackled by proposing a new type of neural network, which benefits from the adaptation of the key components of a fuzzy CMAC and a brain emotional learning controller (BELC) [25],

F. Chao, D. Zhou, C. Zhou are with the Cognitive Science Department, School of Information Science and Engineering, Xiamen University, China. e-mail: fchao@xmu.edu.cn

C.-M. Lin is with the Department of Electrical Engineering, Yuan Ze University, Chung-Li, Tao-Yuan 320, Taiwan. e-mail: cml@saturn.yzu.edu.tw.

L. Yang is with the Department of Computer and Information Sciences, Faculty of Engineering and Environment, Northumbria University, NE1 8ST, UK. e-mail: longzhi.yang@northumbria.ac.uk.

C. Shang and F. Chao are with the Department of Computer Science, Institute of Mathematics, Physics and Computer Science, Aberystwyth University, SY23 3DB, UK. e-mail: cns@aber.ac.uk.

This work was supported by the Fundamental Research Funds for the Central Universities (No. 20720190142), the National Natural Science Foundation of China (No.61673322 and 61673326), and Natural Science Foundation of Fujian Province of China (No. 2017J01129), and the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No. 663830.

Manuscript received; revised.

[26]. A typical BELC network consists of an sensory sub-system and a neural network judgment sub-system [21]. The network judgment sub-system indirectly impacts the outputs of the sensory sub-system based on the input values [27], [28]. The inputs of the two sub-systems are mapped from the network inputs by a receptive-field mechanism inspired from the CMAC network. The weights in the two sub-systems are adjusted based on a performance parameter, which is calculated from input and output pairs. After the emotional learning process, the network integrates outputs from the two sub-systems forming the final network outputs. Thanks to the interaction of the two sub-systems, such a network structure not only uses network output errors to adjust its network weights, but also benefits from using the network's emotional output as an overall performance to tune its parameters.

The second challenge is dealt with by integrating a Type-2 fuzzy inference system (T2FIS) into the proposed neural network. In contrast to conventional Type-1 fuzzy sets, Type-2 fuzzy sets provide further flexibility in handling uncertainties as they contain more adjustable parameters helping minimize the difficulty in uncertainty representation [24], [29]–[42]. As such, the employment of Type-2 fuzzy sets introduces more degrees-of-freedom into system modeling [9], [43]–[50]. Note that the inclusion of type-reduction of Type-2 fuzzy inference system also introduces extra computational burden, but recently-developed techniques are very efficient, even for general Type-2 fuzzy sets [51], [52]; the implication of such extra computational effort is thus neglectable. Indeed, many applications in robot control have been benefited from the inclusion of Type-2 fuzzy systems [53]–[56]. Through addressing both aforementioned challenges, the novel neural network proposed herein integrates a T2FIS and components from CMAC and BELC, resulting in a Type-2 Fuzzy Hybrid Controller Neural Network (T2FHC).

With the support of the proposed T2FHC, this work further develops an intelligent control system for dynamic non-linear control of robots. In particular, by combining a purpose-built compensator robust controller and the T2FHC, the resultant intelligent controller implements a system of SMC that mimics ideal SMC controllers. The intelligent controller has been applied to a robot manipulator and a mobile robot, whilst applications in other control fields can be readily identified. The simulation experimental investigations systematically evaluate the proposed techniques, with competitive results demonstrating their promising performance in dynamic robot control. The main contributions of this work are twofold: 1) a new brain emotional neural network integrating a Type-2 fuzzy inference system for great non-linear learning abilities, and 2) a neural network based robotic controller built upon a powerful compensator controller with the support of T2FIS and BELC, guaranteeing the robust tracking of the dynamics of robot systems.

II. BACKGROUND

A. Type-2 Fuzzy Cerebellar Model Articulation Controller Network

A CMAC neural network contains a quantization layer and an association weight memory layer, in addition to the relatively trivial input and output layers. Each input activates certain fields in the quantization layer, which subsequently triggers certain association neurons in the association weight memory layer. Form this, the output of CMAC is obtained by computing the weighted summation of the quantized input values. The CMAC has been fuzzified using Type-2 fuzzy sets which effectively improves the quantization scheme allowing for more accurate memory allocation [24].

In the implementation of Type-2 fuzzy CMAC, the input values are firstly fuzzified using predefined interval Type-2 fuzzy sets, which effectively builds the quantization layer of the Type-2 fuzzy CMAC network architecture. In the association memory layer, neurons are represented as the activation strengths based on the corresponding rules, each of which is computed as the aggregation of the degrees of fulfillment of upper and lower membership functions using a triangular norm (T-$norm$) operator [57]. The fuzzified quantization scheme in Type-2 CMAC can be represented as a fuzzy inference rule as defined below:

$$\text{IF } x_1 \ is \ \widetilde{F}_1 \text{ and } x_2 \ is \ \widetilde{F}_2 \text{ and } \cdots \text{ and } x_{n_i} \ is \ \widetilde{F}_{n_i}, \\ \text{THEN } Y = [w^l \ w^r],$$

where x_j $(1 \leq j \leq n_i)$ denotes an input variable; n_i denotes the input dimensionality; $\widetilde{F}$ denotes an interval Type-2 fuzzy set; Y is the output of the rule; and w^l and w^r denote the lower and upper membership degrees, respectively.

Compared to the conventional CMAC, a Type-2 fuzzy CMAC network has an extra layer to perform type reduction and defuzzification operations. The network's output can be concisely expressed by:

$$u_{T2CMAC} = \frac{1}{2}[TR^l(\underline{F}, \overline{F}, W^l) + TR^r(\underline{F}, \overline{F}, W^r)], \quad (1)$$

where $\underline{F}$ and $\overline{F}$ denote the lower and upper activation strengths of an input; W^l and W^r impose the lower and upper bounds on the activated association memory; and $TR^l(\cdot)$ and $TR^r(\cdot)$ denote the lower and upper type reduction functions. As stated above, a binary memory location in the internal memory of the conventional CMAC represents the full contribution or non-contribution of network input to each memory. However, real-valued memories are implemented in the Type-2 fuzzy CMAC networks to enable partial contributions of network inputs towards the memories, which significantly improves the system non-linear modeling ability.

B. Brain Emotional Learning Controller Network

A typical BELC network consists of an input space, a memory space, and an output space. The architecture of the memory is inspired by the functions of the amygdala and orbitofrontal cortex of mammalian brains. The amygdala memory represents a sensory network and the orbitofrontal cortex memory is an emotional network in the memory space. Computationally, the output of the amygdala-like memory, a, is defined by $a = \nu \cdot SI$, where SI denotes the network input and ν is the gain in the amygdala memory. The output of the orbitofrontal memory, o, is presented by $o = w \cdot SI$, where w denotes the gain in the orbitofrontal memory. These two memory systems influence each other to generate the overall output by simply subtracting a from o:

$$u_{BELC} = a - o = (\nu - w) \cdot SI. \tag{2}$$

The learning of the BELC is mainly performed by the sensory network, which has self-learning and adjustment parameters. The learning rule is defined by:

$$\Delta\nu = \alpha[SI \cdot \max(0, d - a)], \tag{3}$$

where α denotes the learning rate in the sensory network and d is an emotional cue. The emotional network undergoes stimulation by external factors and has an indirect impact on the sensory network. The learning rule in the emotional network is defined by:

$$\Delta w = \beta[SI \cdot (u_{BELC} - d)], \tag{4}$$

where β denotes the learning rate in the emotional network, and d is expressed by:

$$d = b \cdot SI + c \cdot u_{BELC}, \tag{5}$$

where b and c are the gain parameters, which are empirically determined in practical control systems.

Eqns. (2-5) jointly imply that the sensory network directly uses perceptions from the environment to generate control signals, and the emotional network uses the inputs and outputs of the control system to assess the performance of the controller, so as to fine adjust the output of the BELC network. The convergence of such controller is guaranteed as proven in [27].

III. Type-2 Fuzzy Hybrid Controller Neural Network

A. Network Structure

The proposed T2FHC is constructed with 6 layers, as illustrated in Fig. 1, including an input layer, a fuzzification layer, a receptive-field layer, a weight memory layer, a summarisation layer, and an output layer. The sub-structures of the input, fuzzification and receptive-field layers are inspired by the specification of a Type-2 CMAC neural network [24], and those of the rest are adopted from a BELC network. In particular, the inputs are fuzzified as Type-2 fuzzy sets by the fuzzification layer, supporting the application of fuzzy inference. The receptive-field layer calculates the activation level of fuzzy rules. The weight memory layer consists of an amygdala weight vector and an emotional weight vector which share the same inputs from the receptive-field layer. The two weight vectors are aggregated in the summarisation layer, and then delivered to the output layer for the generation of the final output.

1) Input layer X**:** The input of a T2FHC network is a continuous multi-dimensional signal. For any given n_i dimensional input signal $\boldsymbol{X} = [x_1, x_2, \cdots, x_{n_i}]^T$, each input state variable must be quantized into discrete regions according to its value space. The number of regions, n_R, is regarded as the resolution of the input layer. For example, Fig. 2 shows a T2FHC network of two dimensions, each dimension contains five regions and have the same number of partitions; thus, the resolution of the input space is $n_R = 5$.

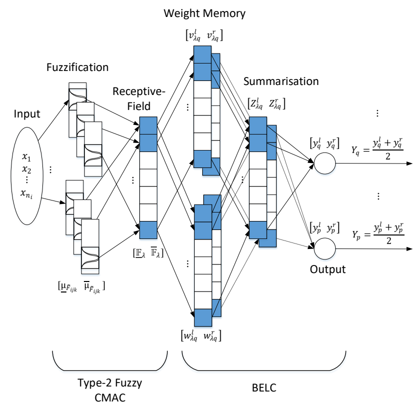


Fig. 1. Architecture of the proposed T2FHC, essentially integrating organically the key components of a Type-2 CMAC network and a BELC network.

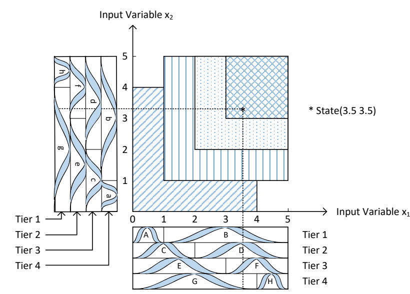


Fig. 2. Schematic diagram of two-dimensional T2FHC network operations with $n_R = 5$ and $n_T = 4$.

2) Fuzzification layer F**:** This layer executes fuzzification operations with respect to interval Type-2 fuzzy set representation. The choice of such type of fuzzy sets is based on the balance between expressiveness and computational requirement. Interval Type-2 fuzzy sets represent the fuzziness of Type-1 membership degrees as intervals, which essentially extends the uncertainty representation ability of Type-1 fuzzy sets, thus enables better handling of uncertainty, which may be brought by unexpected disturbances in the dynamic of robotic systems. In the meantime, interval Type-2 fuzzy sets require less computational power compared to general Type-2 fuzzy sets and other higher type fuzzy sets.

Each input dimension in F is evenly partitioned into a number of regions, and a certain number of regions are accumulated into a block. The number of such blocks (n_B) is usually larger than or equal to two; each block is represented as an interval Type-2 fuzzy set with Gaussian membership function used to describe the underlying Type-1 fuzzy sets. Each dimension contains n_T types of blocks, where $n_T \leq n_B$. Different types of blocks are obtained by shifting a certain block to merge with its immediate neighboring regions. Take Fig. 2 as an example, where x_1 and x_2 are the input variables and $n_B = 2$. Each dimension consists of four types of blocks (namely, $n_T = 4$), which are labeled as Tiers 1, 2, 3, and 4. For x_1, Tier 1 is divided into blocks A, and B; and for x_2, Tier 1 is divided into blocks a, and b. From Tier 1 to Tier 4, a block shifts one region each time from 1 to 4.

The underlying Type-1 Gaussian membership function within each block can be represented as follows:

$$\mu_{\tilde{F}ijk}(x_i) = T(x_i, m_{ijk}, \sigma_{ijk}) = \exp\left(-\frac{(x_i - m_{ijk})^2}{2 \cdot \sigma_{ijk}^2}\right), \tag{6}$$

where x_i denotes the ith input; $\mu_{\tilde{F}ijk}(x_i)$, m_{ijk}, and σ_{ijk} indicate the membership function, uncertain mean, and uncertain variance for the jth tier and kth block of the ith input, respectively, and m_{ijk} is within the upper bound $\overline{m}_{ijk}$ and lower bound $\underline{m}_{ijk}$ (i.e., $m_{ijk} \in [\underline{m}_{ijk}, \overline{m}_{ijk}]$). In addition, the lower and upper membership degrees ($\underline{\mu}_{\tilde{F}ijk}$ and $\overline{\mu}_{\tilde{F}ijk}$) for each input of $\mu_{\tilde{F}ijk}$ are defined as:

$$\underline{\mu}_{\tilde{F}ijk}(x_i) = \begin{cases} T(x_i, \overline{m}_{ijk}, \sigma_{ijk}), & x_i < \frac{\underline{m}_{ijk} + \overline{m}_{ijk}}{2} \\ T(x_i, \underline{m}_{ijk}, \sigma_{ijk}), & x_i > \frac{\underline{m}_{ijk} + \overline{m}_{ijk}}{2}, \end{cases} \tag{7}$$

$$\overline{\mu}_{\tilde{F}ijk}(x_i) = \begin{cases} T(x_i, \overline{m}_{ijk}, \sigma_{ijk}), & x_i < \underline{m}_{ijk} \\ 1, & \underline{m}_{ijk} < x_i < \overline{m}_{ijk} \\ T(x_i, \underline{m}_{ijk}, \sigma_{ijk}), & x_i > \overline{m}_{ijk}. \end{cases} \tag{8}$$

To summarise, each block has three adjustable parameters: 1) the upper bound of the uncertain mean $\overline{m}$, 2) the lower bound of the uncertain mean $\underline{m}$, and 3) the variance value σ of the Type-1 Gaussian membership function.

3) Receptive-field layer T: This layer consists of a batch of "receptive-fields"; each receptive-field calculates the total firing strength of its corresponding Tiers from all dimensions usually through a product calculation. For instance, in Fig. 2, the firing strength of the first receptive-field is the continuous sequence of the outputs of multiplication operations of Tier 1 of x_1 and Tier 1 of x_2. The receptive-field layer of T2FHC is formally defined as:

$$\mathbb{F}_\lambda = [\underline{\mathbb{F}}_\lambda \quad \overline{\mathbb{F}}_\lambda] = [\prod_{i=1}^{n_i} \underline{\mu}_{\tilde{F}ijk} \quad \prod_{i=1}^{n_i} \overline{\mu}_{\tilde{F}ijk}]^T, \tag{9}$$

where $\mathbb{F}_\lambda$ denotes the λth receptive-field, $\lambda \in \{1, 2, \cdots, n_T\}$. Recall that the outputs of the fuzzification layer are interval Type-2 interval sets. Therefore, the outputs of the receptive-field layer are also interval Type-2 fuzzy sets, which means the output space is bounded by its lower bound, $\underline{\mathbb{F}}_\lambda$, and upper bound, $\overline{\mathbb{F}}_\lambda$.

4) Weight memory layer W: The structure of the weight memory layer is developed from a fuzzy BELC network. This layer contains two memory spaces, including an amygdala-like memory, $\nu_{\lambda q}$, and an orbitofrontal-like memory, $w_{\lambda q}$, which simulate their counterparts in a human brain. Here, q in both $\nu_{\lambda q}$ and $w_{\lambda q}$ denotes the qth output of the T2FHC network. For simplicity, as with common approaches, in implementation, each memory is expressed as a centroid set with a unity membership grade [58]. Within this layer, each receptive-field in the preceding receptive-field T is mapped onto a corresponding weight in $\nu_{\lambda q}$ and another in $w_{\lambda q}$. In addition, each element in both $\nu_{\lambda q}$ and $w_{\lambda q}$ contains a left-most and a right-most point; that is, $\nu_{\lambda q}$ and $w_{\lambda q}$ are obtained as follows:

$$\nu_{\lambda q} = [\nu_{\lambda q}^l \quad \nu_{\lambda q}^r], \tag{10}$$

$$w_{\lambda q} = [w_{\lambda q}^l \quad w_{\lambda q}^r], \tag{11}$$

where l and r indicate the left-most and right-most point of the centroid set for $\nu_{\lambda q}$ or $w_{\lambda q}$, respectively.

By adapting the updating rules of the BELC as specified in Eqns. (3) and (4), whilst ensuring that the control system implements the desirable backstepping control technology [59], the updating rules of $\nu_{\lambda q}$ and $w_{\lambda q}$ are introduced in the derivative form as:

$$\dot{\nu}_{\lambda q} = \alpha[\mathbb{F}_\lambda \cdot (\max[0, d_q - a_q])], \tag{12}$$

$$\dot{w}_{\lambda q} = \beta[\mathbb{F}_\lambda \cdot (u_{T2FHC_q} - d_q)], \tag{13}$$

where α and β are learning rates of the updating rules; a_q denotes the $\nu_{\lambda q}$'s output and u_{T2FHC_q} denotes the output of $w_{\lambda q}$; and d_p is an emotional parameter given by:

$$d_q = b_i \cdot x_i + c_q \cdot u_{T2FHC_q}, \tag{14}$$

where, b_i and c_q are gain parameters. Note that the learning objective of T2FHC is to obtaining the minimum value of d_q, which is the sum of the input of the T2FHC and the qth output.

Presentation-wise, ν_λ and w_λ share the same implementation structure that can be expressed as:

$$\boldsymbol{W} = \begin{bmatrix} \mathrm{w}_{11} & \cdots & \mathrm{w}_{1o} & \cdots & \mathrm{w}_{1p} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \mathrm{w}_{k1} & \cdots & \mathrm{w}_{ko} & \cdots & \mathrm{w}_{kp} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \mathrm{w}_{n_T 1} & \cdots & \mathrm{w}_{n_T o} & \cdots & \mathrm{w}_{n_T p} \end{bmatrix}, \tag{15}$$

where p is the dimensionality of the network's output.

5) Summarisation layer S: The summarisation layer summarises the values of the two spaces and reduces the fuzzy type. In the Type-1 fuzzy BELC network, the output of the amygdala-like memory is defined as $a_q = \sum_{\lambda=1}^{n_\lambda} f_{\lambda q} \nu_{\lambda q}$; and that of the orbitofrontal-like memory as $o_q = \sum_{\lambda=1}^{n_\lambda} f_{\lambda q} w_{\lambda q}$ [5]. The T2FHC herein generalises this, and the output of the summarisation layer therefore is:

$$S_{net} = a_q - o_q = \sum_{\lambda=1}^{n_\lambda} \mathbb{F}_{\lambda q}(\nu_{\lambda q} - w_{\lambda q}) = \sum_{\lambda=1}^{n_\lambda} \mathbb{F}_{\lambda q} Z_{\lambda q}, \tag{16}$$

where $Z_{\lambda q}$ is the summarised weight, which is defined by:

$$Z_{\lambda q} = [Z_{\lambda q}^l \quad Z_{\lambda q}^r]^T = [(\nu_{\lambda q}^l - w_{\lambda q}^l) \quad (\nu_{\lambda q}^r - w_{\lambda q}^r)]^T. \tag{17}$$

The type-reduction method as reported in [24] is applied here to convert interval Type-2 fuzzy sets into Type-1 ones (although any established reduction method available in the literature may be adapted as an alternative for this):

$$y_q^l = \frac{\sum_{\lambda=1}^{\mathcal{L}} \overline{\mathbb{F}}_\lambda Z_{\lambda q}^l + \sum_{\lambda=\mathcal{L}+1}^{n_\lambda} \underline{\mathbb{F}}_\lambda Z_{\lambda q}^l}{\sum_{\lambda=1}^{\mathcal{L}} \overline{\mathbb{F}}_\lambda + \sum_{\lambda=\mathcal{L}+1}^{n_\lambda} \underline{\mathbb{F}}_\lambda} \tag{18}$$

$$y_q^r = \frac{\sum_{\lambda=1}^{\mathcal{R}} \overline{\mathbb{F}}_\lambda Z_{\lambda q}^r + \sum_{\lambda=\mathcal{R}+1}^{n_\lambda} \underline{\mathbb{F}}_\lambda Z_{\lambda q}^r}{\sum_{\lambda=1}^{\mathcal{R}} \overline{\mathbb{F}}_\lambda + \sum_{\lambda=\mathcal{R}+1}^{n_\lambda} \underline{\mathbb{F}}_\lambda}, \tag{19}$$

where $Z_{\lambda q} = [Z_{\lambda q}^l \quad Z_{\lambda q}^r]^T = [Z_{1q}^l, Z_{2q}^l, \cdots, Z_{n_\lambda q}^l \quad Z_{1q}^r, Z_{2q}^r, \cdots, Z_{n_\lambda q}^r]^T$; and $\mathcal{L}$ and $\mathcal{R}$ indicate the left-most and right-most points of the summarisation layer. Details regarding the computation of $\mathcal{L}$ and $\mathcal{R}$ are beyond the scope of this paper but can be found in [24].

6) Output layer Y: This is a trivial but necessary final layer within the proposed network. It performs defuzzification operation to produce crisp outputs. In implementation, the q^{th} output is simply computed by:

$$Y_q = \frac{y_q^l + y_q^r}{2}, \tag{20}$$

which completes the entire computation process of a T2FHC.

B. Rule Updating

As described above, a T2FHC contains seven tunable parameters, which are $\underline{m}_{i\lambda}$, $\overline{m}_{i\lambda}$, $\sigma_{i\lambda}$, $\nu^l_{\lambda q}$, $\nu^r_{\lambda q}$, $w^l_{\lambda q}$, and $w^r_{\lambda q}$. Based on the gradient descent method, the updating rules of these parameters can be devised as summarized below:

$$\underline{m}_{i\lambda}(k+1) = \underline{m}_{i\lambda}(k) + \dot{\underline{\hat{m}}}_{i\lambda}, \tag{21}$$
$$\overline{m}_{i\lambda}(k+1) = \overline{m}_{i\lambda}(k) + \dot{\overline{\hat{m}}}_{i\lambda}, \tag{22}$$
$$\sigma_{i\lambda}(k+1) = \sigma_{i\lambda}(k) + \dot{\hat{\sigma}}^l_{i\lambda} + \dot{\hat{\sigma}}^r_{i\lambda}, \tag{23}$$
$$\nu^l_{\lambda q}(k+1) = \nu^l_{\lambda q}(k) + \dot{\nu}^l_{\lambda q}, \tag{24}$$
$$\nu^r_{\lambda q}(k+1) = \nu^r_{\lambda q}(k) + \dot{\nu}^r_{\lambda q}, \tag{25}$$
$$w^l_{\lambda q}(k+1) = w^l_{\lambda q}(k) + \dot{w}^l_{\lambda q}, \tag{26}$$
$$w^r_{\lambda q}(k+1) = w^r_{\lambda q}(k) + \dot{w}^r_{\lambda q}, \tag{27}$$

where $\dot{\underline{\hat{m}}}_{i\lambda}$ and $\dot{\overline{\hat{m}}}_{i\lambda}$ denote the adjustments of the lower and upper bound of $m_{i\lambda}$; $\dot{\hat{\sigma}}^l_{i\lambda}$ and $\dot{\hat{\sigma}}^r_{i\lambda}$ denote the adjustments of $\sigma_{i\lambda}$ from Z^l_λ and Z^r_λ; and ($\dot{\nu}^l_{\lambda q}$, $\dot{\nu}^r_{\lambda q}$), and ($\dot{w}^l_{\lambda q}$, $\dot{w}^r_{\lambda q}$) indicate the left and right bound weight adjustments of $\nu_{\lambda q}$ and $w_{\lambda q}$, respectively.

For parameters $\dot{\underline{\hat{m}}}_{i\lambda}$, $\dot{\overline{\hat{m}}}_{i\lambda}$, $\dot{\hat{\sigma}}^l_{i\lambda}$ and $\dot{\hat{\sigma}}^r_{i\lambda}$, $\mathcal{L}$ and $\mathcal{R}$ determine the left-most and right-most positions of the Summarisation layer, and the right-most position is generally not smaller than the left-most one (i.e., $\mathcal{R} \geq \mathcal{L}$). Therefore, the adjustments of these include three different situations: 1) $\lambda \leq \mathcal{L}$, 2) $\mathcal{L} < \lambda \leq \mathcal{R}$, and 3) $\lambda > \mathcal{R}$. Similarly, for the four weights $\nu^l_{\lambda q}$, $\nu^r_{\lambda q}$, $w^l_{\lambda q}$, and $w^r_{\lambda q}$, whilst their adjustment is based on the update rule of BELC as defined in Eqns. (12-14), the adjusting method for the λth left or right bound weights is determined by the output of the λth receptive-field $\mathbb{F}_\lambda$. As such, Eqns. (12) and (13) must be rewritten on the basis of $\mathbb{F}_\lambda$ and therefore, the computation must also be divided into the three situations.

Algorithm 1 The T2FHC network

1: Normalize each dimension (x_i) of $\boldsymbol{X}$ from 0 to n_R;
2: Compute $\mathbb{F}_\lambda$ using Eqns. (7) to (9);
3: Calculate $Z_{\lambda q}$ in Eqn. (17), and then y^l_q and y^r_q in Eqns. (18) and (19);
4: Derive the output Y_q of the network by Eqn. (20);
5: Update $\underline{m}_{i\lambda}$, $\overline{m}_{i\lambda}$, $\sigma_{i\lambda}$, $\nu^l_{\lambda q}$, $\nu^r_{\lambda q}$, $w^l_{\lambda q}$, and $w^r_{\lambda q}$ using the updating rules from Eqn. (21) to Eqn. (55).

For conciseness, the adjustments in the three situations are summarised as follows:

Situation 1: $\lambda \leq \mathcal{L}$

$$\dot{\underline{\hat{m}}}_{i\lambda} = \eta^m \cdot \underline{\mathbb{F}}^m_{i\lambda} \cdot \hat{Z}^r_\lambda \cdot s(\underline{e}(t)) \tag{28}$$
$$\dot{\overline{\hat{m}}}_{i\lambda} = \eta^m \cdot \overline{\mathbb{F}}^m_{i\lambda} \cdot \hat{Z}^l_\lambda \cdot s(\underline{e}(t)) \tag{29}$$
$$\dot{\hat{\sigma}}^l_{i\lambda} = \eta^\sigma \cdot \overline{\mathbb{F}}^\sigma_{i\lambda} \cdot \hat{Z}^l_\lambda \cdot s(\underline{e}(t)) \tag{30}$$
$$\dot{\hat{\sigma}}^r_{i\lambda} = \eta^\sigma \cdot \underline{\mathbb{F}}^\sigma_{i\lambda} \cdot \hat{Z}^r_\lambda \cdot s(\underline{e}(t)) \tag{31}$$
$$\mathbb{F}^l_\lambda = \frac{\overline{\mathbb{F}}_\lambda}{\sum_{\lambda=1}^{L} \overline{\mathbb{F}}_\lambda + \sum_{\lambda=L+1}^{nL} \underline{\mathbb{F}}_\lambda} \tag{32}$$
$$\mathbb{F}^r_\lambda = \frac{\underline{\mathbb{F}}_\lambda}{\sum_{\lambda=1}^{R} \overline{\mathbb{F}}_\lambda + \sum_{\lambda=R+1}^{nR} \underline{\mathbb{F}}_\lambda} \tag{33}$$
$$\dot{\nu}^l_{\lambda q} = \alpha[\mathbb{F}^l_\lambda \cdot (\max[0, d_q - a_q])] \tag{34}$$
$$\dot{w}^l_{\lambda q} = \beta[\mathbb{F}^l_\lambda \cdot (u_{T2FHC_q} - d_q)] \tag{35}$$
$$\dot{\nu}^r_{\lambda q} = \alpha[\mathbb{F}^r_\lambda \cdot (\max[0, d_q - a_q])] \tag{36}$$
$$\dot{w}^r_{\lambda q} = \beta[\mathbb{F}^r_\lambda \cdot (u_{T2FHC_q} - d_q)] \tag{37}$$

Situation 2: $\mathcal{L} < \lambda \leq \mathcal{R}$

$$\dot{\underline{\hat{m}}}_{i\lambda} = \eta^m \cdot \underline{\mathbb{F}}^m_{i\lambda} \cdot \frac{\hat{Z}^l_\lambda + \hat{Z}^r_\lambda}{2} \cdot s(\underline{e}(t)) \tag{38}$$
$$\dot{\overline{\hat{m}}}_{i\lambda} = 0 \tag{39}$$
$$\dot{\hat{\sigma}}^l_{i\lambda} = \eta^\sigma \cdot \underline{\mathbb{F}}^\sigma_{i\lambda} \cdot \hat{Z}^l_\lambda \cdot s(\underline{e}(t)) \tag{40}$$
$$\dot{\hat{\sigma}}^r_{i\lambda} = \eta^\sigma \cdot \underline{\mathbb{F}}^\sigma_{i\lambda} \cdot \hat{Z}^r_\lambda \cdot s(\underline{e}(t)) \tag{41}$$
$$\mathbb{F}^l_\lambda = \frac{\underline{\mathbb{F}}_\lambda}{\sum_{\lambda=1}^{L} \overline{\mathbb{F}}_\lambda + \sum_{\lambda=L+1}^{nL} \underline{\mathbb{F}}_\lambda} \tag{42}$$
$$\mathbb{F}^r_\lambda = \frac{\underline{\mathbb{F}}_\lambda}{\sum_{\lambda=1}^{R} \overline{\mathbb{F}}_\lambda + \sum_{\lambda=R+1}^{nR} \underline{\mathbb{F}}_\lambda} \tag{43}$$
$$\dot{\nu}^l_{\lambda q} = \alpha[\mathbb{F}^l_\lambda \cdot (\max[0, d_q - a_q])] \tag{44}$$
$$\dot{w}^l_{\lambda q} = \beta[\mathbb{F}^l_\lambda \cdot (u_{T2FHC_q} - d_q)] \tag{45}$$
$$\dot{\nu}^r_{\lambda q} = \alpha[\mathbb{F}^r_\lambda \cdot (\max[0, d_q - a_q])] \tag{46}$$
$$\dot{w}^r_{\lambda q} = \beta[\mathbb{F}^r_\lambda \cdot (u_{T2FHC_q} - d_q)] \tag{47}$$

Situation 3: $\lambda > \mathcal{R}$

$$\dot{\underline{\hat{m}}}_{i\lambda} = \eta^m \cdot \underline{\mathbb{F}}^m_{i\lambda} \cdot \hat{Z}^l_\lambda \cdot s(\underline{e}(t)) \tag{48}$$
$$\dot{\overline{\hat{m}}}_{i\lambda} = \eta^m \cdot \overline{\mathbb{F}}^m_{i\lambda} \cdot \hat{Z}^r_\lambda \cdot s(\underline{e}(t)) \tag{49}$$
$$\dot{\hat{\sigma}}^l_{i\lambda} = \eta^\sigma \cdot \underline{\mathbb{F}}^\sigma_{i\lambda} \cdot \hat{Z}^l_\lambda \cdot s(\underline{e}(t)) \tag{50}$$
$$\dot{\hat{\sigma}}^r_{i\lambda} = \eta^\sigma \cdot \overline{\mathbb{F}}^\sigma_{i\lambda} \cdot \hat{Z}^r_\lambda \cdot s(\underline{e}(t)) \tag{51}$$
$$\mathbb{F}^l_\lambda = \frac{\underline{\mathbb{F}}_\lambda}{\sum_{\lambda=1}^{L} \overline{\mathbb{F}}_\lambda + \sum_{\lambda=L+1}^{nL} \underline{\mathbb{F}}_\lambda} \tag{52}$$
$$\mathbb{F}^r_\lambda = \frac{\overline{\mathbb{F}}_\lambda}{\sum_{\lambda=1}^{R} \overline{\mathbb{F}}_\lambda + \sum_{\lambda=R+1}^{nR} \underline{\mathbb{F}}_\lambda} \tag{53}$$
$$\dot{\nu}^l_{\lambda q} = \alpha[\mathbb{F}^l_\lambda \cdot (\max[0, d_q - a_q])] \tag{54}$$
$$\dot{w}^l_{\lambda q} = \beta[\mathbb{F}^l_\lambda \cdot (u_{T2FHC_q} - d_q)] \tag{55}$$
$$\dot{\nu}^r_{\lambda q} = \alpha[\mathbb{F}^r_\lambda \cdot (\max[0, d_q - a_q])] \dot{w}^r_{\lambda q} = \beta[\mathbb{F}^r_\lambda \cdot (u_{T2FHC_q} - d_q)] \tag{56}$$

The working procedure of the proposed T2FHC network is summarised in Algorithm 1. The computational complexity of the algorithm depends on the number of inputs (n_i), the number of block types (n_T), and the number of outputs (n_o). The values of n_i and n_o are determined once the controlled system is specified. In Algorithm 1, the computational complexity of $\mathbb{F}_\lambda$ is $O(n_i * n_T)$; the computations of y_q^l and y_q^r depend on the Karnik-Mendel Algorithms [24], and the Karnik-Mendel algorithms are proven to be of super-exponential convergence based on the work of [60], which is therefore approximated as $O(KMT(n_T))$. The computational complexity of the proposed T2FHC network can then be summarised as $O(n_i * n_T + 2 * KMT(n_T) * n_o)$.

IV. Fuzzy Sliding Mode Control Using T2FHC

The novel signal processing neural network T2FHC presented in Section III is utilised herein to form a new controller for non-linear control problems. The structure of the proposed controller is illustrated in Fig. 3, which takes the errors of a non-linear system that need to be minimized as inputs, and produces acceptable control values as system outputs. The controller is comprised of three interconnected sub-systems, including a sliding surface generator, a T2FHC, and a baseline robust controller. The input error values are first processed to form a sliding surface, which is then fed into the other two sub-systems for control signal generation. The control signals generated from both controllers are then aggregated to produce the final output of the overall control system.

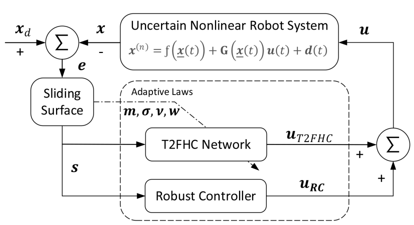


Fig. 3. T2FHC-based robust control for uncertain non-linear robotic systems.

Without losing generality, suppose that the state vector of an nth order uncertain non-linear robotic system is expressed as:

$$\boldsymbol{x}^{(n)}(t) = \boldsymbol{f}(\underline{\boldsymbol{x}}(t)) + \boldsymbol{G}(\underline{\boldsymbol{x}}(t))\boldsymbol{u}(t) + \boldsymbol{d}(t), \tag{57}$$

where $\boldsymbol{x}(t) = [x_1(t)\ x_2(t)\ \cdots\ x_\kappa(t)]^T \in R^\kappa$ denotes the output of the system; $\underline{\boldsymbol{x}}(t) = [\boldsymbol{x}^T(t)\ \dot{\boldsymbol{x}}^T(t)\ \cdots\ \boldsymbol{x}^{(n-1)T}(t)]$ denotes the state vector of the system; κ denotes the dimensionality of the input or output of the system which are the same in this particular application; $\boldsymbol{f}(\underline{\boldsymbol{x}}(t)) \in R^\kappa$ denotes an unknown, but bounded non-linear function; $\boldsymbol{G}(\underline{\boldsymbol{x}}(t)) \in R^\kappa$ denotes an unknown, but bounded control input gain matrix $\boldsymbol{G}(\underline{\boldsymbol{x}}(t)) \in R^{\kappa\times\kappa}$; $\boldsymbol{d}(t) \in R^\kappa$ indicates the disturbance: $\boldsymbol{d}(t) = [d_1(t)\ d_2(t)\ \cdots, d_\kappa(t)]^T \in R^\kappa$; $\boldsymbol{u}(t) = [u_1(t)\ u_2(t)\ \cdots\ u_\kappa(t)]^T \in R^\kappa$ denotes the output of the sliding mode controller.

The objective of the (overall) controller is to enable the system trajectory $\boldsymbol{x}(t)$ to match a desired reference trajectory $\boldsymbol{x}_d(t) \in R^\kappa$. To reflect this, the tracking error $\boldsymbol{e}(t) \in R^\kappa$ is defined as: $\boldsymbol{e}(t) = \boldsymbol{x}_d(t) - \boldsymbol{x}(t)$. By considering the state vector of the system $\underline{\boldsymbol{x}}(t)$, the tracking error vector, $\underline{\boldsymbol{e}}(t)$, of the system can therefore be defined as:

$$\underline{\boldsymbol{e}}(t) = [\boldsymbol{e}^T\ \dot{\boldsymbol{e}}^T\ \cdots\ \boldsymbol{e}^{(n-1)T}]^T. \tag{58}$$

In the proposed approach, as shown in Fig. 3, a sliding surface is defined by:

$$\boldsymbol{s}(\underline{\boldsymbol{e}}(t)) = \boldsymbol{e}^{(n-1)}(t) + \boldsymbol{\zeta_1}\boldsymbol{e}^{(n-2)}(t) + \cdots + \boldsymbol{\zeta}_{n-1}\boldsymbol{e}(t) + \boldsymbol{\zeta}_n\int_0^t \boldsymbol{e}(t)dt, \tag{59}$$

where $\boldsymbol{s} = [s_1\ s_2\ \cdots\ s_\kappa]^T$, $\boldsymbol{\zeta}_i = diag(\zeta_{i1}, \zeta_{i2}, \cdots, \zeta_{i\kappa})$, $i = 1, 2, \cdots, n$, with each element in ζ_{ij} being a positive constant. In particular, $\boldsymbol{\zeta}_i$ is defined to ensure the satisfaction of the Hurwitz characteristic polynomial. Differentiating $\boldsymbol{s}(\underline{\boldsymbol{e}}(t))$ with respect to time leads to:

$$\begin{aligned}\dot{\boldsymbol{s}}(\underline{\boldsymbol{e}}(t)) &= \boldsymbol{e}^{(n)}(t) + \boldsymbol{\zeta}_1\boldsymbol{e}^{(n-1)}(t) + \cdots + \boldsymbol{\zeta}_n\boldsymbol{e}(t)\\ &= \mathbf{C}^T\dot{\underline{\boldsymbol{e}}}(t) + \mathbf{K}^T\underline{\boldsymbol{e}}(t),\end{aligned} \tag{60}$$

where $\mathbf{C} = [\mathbf{0}\ \mathbf{0}\ \cdots\ \boldsymbol{I}]^T$, and $\mathbf{K} = [\boldsymbol{\zeta}_n\ \boldsymbol{\zeta_{n-1}}\ \cdots\ \boldsymbol{\zeta}_1]^T$ denotes the feedback gain matrix. Note that the output of the sliding mode controller is obtained by aggregating the outputs of both T2FHC ($\boldsymbol{u}_{T2FHC}$) and the baseline robust controller ($\boldsymbol{u}_{RC}$) such that:

$$\boldsymbol{u} = \boldsymbol{u}_{T2FHC} + \boldsymbol{u}_{RC}. \tag{61}$$

Using the nominal function and constant gain, Eqn. (57) can be re-expressed as:

$$\boldsymbol{x}^{(n)}(t) = \boldsymbol{f}_n(\underline{\boldsymbol{x}}(t)) + \mathbf{G}_n\boldsymbol{u}(t) + \boldsymbol{l}(\underline{\boldsymbol{x}}(t), t), \tag{62}$$

where $\boldsymbol{f}_n(\underline{\boldsymbol{x}}(t))$ denotes the nominal version of $\boldsymbol{f}(\underline{\boldsymbol{x}}(t))$; $\mathbf{G}_n$ indicates the nominal constant gain of $\boldsymbol{G}(\underline{\boldsymbol{x}}(t))$ which must be positive and invertible; and $\boldsymbol{l}(\underline{\boldsymbol{x}}(t))$ represents the lumped uncertainty in the model.

If there exists an ideal situation where $\boldsymbol{f}_n(\underline{\boldsymbol{x}}(t))$, $\mathbf{G}_n$, and $\boldsymbol{l}(\underline{\boldsymbol{x}}(t))$ are known, an ideal controller can be obtained by:

$$\boldsymbol{u}_{ISM} = \mathbf{G}_n^{-1}[\boldsymbol{x}_d^{(n)} - \boldsymbol{f}_n(\underline{\boldsymbol{x}}) - \boldsymbol{l}(\underline{\boldsymbol{x}}, t) + \mathbf{K}^T\underline{\boldsymbol{e}} + \varrho sgn[\boldsymbol{s}(\underline{\boldsymbol{e}}(t))]], \tag{63}$$

where $\varrho sgn[\boldsymbol{s}(\underline{\boldsymbol{e}}(t))]$ denotes the constant reaching law of the sliding mode controller, $\varrho > 0$; and $\boldsymbol{s}$ denotes the system error $\boldsymbol{e}$ processed by the sliding surface. It follows that:

$$\dot{\boldsymbol{s}}(\underline{\boldsymbol{e}}(t)) = \mathbf{G}_n[\boldsymbol{u}_{ISM} - \boldsymbol{u}] - \varrho sgn[\boldsymbol{s}(\underline{\boldsymbol{e}}(t))]. \tag{64}$$

Suppose that an optimal T2FHC neural network, $\boldsymbol{u}^*_{T2FHC}$, is known to learn the ideal sliding mode controller, $\boldsymbol{u}_{ISM}$. In this case, $\boldsymbol{u}_{ISM}$ should then be:

$$\boldsymbol{u}_{ISM} = \boldsymbol{u}^*_{T2FHC}(\boldsymbol{X}, \boldsymbol{Z}^*, \boldsymbol{m}^*, \boldsymbol{\sigma}^*) + \boldsymbol{\varepsilon} = \boldsymbol{Z}^{*T}\mathbb{F}^* + \boldsymbol{\varepsilon}, \tag{65}$$

where $\boldsymbol{Z}^*$, $\boldsymbol{m}^*$, $\boldsymbol{\sigma}^*$, and $\mathbb{F}^*$ are the optimal parameters of $\boldsymbol{Z}$, $\boldsymbol{m}$, $\boldsymbol{\sigma}$, and $\mathbb{F}$, respectively; $\mathbb{F}$ is defined in Eqn. (9); and $\boldsymbol{\varepsilon}$ denotes a minimum reconstructed error vector.

Unfortunately, as indicated previously, such an ideal control network can hardly be obtained. The alternative approach proposed herein is to approximate the optimal T2FHC. For this purpose, Eqn. (61) can be rewritten as:

$$\boldsymbol{u} = \hat{\boldsymbol{u}}_{T2FHC}(\boldsymbol{X}, \hat{\boldsymbol{Z}}, \hat{\boldsymbol{m}}, \hat{\boldsymbol{\sigma}}) + \boldsymbol{u}_{RC} = \hat{\boldsymbol{Z}}^T\hat{\mathbb{F}} + \boldsymbol{u}_{RC}. \tag{66}$$

Then, by substituting (64) with (65) and (66), the following can be derived:

$$\begin{aligned}\dot{\boldsymbol{s}}(\underline{\boldsymbol{e}}(t)) &= \mathbf{G}_n[\boldsymbol{u}^*_{T2FHC} + \boldsymbol{\varepsilon} - \hat{\boldsymbol{u}}_{T2FHC} - \boldsymbol{u}_{RC}] - \varrho sgn[\boldsymbol{s}(\underline{\boldsymbol{e}}(t))]\\ &= \mathbf{G}_n[\boldsymbol{Z}^{*T}\mathbb{F}^* - \hat{\boldsymbol{Z}}^T\hat{\mathbb{F}} + \boldsymbol{\varepsilon} - \boldsymbol{u}_{RC}] - \varrho sgn[\boldsymbol{s}(\underline{\boldsymbol{e}}(t))]\\ &= \mathbf{G}_n[\tilde{\boldsymbol{Z}}^T\mathbb{F}^* + \hat{\boldsymbol{Z}}^T\tilde{\mathbb{F}} + \boldsymbol{\varepsilon} - \boldsymbol{u}_{RC}] - \varrho sgn[\boldsymbol{s}(\underline{\boldsymbol{e}}(t))],\end{aligned} \tag{67}$$

where $\tilde{\boldsymbol{Z}} = \boldsymbol{Z}^* - \hat{\boldsymbol{Z}}$ and $\tilde{\mathbb{F}} = \mathbb{F}^* - \hat{\mathbb{F}}$. Hence, according to the T2FHC's structure, the following holds:

$$\begin{aligned}\tilde{\boldsymbol{u}}_{T2FHC} &= \boldsymbol{u}^*_{T2FHC} - \hat{\boldsymbol{u}}_{T2FHC} = \boldsymbol{Z}^{*T}\mathbb{F}^* - \hat{\boldsymbol{Z}}^T\mathbb{F}^* \\ &= (\boldsymbol{Z}^{*T} - \hat{\boldsymbol{Z}}^T)\mathbb{F}^*.\end{aligned} \tag{68}$$

Recall that Eqn. (14) expresses the emotional parameter in the learning rules. Thus, both $\boldsymbol{Z}^*$ and $\hat{\boldsymbol{Z}}$ need to be computed in a way to minimize the system error $\boldsymbol{s}(\underline{\boldsymbol{e}}(t))$ and hence, the corresponding network output. This implies that a bounded extreme small real number γ exists such that:

$$\lim |\tilde{u}_{T2FHC_q}| = \lim |(Z^{*T}_q - \hat{Z}^T_q)\mathbb{F}^*| = |\gamma_q|. \tag{69}$$

In this study, the Taylor linearization method is used to expand the receptive-field membership functions into partially linear ones. Thus, $\tilde{\mathbb{F}}$ can be obtained as follows:

$$\begin{aligned}\tilde{\mathbb{F}} &= \begin{bmatrix}\tilde{\mathbb{F}}_1 \\ \vdots \\ \tilde{\mathbb{F}}_\lambda \\ \vdots \\ \tilde{\mathbb{F}}_{n_\lambda}\end{bmatrix} = \begin{bmatrix}(\frac{\partial \mathbb{F}_1}{\partial \boldsymbol{m}})^T \\ \vdots \\ (\frac{\partial \mathbb{F}_\lambda}{\partial \boldsymbol{m}})^T \\ \vdots \\ (\frac{\partial \mathbb{F}_{n_\lambda}}{\partial \boldsymbol{m}})^T\end{bmatrix} |_{m=\hat{m}}(\boldsymbol{m}^* - \hat{\boldsymbol{m}}) \\ &+ \begin{bmatrix}(\frac{\partial \mathbb{F}_1}{\partial \boldsymbol{\sigma}})^T \\ \vdots \\ (\frac{\partial \mathbb{F}_\lambda}{\partial \boldsymbol{\sigma}})^T \\ \vdots \\ (\frac{\partial \mathbb{F}_{n_\lambda}}{\partial \boldsymbol{\sigma}})^T\end{bmatrix} |_{\sigma=\hat{\sigma}}(\boldsymbol{\sigma}^* - \hat{\boldsymbol{\sigma}}) + \boldsymbol{\beta} \\ &\equiv \boldsymbol{f}_m\tilde{\boldsymbol{m}} + \boldsymbol{f}_\sigma\tilde{\boldsymbol{\sigma}} + \boldsymbol{\beta},\end{aligned} \tag{70}$$

where $\boldsymbol{\beta}$ is a vector of higher order terms; and $\frac{\partial \mathbb{F}_\lambda}{\partial \boldsymbol{m}}$ and $\frac{\partial \mathbb{F}_\lambda}{\partial \boldsymbol{\sigma}}$ are defined as follows:

$$[\frac{\partial \mathbb{F}_\lambda}{\partial \boldsymbol{m}}] = [\underbrace{0, \cdots, 0}_{(\lambda-1)n_i}, \frac{\partial \mathbb{F}_\lambda}{\partial m_{1\lambda}}, \cdots, \frac{\partial \mathbb{F}_\lambda}{\partial m_{n_i\lambda}}, \underbrace{0, \cdots, 0}_{(n-\lambda)n_i}], \tag{71}$$

$$[\frac{\partial \mathbb{F}_\lambda}{\partial \boldsymbol{\sigma}}] = [\underbrace{0, \cdots, 0}_{(\lambda-1)n_i}, \frac{\partial \mathbb{F}_\lambda}{\partial \sigma_{1\lambda}}, \cdots, \frac{\partial \mathbb{F}_\lambda}{\partial \sigma_{n_i\lambda}}, \underbrace{0, \cdots, 0}_{(n-\lambda)n_i}]. \tag{72}$$

Then, substituting Eqns. (70) and (69) by Eqn. (67), the following can be generated:

$$\begin{aligned}&\dot{\boldsymbol{s}}(\underline{\boldsymbol{e}}(t)) \\ &= \mathbf{G}_n[\gamma + \hat{\boldsymbol{Z}}^T(\boldsymbol{f}_m\tilde{\boldsymbol{m}} + \boldsymbol{f}_\sigma\tilde{\boldsymbol{\sigma}} + \boldsymbol{\beta}) + \boldsymbol{\varepsilon} - \boldsymbol{u}_{RC}] - \varrho sgn[\boldsymbol{s}(\underline{\boldsymbol{e}}(t))] \\ &= \mathbf{G}_n[\hat{\boldsymbol{Z}}^T(\boldsymbol{f}_m\tilde{\boldsymbol{m}} + \boldsymbol{f}_\sigma\tilde{\boldsymbol{\sigma}}) + \hat{\boldsymbol{Z}}^T\boldsymbol{\beta} + \boldsymbol{\varepsilon} + \boldsymbol{\gamma} - \boldsymbol{u}_{RC})] - \varrho sgn[\boldsymbol{s}(\underline{\boldsymbol{e}}(t))] \\ &= \mathbf{G}_n[\hat{\boldsymbol{Z}}^T(\boldsymbol{f}_m\tilde{\boldsymbol{m}} + \boldsymbol{f}_\sigma\tilde{\boldsymbol{\sigma}}) + \boldsymbol{\omega} - \boldsymbol{u}_{RC})] - \varrho sgn[\boldsymbol{s}(\underline{\boldsymbol{e}}(t))],\end{aligned} \tag{73}$$

where $\boldsymbol{\omega}$ denotes the approximation error: $\boldsymbol{\omega} = \hat{\boldsymbol{Z}}^T\boldsymbol{\beta} + \boldsymbol{\varepsilon} + \boldsymbol{\gamma}$. Putting the above together leads to the following theorem which guarantees the stability of the proposed control system.

Theorem 1 *For a non-linear robotic system represented by Eqn. (57), an intelligent control system T2FHC as specified in Eqn. (61) is guaranteed to be stable if the following conditions are satisfied:*

1) *The adaptive rules of T2FHC are designed as follows:*

$$\dot{\hat{\boldsymbol{m}}} = \boldsymbol{\eta}_m\boldsymbol{f}_m^T\hat{\boldsymbol{Z}}\boldsymbol{s}(\underline{\boldsymbol{e}}(t)), \tag{74}$$

$$\dot{\hat{\boldsymbol{\sigma}}} = \boldsymbol{\eta}_\sigma\boldsymbol{f}_\sigma^T\hat{\boldsymbol{Z}}\boldsymbol{s}(\underline{\boldsymbol{e}}(t)), \tag{75}$$

where $\boldsymbol{\eta}_m$ and $\boldsymbol{\eta}_\sigma$ denote the diagonal positive constant learning-rate matrices of $\dot{\hat{\boldsymbol{m}}}$ and $\dot{\hat{\boldsymbol{\sigma}}}$, respectively, and where $\dot{\hat{\boldsymbol{m}}}$ and $\dot{\hat{\boldsymbol{\sigma}}}$ must be used in accordance with the three situations of the Type-2 inference system as specified in Section III-B.

2) *The robust controller is designed as follows:*

$$\boldsymbol{u}_{RC} = (2\boldsymbol{R}^2)^{-1}(\boldsymbol{R}^2 + \boldsymbol{I})\boldsymbol{s}(\underline{\boldsymbol{e}}(t)), \tag{76}$$

where $\boldsymbol{R}$ is a positive diagonal matrix, $\boldsymbol{R} = diag(\phi_1, \phi_2, \cdots, \phi_i)$, and ϕ_i is a robust attenuation coefficient that can be adjusted externally.

Theorem 1 can be proofed using the Lyapunov stability theory, which is provided in the online supplementary.

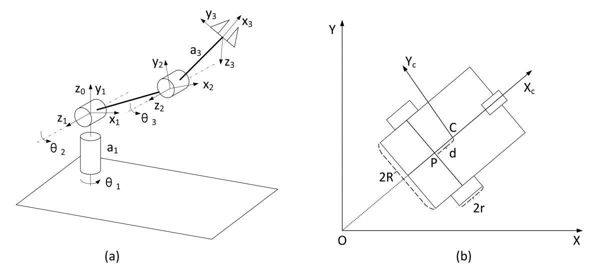


Fig. 4. (a) A simulated three-link robot manipulator; (b) A simulated two-wheeled differentially driven mobile robot.

V. Applications in Intelligent Robot Control

The proposed controller with the new T2FHC was applied to two typical robotic systems, a simulated three-link robot manipulator and a mobile robot, to verify its efficacy. A comparative study is also included in this section to demonstrate the performance of controller over a number of alternative approaches, including a PID controller, an SMC with fuzzy brain emotional learning controller network (FBELC) [5], and an SMC with fuzzy CMAC network (FCMAC) [22].

A. Robot Manipulator Control

1) Simulation Experimental Setup: The configuration of the simulated three-link robot manipulator employed in this experiment is shown in Fig. 4 (a). All three joints, whose angle values are labeled as θ_1, θ_2, and θ_3, are rotation mechanisms. The upper and lower limbs are labeled as a_2 and a_3, and a_1 is the link from the robot frame to the second joint. The non-linear dynamic equation of the manipulator is described using the following second-order differential equation:

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{\tau}_d = \boldsymbol{\tau}, \tag{77}$$

where $\boldsymbol{q}$ is a position vector indicating joint angles; $\dot{\boldsymbol{q}}$ is a velocity vector of the joints; $\ddot{\boldsymbol{q}}$ is an acceleration vector of the joints; $\boldsymbol{M}(\boldsymbol{q})$ is a moment of inertia; $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ denotes the coriolis and centripetal force; $\boldsymbol{g}(\boldsymbol{q})$ denotes the gravitational force; $\boldsymbol{\tau}_d$ denotes an external disturbance; and $\boldsymbol{\tau}$ denotes an input torque vector. The gravity acceleration, g, is set to $9.8m/s$.

TABLE I
PARAMETERS OF THE ROBOT MANIPULATOR

Items	$i=1$	$i=2$	$i=3$
$a_i(m)$	0.6	0.5	0.4
$l_i(m)$	0.3	0.25	0.2
$m_i^l(kg)$	3.0	1.8	1.5
$m_i^m(kg)$	0.3	0.3	0.3
$I_i^m(kgm^2)$	12.0×10^{-3}	12.0×10^{-3}	12.0×10^{-3}
$I_i^l(kgm^2)$	50.45×10^{-3}	32.68×10^{-3}	30.47×10^{-3}
$k_i^r(kgm^2)$	1.0	1.0	1.0

The system parameters of the manipulator are summarised in Table I, where a_i indicates the link length; l_i indicates the distance between the centre of mass and the joint of a link; m_i^l indicates the link mass; m_i^m denotes the motor mass of the joint; I_i^m denotes the moment of inertia of the link; I_i^l denotes the moment of inertia link's center of mass; and k_i^r represents the gearbox reduction ratio of the motor.

The dynamic equation of the robot manipulator is defined by:

$$\ddot{\boldsymbol{x}}(t)=\boldsymbol{f}(\underline{\boldsymbol{x}}(t))+\boldsymbol{G}(\underline{\boldsymbol{x}}(t))\boldsymbol{u}(t)+\boldsymbol{d}(t), \tag{78}$$

where $\boldsymbol{x}(t)$ is defined by:

$$\boldsymbol{x}(t)\triangleq[q_1(t)\quad q_2(t)\quad q_3(t)]^T=[x_1(t)\quad x_2(t)\quad x_3(t)]^T. \tag{79}$$

From which, it follows that:

$$\boldsymbol{f}(\underline{\boldsymbol{x}}(t))=-\boldsymbol{M}^{-1}(\boldsymbol{q})[\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}}+\boldsymbol{g}(\boldsymbol{q})] \tag{80}$$

$$\boldsymbol{G}(\underline{\boldsymbol{x}}(t)=\boldsymbol{M}^{-1}(\boldsymbol{q}) \tag{81}$$

$$\boldsymbol{d}(t)=-\boldsymbol{M}^{-1}(\boldsymbol{q})\boldsymbol{\tau}_d. \tag{82}$$

The external disturbance is given as:

$$\boldsymbol{\tau}_d=\chi\cdot\begin{bmatrix}0.2\sin(2t)\\0.1\cos(2t)\\0.1\sin(t)\end{bmatrix}, \tag{83}$$

and the initial conditions of the robot manipulator are defined as $\boldsymbol{x}(t)=[-0.3\quad 0.1-0.4]^T$ and $\dot{\boldsymbol{x}}(t)=[0\quad 0\quad 0]^T$; χ denotes the disturbance level, which was set to 5, 10, 20 in the experiments.

In the simulation, two reference modes of the manipulator were set. The manipulator needed to track the first reference mode when the robot started to move; after 15 seconds, the robot needed to track the second reference mode. The two reference modes are defined as follows:

$$\boldsymbol{ref}_1=\begin{bmatrix}0.5\sin(t+2.5)+0.35\cos(2t+1.5)\\0.2(\sin(t)+\sin(2t))\\0.13-0.1(\sin(t)+\sin(2t))\end{bmatrix}, \tag{84}$$

$$\boldsymbol{ref}_2=\begin{bmatrix}0.5(\sin(2t)+\cos(t+1))\\0.15\sin(2t)\cos(t+1)\\0.1(\cos(2t)-\sin(t))\end{bmatrix}, \tag{85}$$

where the time unit is set to $0.001s$. The sliding hyperplane is designed as: $\boldsymbol{s}(\underline{\boldsymbol{e}}(t))=10\boldsymbol{e}+0.55\dot{\boldsymbol{e}}(t)$; and the robust controller is designed as: $\boldsymbol{R}=0.075\boldsymbol{I}_{3\times3}$. In particular, for fair comparison, both FBELC and FCMAC methods were designed to share the same robust controller with T2FHC.

TABLE II
INITIALISED PARAMETER VALUES OF THE PROPOSED NEURAL NETWORK

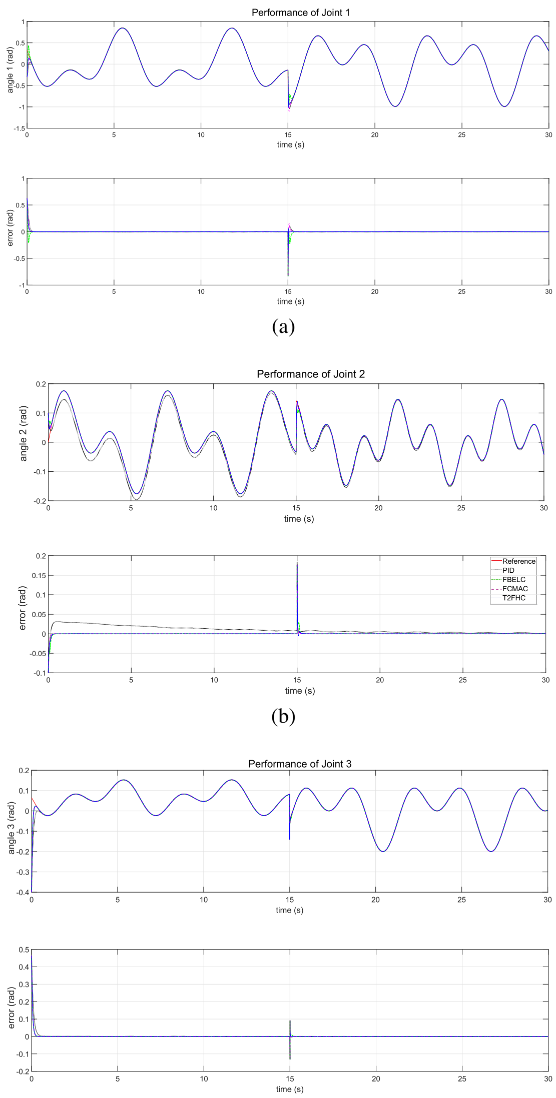
	Robot Manipulator	Mobile Robot
n_i	3	2
n_R	5	5
n_T	4	4
n_B	2	2
n_λ	8	8
$\underline{m}_{i\lambda}$	$[-2.5,-1.9,-1.3,-0.7,$ $-0.1,0.5,1.1,1.7]$	$[-4.6,-3.4,-2.2,-1.0,$ $0.2,1.4,2.6,3.8]$
$\overline{m}_{i\lambda}$	$[-1.7,-1.1,-0.5,0.1,$ $0.7,1.3,1.9,2.5]$	$[-3.8,-2.6,-1.4,-0.2,$ $1.0,2.2,3.4,4.6]$
$\sigma_{i\lambda}$	1.2	1.0
η^m,η^σ	0.001	0.001
α,β	0.5, 0.5	0.8, 0.8
b,c	10, 1	200, 1

The parameters of the T2FHC network were initialized as listed in Table II, where n_i denotes the dimensionality of the network inputs; n_R indicates the number of regions in the input layer; n_T denotes the number of block types; n_B is the number of blocks; n_λ denotes the number of receptive-fields; $\underline{m}_{i\lambda}$, $\overline{m}_{i\lambda}$, and $\sigma_{i\lambda}$ denote the Gaussian function parameters; η^m and η^σ are the brain emotional learning rates; and b and c are the gain parameters of the brain emotional learning.

2) Results: Simulation results of the position responses and tracking errors using different controllers are shown in Fig. 5. Sub-figures 5a, 5b, and 5c illustrate the simulated position responses and tracking errors of Joints 1, 2, and 3. Each sub-figure contains the reference trajectory (the red solid line), the PID output trajectory (the dotted line), the FBELC output trajectory (the dot-dashed line), the FCMAC output trajectory (the dashed line), and the proposed T2FHC output trajectory (the blue solid line).

In each of the sub-figures, the upper row shows the corresponding controller's joint trajectory and the bottom row shows the errors between the controller's trajectory and the reference trajectory. Note that the tracking trajectories of the robot were changed after 15 seconds, which led to sudden changes at the $15th$ second for all the simulated trajectories. For Joint 1, all four controllers successfully followed the reference trajectory after a settling down period. However, PID controller's performance in Joint 2 was much worse than those of other neural network controllers. Such poor performances indicate that each joint motor requires a separate PID parameter setup, rather than being fixed to a single one. Yet, optimizing a range of different PID parameters would require significant human intervention. In contrast to this, all the three neural network based controllers can educe tracking errors automatically, through their online tuning ability.

Examining the results more closely, as reflected by Fig. 5a, since Joint 1 handled forces that were exerted from Joints 2 and 3, all controllers performed less stable for Joint 1 than Joints 2 and 3. At the $15th$ second, the errors of all controllers reached around $-0.8rad$. In particular, the FBELC could not converge rapidly, it always had a tracking delay at the $0th$ second and the $15th$ second. Joint 2 also needed to tackle the force exerted from Joint 3; however, the force was much smaller than that of Joint 1. Thus, all the neural network based controllers generated relatively better performance, with the largest error in the $15th$ second being about $0.2rad$. Since Joint 3 was the terminal joint in the robot manipulator, no



(a)

(b)

(c)

Fig. 5. Simulation results of position responses and tracking errors with different PID, FBELC, FCMAC, and T2FHC controllers: (a) Results in Joint 1; (b) Results in Joint 2; (c) Results in Joint 3.

exerted force needed to be considered; the trajectories of all controllers were close to the reference.

The performances of the FCMAC and T2FHC in Joints 2 and 3 were close to each other, both controllers could rapidly converge in reducing the tracking errors. The FCMAC only showed a slight lead in Joint 2. However, in Joint 1, the T2FHC controller performed much better than the FCMAC controller. Indeed, the trajectories of the T2FHC always achieved the fastest convergence amongst all four controllers.

To demonstrate the disturbance resistance of the proposed network controller, the quantitative comparisons of the PID, FBELC, FCMAC, HC, FHC and the proposed T2FHC under the three levels of disturbances are summarised in Table III. Amongst them, the HC is essentially the proposed T2FHC without the use of Type-2 fuzzy sets, and the FHC is the proposed hybrid controller neural network with Type-1 fuzzy sets. The accumulated RMSE values for Joints 1, 2 and 3 were used to measure the overall performance over the

TABLE III
COMPARISON OF PID, FBELC, FCMAC, AND T2FHC CONTROLLERS FOR ROBOT MANIPULATOR ($RMSE \times 0.01$)

χ	JOINT	PID	FBELC	FCMAC	HC	FHC	T2FHC
5	Joint 1	3.167	2.675	2.703	8.141	2.628	**2.464**
	Joint 2	1.398	0.473	0.392	3.612	0.390	**0.387**
	Joint 3	1.921	1.426	1.427	1.707	1.423	**1.405**
10	Joint 1	3.176	2.675	2.701	8.149	2.627	**2.445**
	Joint 2	1.403	0.474	0.392	3.573	0.385	**0.384**
	Joint 3	1.926	1.414	1.427	1.703	1.427	**1.405**
20	Joint 1	3.209	2.675	2.700	8.157	2.625	**2.445**
	Joint 2	1.422	0.475	0.391	3.530	0.386	**0.384**
	Joint 3	1.942	1.414	1.427	1.699	1.422	**1.405**

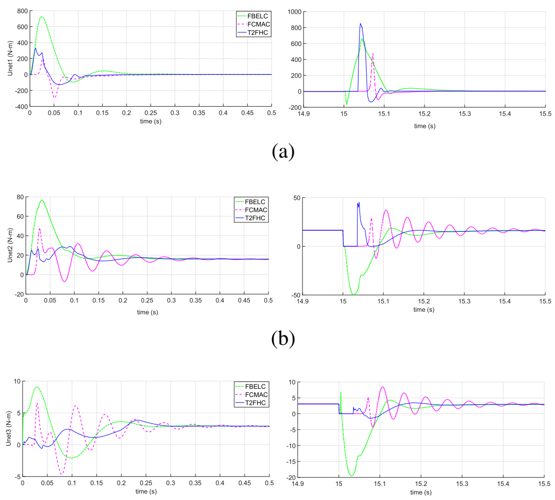
period of $[0s, 30s]$. As can be seen from the table, the RMSE values of the three neural network based controllers are all less than that of the PID controller. Importantly, the T2FHC achieved the best performance in all the joints amongst all controllers. Besides, Joint 1 played a more important role in the entire robot manipulator's tracking precision. Thus, overall, the T2FHC controller achieved the best control performance in this experiment. Note that: the fuzzy sets in the T2HFC and FHC are represented by a set of Gaussian functions with the parameters of uncertain means and variances. These parameters are randomly initialised and can be adjusted by using the updating rules designed in this paper. Therefore, the parameters are totally different from their initial values.

Fig. 6 shows the control efforts of the FBELC, FCMAC, and T2FHC network controllers for the joints; the left column shows the efforts during $[0s, 0.5s]$ and the right does those during $[14.9s, 15.5s]$. When the reference trajectory changed, the FBELC controller immediately reacted to the errors; however, both the FCMAC and the T2FHC controller could not generate any response until their robust controllers have reduced the errors to a certain level (until after about $0.03s$). This is because the sliding surface would sharply increase the input values $\boldsymbol{s}(\underline{\boldsymbol{e}}(t))$ when the tracking trajectory changed, and the increased values were out of the input range of the FCMAC and T2FHC networks. Since these two networks contain a multiplicative mechanism as defined in Eqn. (9), $\mathbb{F}$ tends to being 0. However, the FBELC does not contain such a mechanism, and its react speed is thus faster than the other two. Unfortunately, such a fast speed caused the poor tracking performance as shown in Figs. 5. This situation implies that overly sensitive to reaction can lead to unexpected vibrations.

Different from the FCMAC, however, the proposed T2FHC utilizes a Type-2 fuzzy inference mechanism, which contains a larger input range than that of the FCMAC. Thus, the reacting speed of the T2FHC is faster than that of the FCMAC. Also, in Fig. 6, the T2FHC used less time to stabilise its output; in contrast, the FCMAC generated considerable vibrations that reduced the overall accuracy of the manipulator. Therefore, once again, the T2FHC offered the best control performance for the robot manipulator in the experiment.

B. Mobile Robot Control

1) Simulation Experimental Setup: Fig. 4 (b) illustrates a typical mobile robot with two differentially driven coaxial wheels and a front passive wheel. The coaxial wheels are driven by two independent motors, and the passive wheel



(a)

(b)

(c)

Fig. 6. Simulation results on control efforts of FBELC, FCMAC, and T2FHC network controllers in three joints: (a) Results in Joint 1; (b) Results in Joint 2; and (c) Results in Joint 3.

simply assists to keep the balance. In this figure, r denotes the radius of the wheel, $2R$ denotes the distance between the two wheels, C denotes the centre of gravity of the robot, (x_c, y_c) denotes the geometry centre position of the robot, P denotes the midpoint of the two wheels' axis, and θ denotes the robot's orientation against to the reference coordinate system. The position of the mobile robot in the reference coordinate system is expressed as $\boldsymbol{q} = [x_c \quad y_c \quad \theta]^T$. It follows that $\dot{\boldsymbol{q}} = [\dot{x}_c \quad \dot{y}_c \quad \dot{\theta}]$, where $\boldsymbol{v}(t) = [\upsilon \quad \varpi]^T$, υ and ϖ are the translational and angular velocities of the robot.

In general, the dynamics of a mobile robot with n generalized coordinates can be expressed as:

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{F}(\dot{\boldsymbol{q}}) + \boldsymbol{\tau}_d = \boldsymbol{B}(\boldsymbol{q})\boldsymbol{\tau} - \boldsymbol{A}(\boldsymbol{q})\boldsymbol{\psi}, \tag{86}$$

where $\boldsymbol{q}$ is the position and orientation vector of the robot; $\dot{\boldsymbol{q}}$ is the velocity vector of the position and orientation; $\ddot{\boldsymbol{q}}$ is the acceleration vector of the position and orientation; $\boldsymbol{M}(\boldsymbol{q})$ is the moment of inertia; $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ denotes the coriolis and centripetal force; $\boldsymbol{g}(\boldsymbol{q})$ denotes the gravitational force and for the mobile robot moving on horizontal ground, $\boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{0}$; $\boldsymbol{\tau}$ denotes an input torque vector; $\boldsymbol{B}(\boldsymbol{q})$ denotes an input transformation matrix; $\boldsymbol{F}(\dot{\boldsymbol{q}})$ denotes a friction vector; $\boldsymbol{\tau}_d$ indicates an external disturbance; $\boldsymbol{A}(\boldsymbol{q})$ denotes a constraint matrix; and $\boldsymbol{\psi}$ denotes a Lagrange multiplier vector.

The mobile robot is required to track the reference trajectory, which is defined by $\boldsymbol{q}_r = [x_r \quad y_r \quad \theta_r]^T$. This means that the tracking error, $\boldsymbol{e}_p$, can be obtained by:

$$\boldsymbol{e}_p = \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_r - x \\ y_r - y \\ \theta_r - \theta \end{bmatrix}, \tag{87}$$

and $\dot{\boldsymbol{e}}_p$ is defined by:

$$\dot{\boldsymbol{e}}_p = \begin{bmatrix} \varpi e_2 - \upsilon + \upsilon_r \cos e_3 \\ -\varpi e_1 + \upsilon_r \sin\theta e_3 \\ \varpi_r - \varpi \end{bmatrix}. \tag{88}$$

In order to track the given velocity reference model, the approach reported in [61] is adapted to calculate the desired translational and angular velocities, which is defined by:

$$\boldsymbol{v}_d = \begin{bmatrix} \upsilon_d \\ \varpi_d \end{bmatrix} = \begin{bmatrix} \upsilon_r \cos e_3 + k_1 e_1 \\ \upsilon_r + \frac{k_2 \upsilon_r e_2 \sin e_3}{e_3} + k_3 e_3 \end{bmatrix}, \tag{89}$$

where k_1, k_2, and k_3 are implementation parameters. Thus, the velocity error $\boldsymbol{e}_v$ is calculated by:

$$\boldsymbol{e}_v = \boldsymbol{v}_d - \boldsymbol{v} = [\boldsymbol{e}_\upsilon(t) \quad \boldsymbol{e}_\varpi(t)]^T. \tag{90}$$

Eqn. (90) implies that the following relationships hold between the torques of the left and right wheels, and υ and ϖ:

$$\begin{cases} \upsilon \propto \tau_r + \tau_l \\ \varpi \propto \tau_r - \tau_l \end{cases}. \tag{91}$$

Without losing generality, denote the output of the controller as $\boldsymbol{u} = [u_1 \quad u_2]^T$. It follows that $\tau_r = \frac{u_1+u_2}{2}$ and $\tau_l = \frac{u_1-u_2}{2}$. In this simulation experimental investigation, the parameters of the mobile robot were set as follows: $m = 10kg$, $I = 5kg{\cdot}m^2$, $R = 0.2m$, $r = 0.05m$, $d = 0.05m$, $\boldsymbol{F}(\dot{\boldsymbol{q}}) = \boldsymbol{0}$. In addition, the disturbance, $\boldsymbol{\tau}_d$, is defined as:

$$\boldsymbol{\tau}_d = \chi \cdot \begin{bmatrix} 2.5\sin(4t) \\ 2.5\cos(4t) \end{bmatrix}, \tag{92}$$

where χ denotes the disturbance level, which was set to 5, 10, 20 in the experiments. The reference trajectory is defined as:

$$\begin{cases} \dot{\theta}_r = \varpi_r T \\ \dot{x}_r = \upsilon_r \cos 2\theta_r \\ \dot{y}_r = \upsilon_r \sin\theta_r, \end{cases} \tag{93}$$

where the initial values of the reference trajectory were $\upsilon_r = 0.2m/s$, $\varpi = 0.1rad/s$, $\theta_r = 0$; and the time unit, T, was set to $0.01s$.

The starting positions of the reference trajectory and the robots were $\boldsymbol{q}_r = [2 \quad 0 \quad \frac{\pi}{2}]^T$ and $\boldsymbol{q} = [1 \quad 1 \quad \frac{\pi}{2}]^2$, respectively; the parameters of the velocity reference model were set to: $k_1 = 4$, $k_2 = 80$, and $k_3 = 1$; the sliding hyperplane for the mobile robot was designed as: $\boldsymbol{s}(\underline{e}(t)) = 10\boldsymbol{e} + 0.01\dot{\boldsymbol{e}}(t)$; and the robust controller was designed as: $\boldsymbol{R} = 0.5\boldsymbol{I}_{2\times2}$. The initialization parameters of the T2FHC network is summarised in Table II.

2) Results: Fig. 7 demonstrates the simulated position response of the mobile robot over $65s$. In this figure, the color codes of the trajectory lines were identical to those used previously. The left figure presents the entire tracking process and the right one is a magnified version of the tracking trajectory over the period of $[0s, 10s]$. The performances of the T2FHC and FCMAC controllers were very close to each other, with almost coincide trajectories both reaching the reference trajectory earlier than the FBELC. The PID controller was underperformed compared with the rest; it had a longer vibration time. Thus, the T2FHC and FCMAC offered relatively better results.

The convergence performances in terms of the robot position errors of the compared four controllers is illustrated in Fig. 8. The three plots in the upper row indicate the errors in x, y, and θ; and the other three in the bottom row indicate the

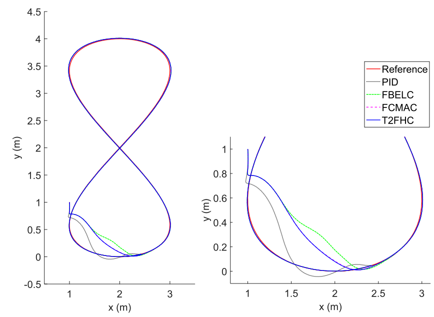


Fig. 7. Simulation results of PID, FBELC, FCMAC, and T2FHC controllers for moving-target tracking.

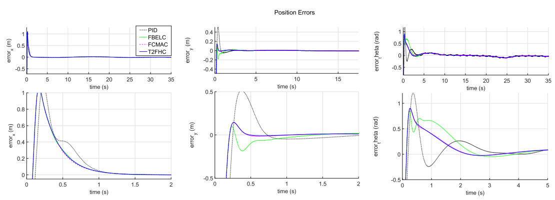


Fig. 8. Position errors of PID, FBELC, FCMAC, and T2FHC controllers: the upper-row plots indicate errors in x, y, and θ; and the bottom-row plots show magnifications of those upper ones.

magnified versions of those upper ones after a few seconds of tracking. The T2FHC and FCMAC controllers generated very similar results regarding the position tracking errors. Although the FBELC controller also generated a similar result with those of the T2FHC and FCMAC controllers in x, it took longer to converge in y and θ, with particularly a significant longer convergence in y.

The velocity tracking errors of the robot is presented in Fig. 9. The two plots in the upper row show the translational and angular velocities (υ, ϖ), whilst the middle and bottom rows show the corresponding magnifications of the upper ones over the period of $[0s, 3s]$. This figure clearly reveals the performance differences between the T2FHC and FCMAC controllers. The convergence speed of the T2FHC controller was much faster than that of the FCMAC, with the former converging at $0.75s$ in both υ and ϖ and the latter at $1.2s$ in υ and $0.9s$ in ϖ. Additionally, this figure also shows that the performance of the FBELC controller is better than that of the FCMAC in υ, but it performed least satisfactorily in ϖ amongst all the neural network-based controllers. Nevertheless, it still outperformed the PID controller which required a long convergence time in both υ and ϖ.

Also to demonstrate the disturbance resistance of the proposed network controller, quantitative performance comparisons of using the PID, FBELC, FCMAC, and T2FHC for mobile robot control under the three levels of disturbance are summarized in Table IV. The accumulated RMSE values over the entire tracking process of the robot's position P_e, orientation θ, translational velocity υ, and angular velocity ϖ were used to measure the performance. This table reflects a very similar phenomenon with that in the simulated robotic manipulator: the proposed T2FHC network controller performed the best regarding the position, orientation, and angular

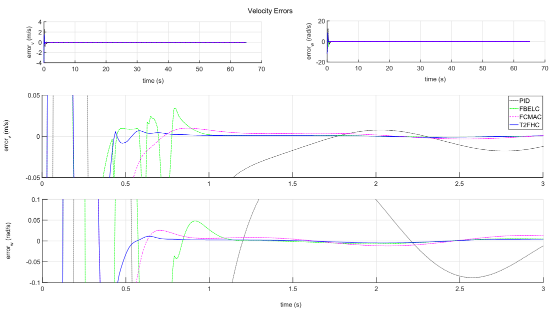


Fig. 9. Velocity errors of PID, FBELC, FCMAC, and T2FHC controllers: Upper row shows translational and angular velocities, and middle and bottom rows show two magnifications of those upper ones over $[0s, 3s]$.

TABLE IV
COMPARISON OF PID, FBELC, FCMAC, AND T2FHC CONTROLLERS FOR MOBILE ROBOT ($RMSE \times 0.01$)

χ	ERR	PID	FBELC	FCMAC	HC	FHC	T2FHC
5	P_e	2.631	2.484	2.386	2.500	2.380	**2.332**
	θ_e	9.599	7.455e	6.815	9.149	6.810	**6.423**
	υ_e	11.40	8.491	6.749	11.830	6.758	**4.520**
	ϖ_e	101.1	82.02	67.25	15.00	67.25	**50.51**
10	P_e	2.964	2.496	2.390	2.501	2.409	**2.340**
	θ_e	11.55	7.448	6.793	9.161	6.813	**6.422**
	υ_e	11.63	8.510	6.745	11.86	6.791	**4.529**
	ϖ_e	102.8	82.35	67.51	150.3	67.52	**50.64**
20	P_e	3.611	2.513	2.656	2.504	3.069	**2.350**
	θ_e	16.45	7.442	7.382	9.180	6.846	**6.430**
	υ_e	12.39	8.548	26.30	11.89	21.53	**4.546**
	ϖ_e	106.61	82.74	68.19	150.6	68.15	**50.79**

velocity tracking. As with all other simulation experimental results, the PID controller was unable to perform so good as any of the neural network-based controllers.

C. Discussion and Analysis

1) Discussions: In both simulation experiments as reported in Sections V-A and V-B, the FCMAC, FBELC, and T2FHC used the same number of Gaussian function units, with each network employing 8 receptive-fields. Given this common ground, overall the T2FHC managed to perform the best in terms of control effectiveness. It also achieved the best performance in terms of error convergence rate. These benefits resulted from the Type-2 fuzzy inference system in the proposed T2FHC network; since the Type-2 system involves more adjustable parameters than the Type-1 fuzzy system used in the other two networks, so as to be able to handle more complex uncertainties.

The final interval Type-2 fuzzy sets for the two experiments are illustrated in Fig. 10. Recall that each input dimension was evenly partitioned into four regions, which were accumulated into 2 blocks in both experiments each represented by an interval Type-2 fuzzy set. The final blocks of the T2FHC for the manipulator control regarding the three system inputs are demonstrated in Figs. 10-(a-c), whilst those for the mobile robot control are illustrated in Figs. 10-(d-e). It can be realized from this figure that the shape of each of these final Type-2 fuzzy set is different from those of other fuzzy sets due to the application of the adaptive updating rules designed in Section III-B. This figure therefore confirms the effectiveness of the automatic rule updating mechanism in the proposed T2FHC.

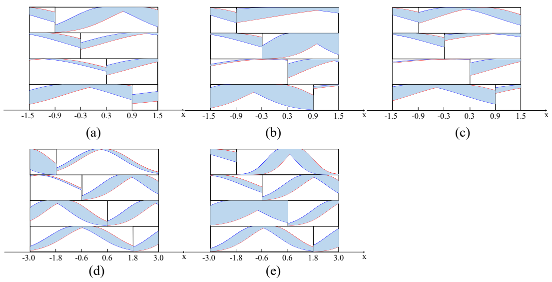


Fig. 10. The evolved Type 2 fuzzy sets of T2FHC controllers with $\chi = 5$. (a) Input x_1 of the manipulator control; (b) Input x_2 of the manipulator control; (c) Input x_3 of the manipulator control; (d) Input x_1 of the mobile robot control; and (e) Input x_2 of the mobile robot control.

The performance of T2FHC in mobile robot position tracking offers significant improvements against the use of FCMAC and FBELC. In contrast to the robot manipulator, the non-linear property exhibited by the mobile robot dynamics is not complicated. Therefore, the application of the T2FHC in the robot manipulator simulation experiment can better reveal its powerful ability in handling uncertainties, non-linearity and dynamics. In particular, Joint 1 of the manipulator needed to deal with the exerted efforts from the upper and lower limbs; also Joint 1 itself had the heaviest motor mass. Under such harsh conditions, the T2FHC network controller achieved an excellent performance, representing the best amongst all the examined controllers. To summarise, the simulation experimental investigations confirm that the proposed T2FHC is more capable of dealing with external disturbance, including that led by the influence of modeling uncertainties.

2) Statistical Analysis: Ten additional repeated experiments for each robotic system with $\chi = 5$, 10 and 20 were also conducted to confirm the statistical significance of the improvement led by the proposed method. The average of the accumulated RMSEs over the ten repeated experiments are summarized in Table V. From this table, it is clear that the proposed T2FHC consistently outperformed all other referenced controllers, given that the average accumulated RMSE values led by the T2FHC are all smaller than their counterparts resulted from other referenced approaches. This demonstrates the stability of the proposed system in producing improved control results, which re-validates the proposed system and reassures its efficacy in dynamic robotic control.

The t-test was additionally conducted for the above experiment as reported in Table V to investigate the statistical significance of the performance of the proposed T2FHC. The null hypothesis was carried out for the t-test; thus, the p-values of FBELC, FCMAC, HC, and FHC against T2FHC are summarized in Table VI, which exhibits that all the p-values are much less than 0.05. Therefore, the performance of the T2FHC-based controller is largely different to those from other referenced approaches, despite of its confirmed superiority as demonstrated in Table V.

TABLE V
STATISTICAL ANALYSIS OF FBELC, FCMAC, HC, FHC, AND T2FHC

Items	FBELC	FCMAC	HC	FHC	T2FHC
J_1-5	2.675e-02	2.702e-02	8.120e-02	2.628e-02	2.452e-02
J_2-5	4.770e-03	3.921e-03	3.679e-02	3.896e-02	3.830e-03
J_3-5	1.413e-02	1.426e-02	1.714e-02	1.422e-02	1.405e-02
J_1-10	2.674e-02	2.702e-02	8.154e-02	2.627e-02	2.442e-02
J_2-10	4.749e-03	3.917e-03	3.570e-02	3.863e-03	3.846e-03
J_3-10	1.413e-02	1.428e-02	1.717e-02	1.423e-02	1.405e-02
J_1-20	2.680e-02	2.702e-02	8.157e-02	2.625e-02	2.448e-02
J_2-20	4.776e-03	3.918e-03	3.573e-02	3.865e-03	3.836e-03
J_3-20	1.415e-02	1.427e-02	1.724e-02	1.423e-02	1.407e-02
P-5	2.486e-02	2.387e-02	2.499e-02	2.381e-02	2.333e-02
θ-5	7.456e-02	6.816e-02	9.144e-02	6.806e-02	6.423e-02
υ-5	8.489e-02	6.746e-02	1.187e-01	6.764e-02	4.517e-02
ϖ-5	8.211e-01	6.724e-01	1.497e-00	6.725e-01	5.051e-01
P-10	2.496e-02	2.391e-02	2.500e-02	2.410e-02	2.339e-02
θ-10	7.450e-02	6.833e-02	9.168e-02	6.816e-02	6.423e-02
υ-10	8.498e-02	6.781e-02	1.179e-01	6.789e-02	4.530e-02
ϖ-10	8.224e-01	6.725e-01	1.502e-00	6.752e-01	5.061e-01
P-20	2.512e-02	2.687e-02	2.504e-02	3.010e-02	2.351e-02
θ-20	7.454e-02	7.384e-02	9.185e-02	6.891e-02	6.429e-02
υ-20	8.495e-02	2.681e-01	1.187e-01	2.154e-01	4.547e-02
ϖ-20	8.266e-01	6.815e-01	1.516e-00	6.820e-01	5.079e-01

TABLE VI
p-VALUES OF FBELC, FCMAC, HC, AND FHC AGAINST T2FHC

Items	FBELC	FCMAC	HC	FHC
J_1-5	2.003e-12	3.802e-13	3.590e-21	1.126e-11
J_2-5	1.193e-11	1.194e-05	2.466e-13	4.970e-02
J_3-5	7.034e-05	9.328e-17	2.631e-12	2.284e-14
J_1-10	1.741e-08	1.159e-08	2.909e-20	4.445e-07
J_2-10	1.210e-12	3.503e-09	4.831e-12	4.416e-05
J_3-10	8.121e-07	2.593e-18	5.190e-11	2.599e-16
J_1-20	3.374e-07	2.627e-07	1.205e-19	1.287e-05
J_2-20	9.419e-15	8.127e-03	2.492e-13	4.044e-02
J_3-20	7.676e-06	3.014e-13	3.413e-12	1.531e-12
P-5	1.230e-13	6.166e-14	1.449e-19	1.061e-11
θ-5	1.241e-12	3.163e-16	7.848e-24	1.753e-16
υ-5	9.596e-18	8.263e-21	5.362e-17	9.455e-21
ϖ-5	3.780e-20	3.651e-31	1.475e-25	4.229e-31
P-10	1.978e-14	1.805e-17	4.480e-20	1.906e-14
θ-10	1.607e-12	1.501e-22	5.369e-25	9.961e-17
υ-10	5.056e-19	5.185e-26	1.511e-18	1.934e-23
ϖ-10	2.722e-19	2.364e-39	5.831e-25	1.083e-34
P-20	8.857e-15	1.294e-06	3.805e-19	4.380e-06
θ-20	1.573e-13	5.485e-07	3.468e-25	1.451e-02
υ-20	3.409e-21	5.591e-07	4.725e-18	4.227e-07
ϖ-20	8.551e-23	2.149e-23	7.101e-25	9.178e-23

VI. CONCLUSION

This paper has proposed a novel fuzzy neural network that integrates the key components of Type-2 fuzzy CMAC and BELC. The resultant network has also been combined with a sliding mode controller for performing dynamic non-linear control. It has been theoretically proven that the system implementing the proposed approach is asymptotically stable with guaranteed convergence. Simulation experimental studies have demonstrated that the implemented system using the T2FHC led to preciser position tracking and more favorable stability in comparison with the results generated using alternative, recently-developed network-based controllers, such as fuzzy CMAC and fuzzy BELC (all of these beat the classical PID controllers significantly). This shows the potential of the proposed approach for real-world applications, especially when concerning multiple-degrees of freedom robot manipulators.

This research can be further improved in several directions. The parameters used in the T2FHC provide great flexibility in modeling non-linearity and uncertainty, but they need to be initialised using empirical knowledge. It is therefore of great practical significance to investigate the automation of such initialization process in an effort to prompt the applicability of the proposed approach. In addition, it is worthwhile to study the interpretability of Type-2 fuzzy sets and the generalisability of the proposed method.

REFERENCES

[1] W. Zhang, X. Lin, and B. S. Chen, "Lasalle-type theorem and its applications to infinite horizon optimal control of discrete-time nonlinear stochastic systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 1, pp. 250–261, Jan 2017.

[2] W. Zhang, W. X. Zheng, and B.-S. Chen, "Detectability, observability and lyapunov-type theorems of linear discrete time-varying stochastic systems with multiplicative noise," *International Journal of Control*, vol. 90, no. 11, pp. 2490–2507, 2017.

[3] R. J. Wai and Y. W. Lin, "Adaptive moving-target tracking control of a vision-based mobile robot via a dynamic petri recurrent fuzzy neural network," *IEEE Transactions on Fuzzy Systems*, vol. 21, no. 4, pp. 688–701, Aug 2013.

[4] Q. Gao, G. Feng, L. Liu, J. Qiu, and Y. Wang, "Robust H_∞ control for stochastic – fuzzy systems via integral sliding-mode approach," *IEEE Transactions on Fuzzy Systems*, vol. 22, no. 4, pp. 870–881, Aug 2014.

[5] C. M. Lin and C. C. Chung, "Fuzzy brain emotional learning control system design for nonlinear systems," *International Journal of Fuzzy Systems*, vol. 17, no. 2, pp. 117–128, 2015.

[6] H. Li, J. Yu, C. Hilton, and H. Liu, "Adaptive sliding-mode control for nonlinear active suspension vehicle systems using T-S fuzzy approach," *IEEE Transactions on Industrial Electronics*, vol. 60, no. 8, pp. 3328–3338, Aug 2013.

[7] F. J. Lin, I. F. Sun, K. J. Yang, and J. K. Chang, "Recurrent fuzzy neural cerebellar model articulation network fault-tolerant control of six-phase permanent magnet synchronous motor position servo drive," *IEEE Transactions on Fuzzy Systems*, vol. 24, no. 1, pp. 153–167, Feb 2016.

[8] C. Shang, D. Reay, and B. Williams, "Adapting CMAC neural networks with constrained LMS algorithm for efficient torque ripple reduction in switched reluctance motors," *IEEE Transactions on Control Systems Technology*, vol. 7, no. 4, pp. 401–413, Jul 1999.

[9] O. Castillo and P. Melin, "A review on interval type-2 fuzzy logic applications in intelligent control," *Information Sciences*, vol. 279, pp. 615–631, 2014.

[10] D. H. Zhai and Y. Xia, "Adaptive fuzzy control of multilateral asymmetric teleoperation for coordinated multiple mobile manipulators," *IEEE Transactions on Fuzzy Systems*, vol. 24, no. 1, pp. 57–70, Feb 2016.

[11] Z. Li, Y. Xia, and F. Sun, "Adaptive fuzzy control for multilateral cooperative teleoperation of multiple robotic manipulators under random network-induced delays," *IEEE Transactions on Fuzzy Systems*, vol. 22, no. 2, pp. 437–450, April 2014.

[12] D. Chwa, "Fuzzy adaptive tracking control of wheeled mobile robots with state-dependent kinematic and dynamic disturbances," *IEEE Transactions on Fuzzy Systems*, vol. 20, no. 3, pp. 587–593, June 2012.

[13] C. L. Hwang and W. L. Fang, "Global fuzzy adaptive hierarchical path tracking control of a mobile robot with experimental validation," *IEEE Transactions on Fuzzy Systems*, vol. 24, no. 3, pp. 724–740, June 2016.

[14] C. F. Wu, B. S. Chen, and W. Zhang, "Multiobjective investment policy for a nonlinear stochastic financial system: A fuzzy approach," *IEEE Transactions on Fuzzy Systems*, vol. 25, no. 2, pp. 460–474, April 2017.

[15] Q. Gao, G. Feng, D. Dong, and L. Liu, "Universal fuzzy models and universal fuzzy controllers for discrete-time nonlinear systems," *IEEE Transactions on Cybernetics*, vol. 45, no. 5, pp. 880–887, May 2015.

[16] Q. Gao, G. Feng, Z. Xi, Y. Wang, and J. Qiu, "A new design of robust H_∞ sliding mode control for uncertain stochastic t-s fuzzy time-delay systems," *IEEE Transactions on Cybernetics*, vol. 44, no. 9, pp. 1556–1566, Sept 2014.

[17] Q. Gao, L. Liu, G. Feng, and Y. Wang, "Universal fuzzy integral sliding-mode controllers for stochastic nonlinear systems," *IEEE Transactions on Cybernetics*, vol. 44, no. 12, pp. 2658–2669, Dec 2014.

[18] C. Fu, A. Sarabakha, E. Kayacan, C. Wagner, R. John, and J. M. Garibaldi, "A comparative study on the control of quadcopter uavs by using singleton and non-singleton fuzzy logic controllers," in *2016 IEEE International Conference on Fuzzy Systems*, 2016, pp. 1023–1030.

[19] H. K. Lam, M. Narimani, H. Li, and H. Liu, "Stability analysis of polynomial-fuzzy-model-based control systems using switching polynomial Lyapunov function," *IEEE Transactions on Fuzzy Systems*, vol. 21, no. 5, pp. 800–813, Oct 2013.

[20] B.-S. Chen and S.-J. Ho, "Multiobjective tracking control design of ts fuzzy systems: Fuzzy pareto optimal approach," *Fuzzy Sets and Systems*, vol. 290, pp. 39 – 55, 2016.

[21] C. C. Chung and C. M. Lin, "Fuzzy brain emotional cerebellar model articulation control system design for multi-input multi-output nonlinear," *Acta Polytechnica Hungarica*, vol. 12, no. 4, pp. 39–58, 2015.

[22] C. M. Lin and H. Y. Li, "TSK fuzzy cmac-based robust adaptive backstepping control for uncertain nonlinear systems," *IEEE Transactions on Fuzzy Systems*, vol. 20, no. 6, pp. 1147–1154, Dec 2012.

[23] T. Mai and Y. Wang, "Adaptive force/motion control system based on recurrent fuzzy wavelet CMAC neural networks for condenser cleaning crawler-type mobile manipulator robot," *IEEE Transactions on Control Systems Technology*, vol. 22, no. 5, pp. 1973–1982, Sept 2014.

[24] C. M. Lin, M. S. Yang, F. Chao, X. M. Hu, and J. Zhang, "Adaptive filter design using Type-2 fuzzy cerebellar model articulation controller," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 10, pp. 2084–2094, Oct 2016.

[25] C. Lucas, D. Shahmirzadi, and N. Sheikholeslami, "Introducing belbic: brain emotional learning based intelligent controller," *Intelligent Automation & Soft Computing*, vol. 10, no. 1, pp. 11–21, 2004.

[26] E. Daryabeigi, G. R. A. Markadeh, and C. Lucas, "Emotional controller (BELBIC) for electric drives: A review," in *36th Annual Conference on IEEE Industrial Electronics Society*, 2010, pp. 2901–2907.

[27] M. A. Sharbafi, C. Lucas, and R. Daneshvar, "Motion control of omni-directional three-wheel robots by brain-emotional-learning-based intelligent controller," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 40, no. 6, pp. 630–638, 2010.

[28] E. Lotfi and M. R. Akbarzadeh-T, "Emotional brain-inspired adaptive fuzzy decayed learning for online prediction problems," in *2013 IEEE International Conference on Fuzzy Systems*, 2013, pp. 1–7.

[29] M. A. Sanchez, J. R. Castro, V. Ocegueda-Miramontes, and L. Cervantes, "Hybrid learning for general type-2 tsk fuzzy logic systems," *Algorithms*, vol. 10, no. 3, p. 99, 2017.

[30] E. Ontiveros-Robles, P. Melin, and O. Castillo, "Comparative analysis of noise robustness of type 2 fuzzy logic controllers," *Kybernetika*, vol. 54, no. 1, pp. 175–201, 2018.

[31] J. M. Mendel and R. I. B. John, "Type-2 fuzzy sets made simple," *IEEE Transactions on Fuzzy Systems*, vol. 10, no. 2, pp. 117–127, Apr 2002.

[32] C. Wagner, S. Miller, J. M. Garibaldi, D. T. Anderson, and T. C. Havens, "From interval-valued data to general type-2 fuzzy sets," *IEEE Transactions on Fuzzy Systems*, vol. 23, no. 2, pp. 248–269, April 2015.

[33] J. H. Aladi, C. Wagner, and J. M. Garibaldi, "Type-1 or interval type-2 fuzzy logic systems – on the relationship of the amount of uncertainty and FOU size," in *2014 IEEE International Conference on Fuzzy Systems*, 2014, pp. 2360–2367.

[34] S. Miller, M. Gongora, J. Garibaldi, and R. John, "Interval type-2 fuzzy modelling and stochastic search for real-world inventory management," *Soft Computing*, vol. 16, no. 8, pp. 1447–1459, 2012.

[35] G. M. Mendez and O. Castillo, "Interval type-2 tsk fuzzy logic systems using hybrid learning algorithm," in *The 14th IEEE International Conference on Fuzzy Systems, 2005. FUZZ '05.*, May 2005, pp. 230–235.

[36] O. Castillo, J. R. Castro, P. Melin, and A. Rodriguez-Diaz, "Application of interval type-2 fuzzy neural networks in non-linear identification and time series prediction," *Soft Computing*, vol. 18, no. 6, pp. 1213–1224, Jun 2014.

[37] J. M. Mendel, R. I. John, and F. Liu, "Interval type-2 fuzzy logic systems made simple," *IEEE Transactions on Fuzzy Systems*, vol. 14, no. 6, pp. 808–821, Dec 2006.

[38] J. H. Aladi, C. Wagner, A. Pourabdollah, and J. M. Garibaldi, "Contrasting singleton type-1 and interval type-2 non-singleton type-1 fuzzy logic systems," in *2016 IEEE International Conference on Fuzzy Systems*, 2016, pp. 2043–2050.

[39] O. Castillo, L. Cervantes, J. Soria, M. Sanchez, and J. R. Castro, "A generalized type-2 fuzzy granular approach with applications to aerospace," *Information Sciences*, vol. 354, pp. 165–177, 2016.

[40] M. A. Sanchez, O. Castillo, and J. R. Castro, "Information granule formation via the concept of uncertainty-based information with interval type-2 fuzzy sets representation and takagi–sugeno–kang consequents optimized with cuckoo search," *Applied Soft Computing*, vol. 27, pp. 602–609, 2015.

[41] L. Cervantes and O. Castillo, "Type-2 fuzzy logic aggregation of multiple fuzzy controllers for airplane flight control," *Information Sciences*, vol. 324, pp. 247–256, 2015.

[42] O. Castillo and P. Melin, "Intelligent systems with interval type-2 fuzzy logic," *International Journal of Innovative Computing, Information and Control*, vol. 4, no. 4, pp. 771–783, 2008.

[43] A. B. Cara, C. Wagner, H. Hagras, H. Pomares, and I. Rojas, "Multiobjective optimization and comparison of nonsingleton type-1 and singleton interval type-2 fuzzy logic systems," *IEEE Transactions on Fuzzy Systems*, vol. 21, no. 3, pp. 459–476, June 2013.

[44] F. Gaxiola, P. Melin, F. Valdez, O. Castillo, and J. R. Castro, "Comparison of t-norms and s-norms for interval type-2 fuzzy numbers in

weight adjustment for neural networks," *Information*, vol. 8, no. 3, p. 114, 2017.

[45] H. B. Sola, J. Fernandez, H. Hagras, F. Herrera, M. Pagola, and E. Barrenechea, "Interval type-2 fuzzy sets are generalization of interval-valued fuzzy sets: Toward a wider view on their relationship," *IEEE Transactions on Fuzzy Systems*, vol. 23, no. 5, pp. 1876–1882, Oct 2015.

[46] C. H. Lee, F. Y. Chang, and C. M. Lin, "An efficient interval Type-2 fuzzy cmac for chaos time-series prediction and synchronization," *IEEE Transactions on Cybernetics*, vol. 44, no. 3, pp. 329–341, March 2014.

[47] Y. Y. Lin, J. Y. Chang, N. R. Pal, and C. T. Lin, "A mutually recurrent interval Type-2 neural fuzzy system (MRIT2NFS) with self-evolving structure and parameters," *IEEE Transactions on Fuzzy Systems*, vol. 21, no. 3, pp. 492–509, June 2013.

[48] O. Castillo, L. Amador-Angulo, J. R. Castro, and M. Garcia-Valdez, "A comparative study of type-1 fuzzy logic systems, interval type-2 fuzzy logic systems and generalized type-2 fuzzy logic systems in control problems," *Information Sciences*, vol. 354, pp. 257–274, 2016.

[49] E. Ontiveros, P. Melin, and O. Castillo, "High order α-planes integration: A new approach to computational cost reduction of general type-2 fuzzy systems," *Engineering Applications of Artificial Intelligence*, vol. 74, pp. 186 – 197, 2018.

[50] M. de los Angeles Hernandez, P. Melin, G. M. Méndez, O. Castillo, and I. López-Juarez, "A hybrid learning method composed by the orthogonal least-squares and the back-propagation learning algorithms for interval a2-c1 type-1 non-singleton type-2 tsk fuzzy logic systems," *Soft Computing*, vol. 19, no. 3, pp. 661–678, Mar 2015.

[51] F. Chiclana and S.-M. Zhou, "Type-reduction of general Type-2 fuzzy sets: The Type-1 owa approach," *International Journal of Intelligent Systems*, vol. 28, no. 5, pp. 505–522, 2013.

[52] S. Greenfield and F. Chiclana, "Type-reduced set structure and the truncated type-2 fuzzy set," *Fuzzy Sets and Systems*, vol. 352, pp. 119 – 141, 2018, theme: Advances in Fuzzy Sets.

[53] K. Shiev, S. Ahmed, N. Shakev, and A. V. Topalov, *Trajectory Control of Manipulators Using an Adaptive Parametric Type-2 Fuzzy CMAC Friction and Disturbance Compensator*. Cham: Springer International Publishing, 2016, pp. 63–82.

[54] C. J. Kim and D. Chwa, "Obstacle avoidance method for wheeled mobile robots using interval type-2 fuzzy neural network," *IEEE Transactions on Fuzzy Systems*, vol. 23, no. 3, pp. 677–687, June 2015.

[55] M. Manceur, N. Essounbouli, and A. Hamzaoui, "Second-order sliding fuzzy interval type-2 control for an uncertain system with real application," *IEEE Transactions on Fuzzy Systems*, vol. 20, no. 2, pp. 262–275, April 2012.

[56] M. A. Sanchez, O. Castillo, and J. R. Castro, "Generalized type-2 fuzzy systems for controlling a mobile robot and a performance comparison with interval type-2 and type-1 fuzzy systems," *Expert Systems with Applications*, vol. 42, no. 14, pp. 5904–5914, 2015.

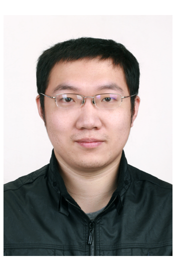
[57] F. Esteva and L. Godo, "Monoidal t-norm based logic: towards a logic for left-continuous t-norms," *Fuzzy sets and systems*, vol. 124, no. 3, pp. 271–288, 2001.

[58] N. N. Karnik and J. M. Mendel, "Centroid of a type-2 fuzzy set," *Information Sciences*, vol. 132, no. 1, pp. 195–220, 2001.

[59] "Robust adaptive backstepping control for a class of nonlinear systems using recurrent wavelet neural network," *Neurocomputing*, vol. 142, pp. 372 – 382, 2014.

[60] J. M. Mendel and F. Liu, "Super-exponential convergence of the karnik–mendel algorithms for computing the centroid of an interval type-2 fuzzy set," *IEEE Transactions on Fuzzy Systems*, vol. 15, no. 2, pp. 309–320, 2007.

[61] S. Blažič, "On periodic control laws for mobile robots," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 7, pp. 3660–3670, 2014.



Fei Chao (M'11) received the B.Sc. degree in Mechanical Engineering from the Fuzhou University, China, and the M.Sc. Degree with distinction in computer science from the University of Wales, Aberystwyth, U.K., in 2004 and 2005, respectively, and the Ph.D. degree in robotics from the Aberystwyty University, U.K. in 2009. He is currently an Associate Professor with the Cognitive Science Department, Xiamen University, China. His research interests include robotics and intelligent control.

Dajun Zhou received his BEng. and MEng. degree from the Cognitive Science Department, School of Informatics, Xiamen University, P. R. China, in 2015 and 2018 respectively. He continues his robotic research in the same department. His research interests include human-robot interactions, machine learning, and intelligent mobile robots.

Chih-Min Lin (M'87-SM'99-F'10) received the B.S. and M.S. degrees from Department of Control Engineering and the Ph.D. degree from Institute of Electronics Engineering, National Chiao Tung University, Hsinchu, Taiwan, in 1981, 1983 and 1986, respectively. He is currently a Chair Professor and the Vice President of Yuan Ze University, Chung-Li, Taiwan. His current research interests include fuzzy neural network, cerebellar model articulation controller, and intelligent control systems.

Longzhi Yang (M'12-SM'17) is the Director of Learning and Teaching and an Associate Professor at the Department of Computer and Information Sciences with Northumbria University, U.K. His research interests include computational intelligence, machine learning, big data, computer vision, intelligent control systems, and the applications of such techniques under real-world uncertain environment. He is the Founding Chair of the IEEE Special Interest Group on Big Data for Cyber Security and Privacy. He received the Best Student Paper Award at the 2010 IEEE International Conference on Fuzzy Systems.

Changle Zhou Changle Zhou received his PhD from Peking University in 1990. Currently, he is a professor of the Cognitive Science Department at the Xiamen University. His research interests lie in the areas of artificial intelligence. He also carries out research on a host of other topics including computational brain modeling, computational modeling of analogy and metaphor and creativity, computational musicology and information processing of data regarding traditional Chinese medicine.

Changjing Shang received a Ph.D. in computing and electrical engineering from Heriot-Watt University, UK. She is a University Research Fellow with the Department of Computer Science, Institute of Mathematics, Physics and Computer Science at Aberystwyth University, UK. Prior to joining Aberystwyth, she worked for Heriot-Watt, Loughborough and Glasgow Universities. Her research interests include pattern recognition, data mining and analysis, and space robotics.