Journal of Artificial Intelligence and Data Mining (JAIDM), Vol. 12, No. 4, 2024, 545-566.

Shahrood University of Technology

Journal of Artificial Intelligence and Data Mining (JAIDM)
Journal homepage: http://jad.shahroodut.ac.ir

Research paper

Stable Synchronization in Fuzzy Recurrent Neural Networks within a Fixed Time Frame

Farnaz Sabahi*

Department of Electrical Engineering, Faculty of Electrical and Computer Engineering, Urmia University, Urmia, Iran.

Article Info

Article History:
Received 29 November 2024
Revised 22 December 2024
Accepted 31 December 2024

DOI:10.22044/jadm.2024.14138.2647

Keywords:
Discontinuous Neural Networks, Fixed-time Synchronization, Lyapunov Function, Time-varying Delays.

**Corresponding author: f.sabahi@urmia.ac.ir (F. Sabahi).*

Abstract

This paper explores fixed-time synchronization for discontinuous fuzzy delay recurrent neural networks (DFRNNs) with time-varying delays. Based on a generalized variable transformation, the error system has been developed to effectively manage discontinuities in neural systems. This research addresses the fixed-time stability problem using a novel discontinuous state-feedback control input and a simple switching adaptive control scheme. The proposed method ensures robust synchronization of the drive and response neural systems within a fixed time. Practical applications of this work include improvements in protocols for secure communications, robotic control systems, and intelligent control frameworks over dynamic systems. A numerical example substantiates the theoretical claims, demonstrating the strengths of the proposed approach. The results show fixed-time convergence of error margins to zero, ensuring unbiased performance within a predefined timeframe, independent of initial conditions.

1. Introduction

Synchronization of two or more coupled systems is the phenomenon of showing an identical behavior in time. If the initial condition is quite large, then it results in an undesirable time for convergence. A continuous-time ordinary differential equation is commonly used to model synchronization, which is expressed as follows:

$$\dot{x}_i(t) = f\left(x_i(t)\right) + g_i\left(x(t)\right), \quad i = 1,\ldots,n \qquad (1)$$

In this equation, $x_i \in R^n$ denotes the status of the i th node, $f\left(x_i(t)\right)$ signifies the inherent behavior or natural tendencies of the i th node, and in the case where $f = 0$, the synchronization model transforms into the consensus model. The variable $x(t) = \left(x_1(t),\ldots,x_n(t)\right)$ and g_i denotes the diffusion-based interconnection or coupling from the neighboring nodes of node i. Despite each node only needing local information from its neighbors, the entire network can display collective behavior, specifically synchronization, such that $lim_{t\to\infty} \| x_i(t) - x_j(t) \| = 0$, where $\|.\|$ represents some norm. The synchronization model commonly used in literature is as follows:

$$\dot{x}_i(t) = f\left(x_i(t)\right) + c\sum a_{ij} A(t) \qquad (2)$$

where $A(t) = \left(x_j(t) - x_i(t)\right)$.

This is a common problem in system theory, especially in robotic systems: the design of controllers that drive a system to a desired position in finite time. Finite-time synchronization is particularly important in many applications because only exponential synchronization may require the existence of coupling mechanisms or external control systems for an infinite period of time. The attainment of finite-time synchronization can enhance the performance of a system by aiding in the rejection of disturbances and improving its ability to withstand uncertainties. The exploration of novel coupling protocols for achieving finite-time synchronization holds significance for both theoretical scrutiny and practical implementation. However, the settling time depends on the initial conditions of the system, which in turn limits the

practical applicability of finite-time synchronization since it may not be possible to know the initial conditions in advance. Besides, discontinuities in systems and NNs result in instability. To that end, Polyakov proposed fixed-time stability (FTS) to overcome the shortcomings of settling time in analyzing the stability of linear systems over a finite time [1]. FTS has found popularity in power systems, space technology, and many other fields. Works have been conducted on FTS of nonlinear systems, including discontinuous systems which are very common in neuroscience and engineering applications. In such systems, Lyapunov function techniques have emerged as an important approach to investigate FTS. Compared to that, the FTS has a strictly limited constraint on the settling time and guarantees convergence within the pre-defined limit of a settling time for any system initial conditions. The FTS has been actively investigated within the stochastic [1] and deterministic continuous-time systems [2], design of a fixed-time control [3]–[9], design of the fixed-time observers [10]–[14], and design of a fixed-time identification [15]–[19]. The scholars have also developed several approaches to address FTS in discontinuous systems using methods of Lyapunov function, such as the unified theorem proposed by Ji et al. to address the finite-time stability and FTS in networks that exhibit discontinuous activations [20], and studies on fixed-time stability of discontinuous NNs [21]. Several synchronization control techniques for fuzzy NNs have been suggested in the literature over these years. In [22], the paper has established new criteria on global exponential stability of equilibrium points and globally exponential lag synchronization in memristor-based FNNs, by using different mathematical techniques and numerical simulations.

It is worth pointing out that the existing methods indeed try to handle discontinuities with smooth approximations, yet they usually lack robustness against real-world time-varying delays. For the alternative approaches concerning this context, sliding mode control and adaptive control can be mentioned. In this sense, although the sliding mode control and adaptive techniques may work for some scenarios, they usually cannot resolve the challenges brought about by discontinuities and time-varying delays in neural systems. Our approach avoids these deficiencies by ensuring fixed-time convergence through a robust control design.

Recurrent neural networks are a class of NNs with feedback connections such that the network may keep past inputs in memory. The network output is influenced jointly by the current input and the past state of the network. Fixed time synchronization control for RNNs involves adjusting the weights of the connections between nodes to achieve a desired synchronization behavior while in view of the recurrent dynamics of the network. The updates in the hidden state, for every step in an RNN, are a function of the present input and previous hidden state; then the output is produced by an activation function such as sigmoid or softmax from the current hidden state.

DRNN is also a kind of NN, and it integrates the principle of fixed-length delay lines for processing the input data in a time-delayed manner. The input data is allowed to be stored in the delay line for a certain number of time steps before being relayed to the hidden layer of the network. Then, by using the time-delayed input data, the network produces the desired output. The advantage of the DRNN is that it is able to show dependencies within the input data concerning a standard RNN. While the DRNN is delaying the input data, then it can process information from earlier time steps, which may be useful in tasks like time series prediction. However, a DRNN is also more complex than the standard RNN since an extra part needs to be added to its architecture: the delay-line component. Moreover, a fixed-length delay may impose some limitation, because in most applications, not all relevant information of interest is really contained in the fixed length of the delay line of the input data.

The key differences between DRNN and delayed fuzzy RNN are how these two process inputs and what kind of activation functions are applied. A DRNN processes its input in a delayed manner using a fixed-length delay line. First, the input is fed through a delay line, which holds the input for a fixed number of time steps before feeding it to the network's hidden layer. It is in this hidden layer that the actual processing of the delayed input data computes the output. The activation function in DRNNs can be either linear or nonlinear, which can be sigmoid or ReLU. The delay dependent RNN dynamic equation is generally defined as:

$$y_i(t) = f_i\left(x_i(t)\right) \tag{3}$$

$$\dot{x}_i(t) = -\sum_{j=1}^{n} a_{ij}\left(w_{ij}x_j\left(t-\tau_{ij}\right)-\theta_{ij}\right)$$

$$+\sum_{j=1}^{n} b_{ij}y_j\left(t-\tau_{ij}\right)$$

where $x_i(t)$ represents the statue of node i at t, $y_i(t)$ is the node-output of i at t, a_{ij} and b_{ij} are weighting coefficients, w_{ij} is the connection strength between node i and j, θ_{ij} is the threshold value, $f_i(\cdot)$ is the discontinuous activation function of node i, τ_{ij} is the time delay between node i and j.

By integrating fuzzy logic into both the architecture and the form of RNN's activation function, we achieve DFRNN. Neurons in DFRNN bear a more complicated architecture due to the addition of such elements as self-feedback terms, fuzzy weights, fuzzy operators, and external inputs. The activation function of DFRNN is more complex by combining nonlinear functions and fuzzy operators. The most significant benefit of a DFRNN is that it possesses an elaborate architecture and an activation function which may be more appropriate to capture the complicated patterns and relationships present in the input data than a DRNN. On the other hand, the DFRNN is computational and challenging with respect to the DRNN. Another difference between the DRNN and DFRNN lies in dealing with the input data and the applied activation functions.

While DRNN uses a fixed-length delay line to process the input data and can have a linear or nonlinear activation function, DFRNN incorporates the fuzzy logics in architecture and activation function for capturing complex pattern and relationship in input data. Due to the discontinuous activation function in DFRNN, the network becomes unstable. Thus, stability and synchronization issues are an important field of study in discontinuous dynamical networks. The paper deals with discontinuities in neural systems. In this regard, we use a generalized variable transformation to circumvent the discontinuity but obtain an error system. In this paper, the FTS issue of the error system generated in the drive and response systems is investigated. To be specific, a new form of discontinuous control input is introduced for the response neural system. A switching state-feedback control law is designed. The proposed approach ensures stable synchronization and allows one to easily estimate the settling time.

It should be noted that while generalized variable transformations and discontinuous state-feedback control have been explored in related works, their application in addressing the challenges of fixed-time synchronization for discontinuous systems with time-varying delays is unique to this study. In this context, FRNN refers to discontinuous fuzzy recurrent neural networks with time-varying delays. These networks encompass fuzzy logic in their architecture.

The proposed approach offers a robust and innovative approach to synchronization, addressing several key challenges in neural networks. Unlike many of the existing methods, which emphasize asymptotic or exponential convergence, the proposed approach guarantees synchronization within a predefined fixed time, irrespective of the initial conditions. This allows for strong guarantees on settling time, which is quite critical for applications that are time-sensitive. In addition, through a generalized variable transformation, the approach provides a generalized treatment of system discontinuities and develops an effective continuous Lyapunov-based framework which is hard for traditional methods to achieve, such as sliding mode control or adaptive control. Due to the inclusion of an adaptive switching control scheme in this paper, the designed controller handles time-varying delays' uncertainties and nonlinearity, assuring synchronization performance without asking for an exact delay compensation– a limitation required by most of the existing literature. The discontinuous state-feedback control and the adaptive switching mechanism are easy to implement, hence less computation-intensive with respect to methods in adaptive or observer-based controls. Another key feature of the proposed approach is that it is not constrained on a particular class of networks but, as a matter of fact, fits a wide class of discontinuous and delayed fuzzy recurrent neural networks, hence it becomes more versatile and applicable in various real-world application. Table 1 compares the strengths and weaknesses of the proposed approach with the important existing approaches.

The proposed approach can find quite broad applications in the following fields. Robotic systems need enhancing synchronization to produce accurate and coordinated movements or execution of tasks in areas like assembly lines and autonomous navigation. Another application is *Secure communication protocols that need* enhancing the synchronization of the data being transmitted that resist timing attacks. *Intelligent Control Systems need* ensuring stable synchronization in multi-agent systems for industrial automation and networked control. In addition, in Signal Processing, it is important ensure synchronous sampling to correctly represent and process a signal.

Table 1. Comparison with existing approach.

Synchronization Method	Strengths	Weaknesses	Comparison with Proposed Approach
Exponential and Asymptotic	- Ensures synchronization error decays exponentially or asymptotically. - Suitable for systems with continuous dynamics and smooth nonlinearities.	- Requires longer convergence times. - Sensitive to initial conditions. - May not guarantee synchronization within a predefined timeframe.	- **Advantage:** Proposed approach guarantees fixed-time convergence, providing clear timeframes for synchronization.
Finite-Time	- Ensures synchronization within a finite time. - Suitable for applications requiring quick convergence.	- Settling time depends on initial conditions. - Complex controller design and parameter tuning.	- **Advantage:** Proposed method guarantees fixed-time synchronization regardless of initial conditions.
Adaptive	- Handles system uncertainties and parameter variations. - Real-time adjustment of control parameters enhances robustness.	- Computationally intensive. - May not guarantee convergence within a fixed timeframe, especially under significant delays or discontinuities.	- **Advantage:** Simple switching adaptive control reduces computational complexity while ensuring fixed-time synchronization.
Sliding Mode Control	- Highly robust to system uncertainties and disturbances. - Effective for systems with discontinuous dynamics.	- Can cause chattering. - Requires precise knowledge of system bounds. - Challenging to implement in systems with significant delays.	- **Advantage:** Avoids chattering and effectively handles discontinuities and time-varying delays.
Proposed Fixed-Time Synchronization in DFRNNs	- Guarantees fixed-time synchronization regardless of initial conditions. - Handles discontinuities and time-varying delays effectively. - Simple switching adaptive control is easy to implement. - Applicable to a wide range of networks. - Reduces computational complexity.	- Complexity increases with the intricacy of fuzzy logic and network structures. - Requires careful design of transformations and control inputs.	

We can summarize the main findings of this paper as follows:

1. Development of a novel approach for gaining fixed-time synchronization in DFRNN.
2. Establishment of Lyapunov circumstances concerning FST in DFRNN.
3. Consistency as a design principle in the control system to ensure that the tracking, after adjustment, will actually be consistent with the intended trajectory.
4. Fixed upper bounds derivation concerning settling time of DFRNN.
5. Contribution to the synchronization of NNs and, equally importantly, insights into stability analysis of discontinuous systems.
6. The main design principle of consistency provides an adaptive and responsive control system.

The paper is organized as follows. A review of preliminary concepts is provided in Section 2. In Section 3, we will introduce a control design scheme to ensure fixed-time robust synchronization with consideration of time delay. In Section 4, we provide some simulations to verify the proposed control. Finally, Section 5 concludes the paper.

2. System Description

A DRNN's response may be modeled by a set of dynamical equations which, in summary, reflect the way in which the hidden state of the network varies in time.

Discontinuous neural networks and systems with time-varying delays are great challenges to stability and synchronization. Discontinuities usually give rise to abrupt changes in dynamics, which make the analysis of stability and control design much more complicated. Time-varying delays introduce additional nonlinearity and uncertainty, making it hard to predict and manage system behavior. Delays can also add up over time, increasing instability and making it difficult to maintain synchronization.

Let $\mathbf{h}_t \in \mathrm{R}^n$ denote the hidden state of the DRNN at t, where n is the number of neurons. Then, the dynamical equation for the DRNN can be written as:

$$\mathbf{h}_t = f\left(\mathbf{W}\mathbf{x}_t + \mathbf{U}\mathbf{h}_{t-1} + \mathbf{b}\right) \tag{4}$$

where $\mathbf{x}_t \in \mathrm{R}^m$ is the input vector at t, $\mathbf{W} \in \mathrm{R}^{n\times m}$ is the weight matrix that maps the input to the hidden state, $\mathbf{U} \in \mathrm{R}^{n\times n}$ is the weight matrix that maps the previous hidden state to the current hidden state, $\mathbf{b} \in \mathrm{R}^n$ is the bias vector, and $f(\cdot)$ is the activation function, such as sigmoid or ReLU.

In a DRNN, the input vector $\mathbf{x}_t$ is typically delayed by a fixed number of time steps before being fed into the network's hidden layer. This

delay can be implemented using a shift register or a delay line in the input layer. Let τ denote the delay in time steps, then the input vector at t is given by $\mathbf{x}_{t-\tau}$.

Therefore, the dynamical equation for a DRNN with input delay can be written as:

$$\mathbf{h}_t = f\left(\mathbf{W}\mathbf{x}_{t-\tau} + \mathbf{U}\mathbf{h}_{t-1} + \mathbf{b}\right) \quad (5)$$

Note that the delay in the input forces the network to process information of earlier time steps, which may be useful when modeling long-term dependencies in an input. The drive-response synchronization is applied to a DRNN by constructing a response system that synchronizes with the DRNN dynamics. The output of such a system can be described by the following formula:

$$\mathbf{y}_t = \mathbf{W}_y \mathbf{h}_{t-\tau_y} + \mathbf{b}_y, \quad (6)$$

where $\mathbf{y}_t \in \mathrm{R}^m$ is the output of the response system at t, $\mathbf{W}_y \in \mathrm{R}^{m\times n}$ is the weight matrix that maps the delayed hidden state of the DRNN to the output of the response system, $\mathbf{b}_y \in \mathrm{R}^m$ is the bias vector of the response system, and τ_y is the delay in the system.

The response system's output gets computed by the sum of a product between the weights in $\mathbf{W}_y$ and the DRNN's delayed hidden state, where the weights are to be learned during training. This delay makes the response system to get information from earlier time lags that is useful to model long-term dependencies on the DRNN dynamics. We can synchronize the response system with the DRNN by minimizing the error by adjusting the weights in $\mathbf{W}_y$ and the delay τ_y.

Let $\mathbf{x}(t) = [x_1(t), x_2(t), \ldots, x_n(t)]^T$ be the state vector, $\mathbf{w}(t) = \left[w_{ij}(t)\right]$ be the time-varying weighted matrix, $\mathbf{f}(t) = [f_1(x_1(t-\tau_1(t))), f_2(x_2(t-\tau_2(t))), \ldots, f_n(x_n(t-\tau_n(t)))]^T$ be the time-delayed activation function vector, $\mathbf{v} = \left[v_{ij}\right]$ be the coupling matrix, $\mathbf{T}(t) = \left[T_{ij}(t)\right]$ be the time-varying transmission delay matrix, $\boldsymbol{\alpha}(t) = \left[\alpha_{ij}(t)\right]$ and $\boldsymbol{\beta}(t) = \left[\beta_{ij}(t)\right]$ be the time-varying feedback and feedforward matrices respectively, and $\mathbf{b} = [b_1, b_2, \ldots, b_n]^T$ be the bias vector. Then, we can write the DRNN dynamic equation in matrix form as:

$$\begin{aligned}\dot{\mathbf{x}}(t) &= \sigma(\mathbf{w}(t)\mathbf{f}(t-\boldsymbol{\tau}(t)) + \mathbf{v}\mathbf{x}(t) \\ &+ \mathbf{T}(t)\wedge\mathbf{f}(t-\boldsymbol{\tau}(t)) + \boldsymbol{\alpha}(t)\mathbf{f}(t-\boldsymbol{\tau}(t)) \\ &+ \boldsymbol{\beta}(t)\mathbf{f}(t-\boldsymbol{\tau}(t)) + \mathbf{b}) - \mathbf{x}(t)\end{aligned} \quad (7)$$

where σ is a vectorized activation function that applies the activation function element-wise to the vectorized input, $\boldsymbol{\tau}(t)$ is the vector of time delays for each neuron.

$$\wedge\mathbf{f}(t-\boldsymbol{\tau}(t)) = [\min\left\{\begin{matrix} f(x_1(t-\tau_1(t))), \\ \ldots, f(x_n(t-\tau_n(t))) \end{matrix}\right\}]^T \quad (8)$$

$$\begin{aligned}\dot{y}(t) &= \sigma(\mathbf{w}(t)\mathbf{f}(t-\boldsymbol{\tau}(t)) + \mathbf{v}y(t) \\ &+ \mathbf{T}(t)\wedge\mathbf{v}y(t) + \boldsymbol{\alpha}(t)\mathbf{f}(t-\boldsymbol{\tau}(t)) \\ &+ \boldsymbol{\beta}(t)\mathbf{f}(t-\boldsymbol{\tau}(t)) + \mathbf{b}) - y(t)\end{aligned}$$

The DRNN response equation can be:

$$\begin{aligned}\dot{y}(t) &= \sigma(\mathbf{w}(t)\mathbf{f}(t-\boldsymbol{\tau}(t)) + \mathbf{v}y(t) \\ &+ \mathbf{T}(t)\wedge\mathbf{v}y(t) + \boldsymbol{\alpha}(t)\mathbf{f}(t-\boldsymbol{\tau}(t)) \\ &+ \boldsymbol{\beta}(t)\mathbf{f}(t-\boldsymbol{\tau}(t)) + \mathbf{b}) - y(t)\end{aligned} \quad (9)$$

This is an equation representing a response system, which is supposed to synchronize with the DRNN by mapping its hidden state, delayed by the means of a learned weight matrix and delay, to an output. Given the dynamics of the *ith* neuron in a DRNN, whose current state will be denoted as $y_i(t)$. This equation has a number of terms that capture the effects of input to the neuron, recurrent connections, and bias. The activation function σ_i may be chosen depending on the problem at hand. Besides, we can introduce fuzzy logic into the model and rewrite the equation in a bit different form. Let $S(t) = \left[S_{ij}(t)\right]$ be the time-varying fuzzy OR operator matrix, and $I(t)$ be the external input vector. Let $\mathbf{d}(x(t))$ be the self-feedback vector, and $\mathbf{c}(t, x(t)) = [c_1(t, x_1(t)), c_2(t, x_2(t)), \ldots, c_n(t, x_n(t))]^T$ be the discontinuous threshold behavior vector.

Then, we can write DFRNN with discontinuous activation in matrix form as:

$$\begin{aligned}&\dot{\mathbf{x}}(t) = \\ &\mathbf{d}(\mathbf{x}(t))[-\mathbf{c}(t, \mathbf{x}(t)) + \mathbf{w}(t)\mathbf{f}(t-\boldsymbol{\tau}(t)) + \\ &\mathbf{v}\mathbf{y}(t) + \mathbf{T}(t)\wedge\mathbf{v}\mathbf{y}(t) + \boldsymbol{\alpha}(t)\mathbf{f}(t-\boldsymbol{\tau}(t)) \\ &+ \boldsymbol{\beta}(t)\mathbf{f}(t-\boldsymbol{\tau}(t)) + \mathbf{S}(t)\vee\mathbf{v}\mathbf{y}(t) + \mathbf{I}(t)]\end{aligned} \quad (10)$$

The fuzzy AND and fuzzy OR operators can also be written as:

$$\mathbf{T}(t) \wedge \mathbf{vy}(t) = [T_{ij}(t) \wedge v_j y_j(t)]_{i=1,\ldots,n} \quad (11)$$
$$\mathbf{S}(t) \vee \mathbf{vy}(t) = [S_{ij}(t) \vee v_j y_j(t)]_{i=1,\ldots,n}.$$

where $\wedge$ and $\vee$ denote the element-wise minimum and maximum operations, respectively, and $y_j(t)$ is the j-th element of the output vector $\mathbf{y}(t)$.

The initial condition for the DFRNN can be expressed as:

$$\mathbf{x}_{i0}(t) = \mathbf{\Phi}_i(t), \quad t \in [-\tau_i, 0], \quad (12)$$

where $\Phi_i(t)$ specifies the initial state of the i-th neuron at a time t in the past that is within the delay range of the network.

Drawing upon the principle of drive-response synchronization, we can consider DFRNN equation (10) as the drive system and design a response system to synchronize with it. The response of it can be described by:

$$\dot{\mathbf{y}}(t) = \mathbf{d}(\mathbf{y}(t))[-\mathbf{c}(t, \mathbf{y}(t)) + \mathbf{W}(t)\mathbf{\Phi}(t - \boldsymbol{\tau}(t)) + \mathbf{V}\mathbf{y}(t) + \quad (13)$$
$$\mathbf{T}(t) \wedge \mathbf{V} + \boldsymbol{\alpha}(t) \square\, \mathbf{f}(\mathbf{y}(t - \boldsymbol{\tau}(t))) + \boldsymbol{\beta}(t) \square\, \mathbf{f}(\mathbf{y}(t - \boldsymbol{\tau}(t))) + \mathbf{S}(t) \vee \mathbf{V} + \mathbf{I}(t)] + \mathbf{u}(t)$$

where $\mathbf{y}(t)$ is the state vector of the neurons in the network, $\mathbf{d}(\mathbf{y}(t))$ is a diagonal matrix function of the rate of change of each component of $\mathbf{y}(t)$, $\mathbf{c}(t, \mathbf{y}(t))$ is a diagonal matrix function representing the "leakage" or the tendency of the neuron to return to its resting state, $\mathbf{W}(t)$ is a matrix function characterizing the strength of the connections from all neurons to the neurons at t, $\mathbf{\Phi}(t - \boldsymbol{\tau}(t))$ is a matrix function characterizing the delayed states of all neurons at $t - \boldsymbol{\tau}(t)$, $\mathbf{V}$ is a matrix of weights that represent the connections from all neurons to the i-th neuron in the absence of time delay, $\mathbf{T}(t)$ is a matrix function characterizing the time-delayed weights of the inputs to the neurons from all neurons, $\wedge\, \mathbf{V}$ is a row vector that includes the minimum value of the weighted inputs from all neurons, $\boldsymbol{\alpha}(t)$ and $\boldsymbol{\beta}(t)$ are row vectors of weights that represent the time-delayed connections from all neurons with an element-wise product operator $\square$, $\mathbf{f}(\mathbf{y}(t - \boldsymbol{\tau}(t)))$ is a matrix function demonstrating the activation function outputs of all neurons in the network with a time delay of $\boldsymbol{\tau}(t)$, $\mathbf{S}(t)$ is a matrix function representing the time-delayed weights of the inputs to the i-th neuron from all neurons, $\mathbf{I}(t)$ is a row vector that includes the external inputs, and $\mathbf{u}(t)$ is a row vector that contains the noise or disturbance inputs at time t.

The input variable $u_i(t)$ will be determined at a later stage for each node i in the system. The functions $f_i : \mathrm{R} \to \mathrm{R}$ are piecewise continuous, which means that they are continuous on most points in their domain. However, there are a finite number of isolated points $\{\rho_{i,k}\}$ where f_i has a discontinuity. At these points, f_i has finite right $f_i^+(\rho_{i,k})$ and left $f_i^-(\rho_{i,k})$ limits. In addition, f_i exhibits only a finite number of points of discontinuity within any interval that is both closed and bounded in R.

The expression for the initial circumstance of the response system is given by:

$$y_{i0}(t) = \phi_i(t), \quad t \in [-\tau_i, 0], \quad (14)$$

where $\phi_i(t)$ specifies the initial state of the i-th neuron in the response system at a time t in the past that is within the delay range of the network.

To analyze the system's robustness, we define the disturbance magnitude d_{max} as the maximum external perturbation tolerated by the system. The maximum input magnitude, u_{max}, represents the upper limit of the control input based on system design. The robustness ratio $\gamma = \frac{d_{max}}{u_{max}}$ can be defined to compare the effects of disturbances relative to input capabilities. This ratio is crucial for ensuring synchronization stability under varying external conditions.

3. Fixed-time Robust Synchronization with a Discontinuous Controller

In this section, a control design scheme is given to guarantee fixed-time robust synchronization with consideration of time delay. We will design a simple switching adaptive control scheme to cope with discontinuities. A novel simple switching adaptive control technique is designed in this

approach to achieve fixed-time synchronization for DFRNN with time-varying delays. In the proposed scheme, a state-feedback control law with the mechanism of switching using a sign function is designed in order to handle such discontinuities effectively. Accordingly, the control input to the response system can be derived as $u(t) = -k(t) \cdot sgn(e(t)) + \theta(t) \cdot sgn(e(t) - e(t-\tau))$, where $e(t)$ represents the error between drive-response systems, τ stands for the time delay, and $k(t)$ and $\theta(t)$ are the adaptive gains which are time varying. In particular, the switching mechanism in this system adaptively switches the state of the control input so as to realize quick responses of the control system with respect to a change of the error signal and synchronization.

In the following, we derive the proposed Lyapunov conditions and state the assumptions used in these conditions. The Lyapunov function $V(t) = \sum_{i=1}^{n} |e_i(t)|$ is chosen due to its positive definiteness and regularity properties, where $e_i(t)$ represents the error of the i-th neuron at time t. The system dynamics are governed by piecewise continuous functions with finite discontinuities. The control input $u_i(t)$ is designed to ensure boundedness and convergence. The initial conditions $x_i(0)$ are within the specified range.

To consider fixed-time stability, we first introduce the following lemmas:

[Lemma[23]] If a regular, radially unbounded, and positive definite function $V(x(t)): \mathrm{R}^n \to \mathrm{R}$ exists, in such a way that the inequality

$$\frac{dV(x(t))}{dt} \le -aV^{\delta}(x(t)) - bV^{\theta}(x(t)), \quad x(t) \in \mathrm{R}^n \setminus \{0\},$$

is satisfied by any solution $x(t)$ of the system, where $a, b > 0$, $0 \le \theta < 1$, and $\delta > 1$, then the system's origin is FTS, and $T(x_0)$ that is the settling time function can be approximated by:

$$T(x_0) \le T_{\max} = \frac{1}{b}\left(\frac{b}{a}\right)^{\frac{1}{1-\theta}} \left(1 - \frac{1}{\theta - 1}\right)\left(1 - \frac{1}{\delta - 1}\right). \tag{15}$$

[Lemma Gronwall-Bellman [24]] Let $f(t)$ be a non-negative function that satisfies the differential inequality

$$\frac{d}{dt} f(t) \le \alpha(t) f(t) + \beta(t), \tag{16}$$

where $\alpha(t)$ and $\beta(t)$ are non-negative continuous functions. Then, for any $t \ge 0$, we have:

$$f(t) \le f(0)\exp\left(\int_0^t \alpha(s)ds\right) + \int_0^t \beta(s)\exp\left(\int_s^t \alpha(r)dr\right)ds \tag{17}$$

The above lemma is a very strong tool for establishing the upper bounds on the trajectories of non-negative functions which satisfy certain differential or difference inequalities. This lemma finds widespread application in the study of stability and convergence properties of dynamical systems, and its applications are found in many areas of control theory, optimal control, and so on. In computational neuroscience, DFRNN is an extended model of the conventional RNN, which includes the ideas of time delays, discontinuous activation functions, and fuzzy logic [25, 26]. Fixed-time synchronization means that both the response and drive systems can achieve synchronization within a fixed time, regardless of any initial condition or external disturbance. On the other hand, fixed-time synchronization differs from asymptotic synchronization. Suppose $e(t)$ is a solution of the error system over the interval $[0, T)$. The error system can be expressed as:

$$\begin{aligned} \dot{e}_i(t) = & -c_i(t, x_i(t)) + f_i(x_i(t)) \\ & - \sum_{j=1}^{n} w_{ij}(t) f_j(x_j(t - \tau_{ij}(t))) \\ & - \sum_{j=1}^{m} v_{ij} x_j(t) - \bigwedge_{j=1}^{n} T_{ij}(t) v_j - \\ & \bigwedge_{j=1}^{n} \alpha_{ij}(t) f_j(x_j(t - \tau_j(t))) - \\ & \bigvee_{j=1}^{n} \beta_{ij}(t) f_j(x_j(t - \tau_j(t))) - \\ & \bigvee_{j=1}^{n} S_{ij}(t) v_j + u_i(t) - y_i(t) \end{aligned} \tag{18}$$

where $e_i(t)$ represents the error of the i-th neuron at time t, $x_i(t)$ is the state of the i-th

neuron, and $y_i(t)$ is the actual output of the i -th neuron at time t . The other terms in Equation (18) have the same meanings as in Equation (10). The term $u_i(t)$ represents any external input to the i -th neuron, and the subtraction of $y_i(t)$ from the right-hand side of Equation (18) ensures that the error dynamics are calculated with respect to the desired output of the neuron. The error dynamics of the neuron detail how the deviation between the anticipated and actual output of the neuron varies over time. By considering this information, we can fine-tune the weights and parameters of the network to reduce the error and accomplish the intended task or behavior.

We present a potential Lyapunov function as:

$$V(t)=\sum_{i=1}^{n}\left|e_i(t)\right|. \tag{19}$$

The function $V(t)$ is C -regular. To compute the time derivative of the candidate Lyapunov function $V(t)$ given by Equation (19) moving in the direction of the error dynamics trajectory, we differentiate $V(t)$ with respect to time t using the chain rule:

$$\begin{aligned}
&\frac{dV(t)}{dt}=\sum_{i=1}^{n}\frac{d}{dt}\left|e_i(t)\right| \\
&=\sum_{i=1}^{n}\operatorname{sgn}\left(e_i(t)\right)\frac{d}{dt}e_i(t) \\
&=-\sum_{i=1}^{n}\operatorname{sgn}\left(e_i(t)\right)\begin{bmatrix}c_i\left(t,h_{-i}^{1}\left(w_i(t)\right)\right)- \\ c_i\left(t,h_{-i}^{1}\left(z_i(t)\right)\right)\end{bmatrix} \\
&+\sum_{i=1}^{n}\sum_{j=1}^{m}a_{ij}(t)\left[\operatorname{sgn}\left(e_i(t)\right)-\operatorname{sgn}\left(e_j(t)\right)\right] \\
&+\sum_{i=1}^{n}\sum_{j=1}^{m}a_{ij}(t)\left[\operatorname{sgn}\left(e_i\left(t-\tau_j(t)\right)\right)-\operatorname{sgn}\left(e_j(t)\right)\right] \\
&+\sum_{i=1}^{n}\sum_{j=1}^{n}\beta_{ij}(t)\begin{bmatrix}sgn\left(e_i\left(t-\tau_j(t)\right)\right)- \\ sgn\left(e_j\left(t-\tau_j(t)\right)\right)\end{bmatrix} \\
&+\sum_{i=1}^{n}\operatorname{sgn}\left(u_i(t)\right)d_i'\left(h_{-i}^{1}\left(w_i(t)\right)\right)\frac{d}{dt}h_{-i}^{1}\left(w_i(t)\right) \\
&-\sum_{i=1}^{n}\operatorname{sgn}\left(y_i(t)\right)d_i'\left(h_{-i}^{1}\left(w_i(t)\right)\right)\frac{d}{dt}h_{-i}^{1}\left(w_i(t)\right) \\
&+\sum_{i=1}^{n}\left|u_i(t)\right|\frac{d}{dt}d_i\left(h_{-i}^{1}\left(w_i(t)\right)\right) \\
&-\sum_{i=1}^{n}\left|y_i(t)\right|\frac{d}{dt}d_i\left(h_{-i}^{1}\left(w_i(t)\right)\right),
\end{aligned} \tag{20}$$

where $\operatorname{sgn}(x)$ is the sign function, $h_{-i}^{1}(w_i(t))$ is the vector of states of the network excluding the state of the i-th neuron at time t, $z_i(t)$ is the vector of delayed states of the network that contribute to the coupling of the i-th neuron at time t, and $d_i(\cdot)$ is the derivative of the activation function $d_i(\cdot)$. To calculate the derivative of $V(t)$ along the trajectories of the error dynamics for the delayed fuzzy recurrent neural network (DFRNN) given by Equation (20), we can use the chain rule. Specifically, we have:

$$\begin{aligned}
&\dot{V}(t)=\frac{d}{dt}\left(\sum_{i=1}^{n}\left|e_i(t)\right|\right) \\
&=\sum_{i=1}^{n}\frac{d}{dt}\left|e_i(t)\right| \\
&=\sum_{i=1}^{n}\frac{e_i(t)}{\left|e_i(t)\right|}\cdot\frac{d}{dt}e_i(t) \\
&=\sum_{i=1}^{n}\operatorname{sgn}\left(e_i(t)\right)\cdot\dot{e}_i(t) \\
&=-\sum_{i=1}^{n}\operatorname{sgn}\left(e_i(t)\right)\cdot c_i\left(t,x_i(t)\right) \\
&+\sum_{i=1}^{n}\operatorname{sgn}\left(e_i(t)\right)\cdot f_i\left(x_i(t)\right)- \\
&\sum_{i=1}^{n}\sum_{j=1}^{m}\operatorname{sgn}\left(e_i(t)\right)\cdot w_{ij}(t)\cdot f_j\left(x_j\left(t-\tau_{ij}(t)\right)\right) \\
&-\sum_{i=1}^{n}\sum_{j=1}^{m}\operatorname{sgn}\left(e_i(t)\right)\cdot v_{ij}\cdot x_j(t) \\
&-\sum_{i=1}^{n}\operatorname{sgn}\left(e_i(t)\right)\cdot\bigwedge_{j=1}^{n}T_{ij}(t)v_j- \\
&\sum_{i=1}^{n}\operatorname{sgn}\left(e_i(t)\right)\cdot\bigwedge_{j=1}^{n}\alpha_{ij}(t)f_j\left(x_j\left(t-\tau_j(t)\right)\right) \\
&-\sum_{i=1}^{n}\operatorname{sgn}\left(e_i(t)\right)\cdot\bigvee_{j=1}^{n}\beta_{ij}(t)f_j\left(x_j\left(t-\tau_j(t)\right)\right) \\
&-\sum_{i=1}^{n}\operatorname{sgn}\left(e_i(t)\right)\cdot\bigvee_{j=1}^{n}S_{ij}(t)v_j+ \\
&\sum_{i=1}^{n}\operatorname{sgn}\left(e_i(t)\right)\cdot u_i(t)+ \\
&-\sum_{i=1}^{n}\operatorname{sgn}\left(e_i(t)\right)\cdot y_i(t),
\end{aligned} \tag{21}$$

Let us now show that the time derivative of $V(e(t))$ satisfies the inequality in Lemma Gronwall-

Bellman. Note that the inequality, we need to prove is as follows:

$$\frac{dV\left(e\left(t\right)\right)}{dt} \le -aV^{\delta}\left(e\left(t\right)\right) - bV^{\theta}\left(e\left(t\right)\right), \quad e\left(t\right) \in \mathrm{R}^{n} \setminus \{0\}, \tag{22}$$

where $a,b > 0$ and $0 \le \theta < 1$, $\delta > 1$.

Substituting the expression for $\dot{V}\left(t\right)$ that we obtained earlier, we have:

$$\frac{dV\left(e\left(t\right)\right)}{dt} = \tag{23}$$

$$-\sum_{i=1}^{n} c_i\left(t\ x_i\left(t\right)\right)\cdot \operatorname{sgn}\left(e_i\left(t\right)\right) + \sum_{i=1}^{n} f_i\left(x_i\left(t\right)\right)\cdot \operatorname{sgn}\left(e_i\left(t\right)\right) - \sum_{i=1}^{n}\sum_{j=1}^{m} w_{ij}\left(t\right)\cdot f_j\left(x_j\left(t-\tau_{ij}\left(t\right)\right)\right)\cdot \operatorname{sgn}\left(e_i\left(t\right)\right) - \sum_{i=1}^{n}\sum_{j=1}^{m} v_{ij}\cdot x_j\left(t\right)\cdot \operatorname{sgn}\left(e_i\left(t\right)\right) - \bigwedge_{j=1}^{n} T_{ij}\left(t\right)v_j \cdot \sum_{i=1}^{n} \operatorname{sgn}\left(e_i\left(t\right)\right) - \bigwedge_{j=1}^{n} \alpha_{ij}\left(t\right) f_j\left(x_j\left(t-\tau_j\left(t\right)\right)\right)\cdot \sum_{i=1}^{n} \operatorname{sgn}\left(e_i\left(t\right)\right) - \bigvee_{j=1}^{n} \beta_{ij}\left(t\right) f_j\left(x_j\left(t-\tau_j\left(t\right)\right)\right)\cdot \sum_{i=1}^{n} \operatorname{sgn}\left(e_i\left(t\right)\right) - \bigvee_{j=1}^{n} S_{ij}\left(t\right)v_j \cdot \sum_{i=1}^{n} \operatorname{sgn}\left(e_i\left(t\right)\right) + \sum_{i=1}^{n} u_i\left(t\right).\sum_{i=1}^{n} \operatorname{sgn}\left(e_i\left(t\right)\right) - \sum_{i=1}^{n} y_i\left(t\right).\sum_{i=1}^{n} \operatorname{sgn}\left(e_i\left(t\right)\right).$$

Based on the triangle inequality, we can bound the absolute value of each term in the above expression as follows:

$$\left|-c_i\left(t, x_i\left(t\right)\right)\cdot \operatorname{sgn}\left(e_i\left(t\right)\right)\right| \le c_i\left(t, x_i\left(t\right)\right), \tag{24}$$

$$\left|f_i\left(x_i\left(t\right)\right)\cdot \operatorname{sgn}\left(e_i\left(t\right)\right)\right| \le f_i\left(x_i\left(t\right)\right),$$

$$\left|w_{ij}\left(t\right)\cdot f_j\left(x_j\left(t-\tau_{ij}\left(t\right)\right)\right)\cdot \operatorname{sgn}\left(e_i\left(t\right)\right)\right| \le w_{ij}\left(t\right)\cdot f_j\left(x_j\left(t-\tau_{ij}\left(t\right)\right)\right),$$

$$\left|v_{ij}\cdot x_j\left(t\right)\cdot \operatorname{sgn}\left(e_i\left(t\right)\right)\right| \le v_{ij}\cdot\left|x_j\left(t\right)\right|,$$

$$\left|T_{ij}\left(t\right)v_j\right|\cdot\sum_{i=1}^{n}\operatorname{sgn}\left(e_i\left(t\right)\right)\le\left|T_{ij}\left(t\right)v_j\right|\cdot\sum_{i=1}^{n}\left|e_i\left(t\right)\right|,$$

$$\left|\alpha_{ij}\left(t\right)f_j\left(x_j\left(t-\tau_j\left(t\right)\right)\right)\cdot\sum_{i=1}^{n}\operatorname{sgn}\left(e_i\left(t\right)\right)\right| \le \alpha_{ij}\left(t\right)\cdot f_j\left(x_j\left(t-\tau_j\left(t\right)\right)\right)\sum_{i=1}^{n}\left|e_i\left(t\right)\right|\cdot$$

$$\left|\beta_{ij}\left(t\right)f_j\left(x_j\left(t-\tau_j\left(t\right)\right)\right)\cdot\sum_{i=1}^{n}\operatorname{sgn}\left(e_i\left(t\right)\right)\right| \le \beta_{ij}\left(t\right)\cdot f_j\left(x_j\left(t-\tau_j\left(t\right)\right)\right)\cdot\sum_{i=1}^{n}\left|e_i\left(t\right)\right|,$$

$$\left|S_{ij}\left(t\right)v_j\cdot\sum_{i=1}^{n}\operatorname{sgn}\left(e_i\left(t\right)\right)\right|\le\left|S_{ij}\left(t\right)v_j\right|\cdot\sum_{i=1}^{n}\left|e_i\left(t\right)\right|$$

$$\left|u_i\left(t\right)\cdot\sum_{i=1}^{n}\operatorname{sgn}\left(e_i\left(t\right)\right)\right|\le\left|u_i\left(t\right)\right|\cdot\sum_{i=1}^{n}\left|e_i\left(t\right)\right|$$

$$\left|y_i\left(t\right)\cdot\sum_{i=1}^{n}\operatorname{sgn}\left(e_i\left(t\right)\right)\right|\le\left|y_i\left(t\right)\right|\cdot\sum_{i=1}^{n}\left|e_i\left(t\right)\right|$$

Using these bounds and $\left|e_i\left(t\right)\right| \le V^{\frac{1}{\delta}}\left(e\left(t\right)\right)$, we can write:

$$\frac{dV\left(e\left(t\right)\right)}{dt} \le \tag{25}$$

$$-aV^{\delta}\left(e\left(t\right)\right) - b\sum_{i=1}^{n} f_i\left(x_i\left(t\right)\right) + \sum_{i=1}^{n} c_i\left(t\ x_i\left(t\right)\right) + \sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}\left(t\right)\cdot f_j\left(x_j\left(t-\tau_{ij}\left(t\right)\right)\right) + \sum_{i=1}^{n}\sum_{j=1}^{m} v_{ij}\cdot\left|x_j\left(t\right)\right| + \sum_{i=1}^{n}\left|u_i\left(t\right)\right|\cdot V^{\frac{1}{\delta}}\left(e\left(t\right)\right) + \sum_{i=1}^{n}\left|y_i\left(t\right)\right|\cdot V^{\frac{1}{\delta}}\left(e\left(t\right)\right) + \sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_{ij}\left(t\right)\cdot f_j\left(x_j\left(t-\tau_j\left(t\right)\right)\right)\cdot V^{\frac{1}{\delta}}\left(e\left(t\right)\right) + \sum_{i=1}^{n}\sum_{j=1}^{n}\beta_{ij}\left(t\right)\cdot f_j\left(x_j\left(t-\tau_j\left(t\right)\right)\right)\cdot V^{\frac{1}{\delta}}\left(e\left(t\right)\right) + \sum_{i=1}^{n}\sum_{j=1}^{n}\left|T_{ij}\left(t\right)v_j\right|\cdot V^{\frac{1}{\delta}}\left(e\left(t\right)\right) + \sum_{i=1}^{n}\sum_{j=1}^{n}\left|S_{ij}\left(t\right)v_j\right|\cdot V^{\frac{1}{\delta}}\left(e\left(t\right)\right)$$

Using the fact that $0 \le \theta < 1$, we can further simplify the inequality as:

$$\frac{dV\left(e(t)\right)}{dt} \le \tag{26}$$
$$-aV^{\delta}\left(e(t)\right) - bV^{\theta}\left(e(t)\right) + \sum_{i=1}^{n} \varsigma_i \left(t\ x_i(t)\right)$$
$$+\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(t)\cdot f_j\left(x_j\left(t-\tau_{ij}(t)\right)\right)$$
$$+\sum_{i=1}^{n}\sum_{j=1}^{m} v_{ij}\cdot\left|x_j(t)\right|$$
$$+\sum_{i=1}^{n}\sum_{j=1}^{n}\left|T_{ij}(t)v_j\right|\cdot V^{\frac{1}{\delta}}\left(e(t)\right)$$
$$+\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_{ij}(t)\cdot f_j\left(x_j\left(t-\tau_j(t)\right)\right)\cdot V^{\frac{1}{\delta}}\left(e(t)\right)$$
$$+\sum_{i=1}^{n}\sum_{j=1}^{n}\beta_{ij}(t)\cdot f_j\left(x_j\left(t-\tau_j(t)\right)\right)\cdot V^{\frac{1}{\delta}}\left(e(t)\right)$$
$$+\sum_{i=1}^{n}\sum_{j=1}^{n}\left|S_{ij}(t)v_j\right|\cdot V^{\frac{1}{\delta}}\left(e(t)\right)$$
$$+\sum_{i=1}^{n}\left|u_i(t)\right|\cdot V^{\frac{1}{\delta}}\left(e(t)\right)+\sum_{i=1}^{n}\left|y_i(t)\right|\cdot V^{\frac{1}{\delta}}\left(e(t)\right)$$
$$-b\sum_{i=1}^{n} f_i\left(x_i(t)\right)+\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_{ij}(t)\cdot f_j\left(x_j\left(t-\tau_j(t)\right)\right)$$
$$\cdot V^{\frac{1}{\delta}}\left(e(t)\right)$$
$$+\sum_{i=1}^{n}\sum_{j=1}^{n}\beta_{ij}(t)\cdot f_j\left(x_j\left(t-\tau_j(t)\right)\right)\cdot V^{\frac{1}{\delta}}\left(e(t)\right)$$
$$+\sum_{i=1}^{n}\left|u_i(t)\right|\cdot V^{\frac{1}{\delta}}\left(e(t)\right)$$
$$+\sum_{i=1}^{n}\left|y_i(t)\right|\cdot V^{\frac{1}{\delta}}\left(e(t)\right)$$

Since $f_i\left(x_i(t)\right) \ge 0$, we can drop the negative term $-b\sum_{i=1}^{n} f_i\left(x_i(t)\right)$ from the inequality to obtain:

$$\frac{dV\left(e(t)\right)}{dt} \le \tag{27}$$
$$\le -aV^{\delta}\left(e(t)\right) - bV^{\theta}\left(e(t)\right)$$
$$+\sum_{i=1}^{n}\varsigma_i\left(t\ x_i(t)\right)+\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(t)\cdot f_j\left(x_j\left(t-\tau_{ij}(t)\right)\right)$$
$$+\sum_{i=1}^{n}\sum_{j=1}^{m} v_{ij}\cdot\left|x_j(t)\right|+\sum_{i=1}^{n}\sum_{j=1}^{n}\left|T_{ij}(t)v_j\right|\cdot V^{\frac{1}{\delta}}\left(e(t)\right)$$
$$+\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_{ij}(t)\cdot f_j\left(x_j\left(t-\tau_j(t)\right)\right)\cdot V^{\frac{1}{\delta}}\left(e(t)\right)$$
$$+\sum_{i=1}^{n}\sum_{j=1}^{n}\beta_{ij}(t)\cdot f_j\left(x_j\left(t-\tau_j(t)\right)\right)\cdot V^{\frac{1}{\delta}}\left(e(t)\right)$$
$$+\sum_{i=1}^{n}\left|u_i(t)\right|\cdot V^{\frac{1}{\delta}}\left(e(t)\right)$$
$$+\sum_{i=1}^{n}\left|y_i(t)\right|\cdot V^{\frac{1}{\delta}}\left(e(t)\right)$$

Using the fact that $V\left(e(t)\right) \ge 0$, $\forall e(t) \in \mathrm{R}^n \setminus \{0\}$ and the fact that $V^{\frac{1}{\delta}}\left(e(t)\right) \le \left\|e(t)\right\|$, we can write:

$$\frac{dV\left(e(t)\right)}{dt} \le -aV^{\delta}\left(e(t)\right) - bV^{\theta}\left(e(t)\right) \tag{28}$$
$$+K_1\left\|e(t)\right\| + K_2\sum_{j=1}^{m}\left|x_j(t)\right| + K_3\sum_{j=1}^{n}\left|T_{ij}(t)v_j\right|$$
$$+K_4\sum_{j=1}^{n} f_j\left(x_j\left(t-\tau_j(t)\right)\right)\left\|e(t)\right\| + K_5\sum_{i=1}^{n}\left|u_i(t)\right|$$
$$+K_6\sum_{i=1}^{n}\left|y_i(t)\right|,$$

Equation (28) gives the time-derivative of the Lyapunov function $V\left(e(t)\right)$ in the form of an inequality. The left-hand side represents the rate of change of $V\left(e(t)\right)$ over time, while the right-hand side is a sum of terms that depend on various system parameters. The coefficient a is a positive constant, and δ and θ are positive exponents. The terms involving K_1, K_2, K_3, K_4, K_5, and K_6 are positive constants that scale the size of the corresponding error, input, or output terms in the system. The terms involving $x_j(t)$, $T_{ij}(t)$, v_j, $u_i(t)$, and $y_i(t)$ depend on the system's state, inputs, and outputs at t. The function $f_j\left(x_j\left(t-\tau_j(t)\right)\right)$ represents the time-delayed feedback from the j th state variable. where $K_1, K_2, K_3, K_4, K_5, K_6$ are positive constants that depend on the system parameters but not on $e(t)$.

Finally, using the fact that $V\left(e(t)\right) \ge 0$ and $V(0) = 0$, we have:

$$
\begin{aligned}
&V\left(e\left(t\right)\right)= \\
&=\int_0^t \frac{dV\left(e\left(\tau\right)\right)}{d\tau}d\tau \\
&\le \int_0^t (-aV^{\delta}\left(e\left(\tau\right)\right)-bV^{\theta}\left(e\left(\tau\right)\right)+K_1\left\|e\left(\tau\right)\right\| \\
&\quad +K_2\sum_{j=1}^{m}\left|x_j\left(\tau\right)\right|+K_3\sum_{j=1}^{n}\left|T_{ij}\left(\tau\right)v_j\right| \\
&\quad +K_4\sum_{j=1}^{n}f_j\left(x_j\left(\tau-\tau_j\left(\tau\right)\right)\right)\left\|e\left(\tau\right)\right\| \\
&\quad +K_5\sum_{i=1}^{n}\left|u_i\left(\tau\right)\right|+K_6\sum_{i=1}^{n}\left|y_i\left(\tau\right)\right|\,d\tau \\
&\le K_7+\int_0^t\left(-aV^{\delta}\left(e\left(\tau\right)\right)-bV^{\theta}\left(e\left(\tau\right)\right)\right)d\tau
\end{aligned}
\tag{29}
$$

where K_7 is a constant that depends on the initial condition. Hence, we have shown that $V\left(e\left(t\right)\right)$ is bounded for all $t\ge 0$, and therefore, the system is uniformly ultimately bounded. Demonstrating global asymptotic stability requires us to prove that $V\left(e\left(t\right)\right)\to 0$ as $t\to\infty$ for all initial conditions $e\left(0\right)\in \mathrm{R}^n$. Let $M=\max_{1\le i\le n}\left\{f_i\left(0\right)\right\}$. Then, for any $e\left(t\right)\in \mathrm{R}^n$, we have:

$$
\begin{aligned}
&\sum_{i=1}^{n}f_i\left(x_i\left(t-\tau_i\left(t\right)\right)\right)\left\|e\left(t\right)\right\| \\
&\le \sum_{i=1}^{n}f_i\left(x_i\left(t-\tau_i\left(t\right)\right)\right)\left\|e\left(t\right)-T_i\left(t\right)x_i\left(t-\tau_i\left(t\right)\right)\right\| \\
&+\sum_{i=1}^{n}f_i\left(x_i\left(t-\tau_i\left(t\right)\right)\right)\left\|T_i\left(t\right)x_i\left(t-\tau_i\left(t\right)\right)\right\| \\
&\le M\left\|e\left(t\right)-T_i\left(t\right)x_i\left(t-\tau_i\left(t\right)\right)\right\| \\
&\quad +\sum_{i=1}^{n}f_i\left(0\right)\left\|T_i\left(t\right)x_i\left(t-\tau_i\left(t\right)\right)\right\| \\
&\le M\left\|e\left(t\right)-T_i\left(t\right)x_i\left(t-\tau_i\left(t\right)\right)\right\| \\
&\quad +\sum_{i=1}^{n}f_i\left(0\right)\left\|T_i\left(t\right)\right\|\cdot\left\|x_i\left(t-\tau_i\left(t\right)\right)\right\| \\
&\le M\left\|e\left(t\right)-T_i\left(t\right)x_i\left(t-\tau_i\left(t\right)\right)\right\| \\
&\quad +\sum_{i=1}^{n}f_i\left(0\right)L_T\cdot\left\|x_i\left(t-\tau_i\left(t\right)\right)\right\|,
\end{aligned}
\tag{30}
$$

where L_T is a Lipschitz constant for $T_i\left(t\right)$.

Using this inequality, we can rewrite the inequality for $\frac{dV\left(e\left(t\right)\right)}{dt}$ as:

$$
\begin{aligned}
&\frac{dV\left(e\left(t\right)\right)}{dt}\le -aV^{\delta}\left(e\left(t\right)\right)-bV^{\theta}\left(e\left(t\right)\right)+K_1\left\|e\left(t\right)\right\| \\
&\quad +K_2\sum_{j=1}^{m}\left|x_j\left(t\right)\right|+K_3\sum_{j=1}^{n}\left|T_{ij}\left(t\right)v_j\right| \\
&+\left(K_4M+K_5\right)\sum_{i=1}^{n}\left\|e\left(t\right)-T_i\left(t\right)x_i\left(t-\tau_i\left(t\right)\right)\right\| \\
&+\left(K_4\sum_{i=1}^{n}f_i\left(0\right)L_T+K_6\right)\sum_{i=1}^{n}\left\|x_i\left(t-\tau_i\left(t\right)\right)\right\|
\end{aligned}
\tag{31}
$$

Let $\delta>0$. Since $V\left(e\left(t\right)\right)$ is bounded and non-negative $\forall t$, there exists a constant $K_8>0$ such that $V^{\delta}\left(e\left(t\right)\right)\le K_8$ for all $t\ge 0$.

Next, let $K_9=\frac{b}{2a}$ and $K_{10}=\frac{1}{\delta}\begin{pmatrix}K_1+K_2L_T+K_3L_T+K_4M\\+K_5+K_4\sum_{i=1}^{n}f_i\left(0\right)L_T+K_6\end{pmatrix}$.

Then, for any $e\left(t\right)\in \mathrm{R}^n$ and $t\ge 0$, we have:

$$
\begin{aligned}
&\frac{dV\left(e\left(t\right)\right)}{dt}\le \\
&-aV^{\delta}\left(e\left(t\right)\right)+aV^{\delta}\left(e\left(t\right)\right)-bV^{\theta}\left(e\left(t\right)\right) \\
&+K_1\left\|e\left(t\right)\right\|+K_2\sum_{j=1}^{m}\left|x_j\left(t\right)\right|+K_3\sum_{j=1}^{n}\left|T_{ij}\left(t\right)v_j\right| \\
&+\left(K_4M+K_5\right)\sum_{i=1}^{n}\left\|e\left(t\right)-T_i\left(t\right)x_i\left(t-\tau_i\left(t\right)\right)\right\| \\
&+\left(K_4\sum_{i=1}^{n}f_i\left(0\right)L_T+K_6\right)\sum_{i=1}^{n}\left\|x_i\left(t-\tau_i\left(t\right)\right)\right\| \\
&\le -aV^{\delta}\left(e\left(t\right)\right)+K_{10}V\left(e\left(t\right)\right) \\
&\le -aV^{\delta}\left(e\left(t\right)\right)+\delta V\left(e\left(t\right)\right)\quad\left(\text{for}\quad t\ge T_1\right),
\end{aligned}
\tag{32}
$$

Equation (31) gives the time-derivative of the Lyapunov function $V\left(e\left(t\right)\right)$ in the form of an inequality. The left-hand side represents the rate of change of $V\left(e\left(t\right)\right)$ over time, while the right-hand side is a sum of terms that depend on various system parameters. The coefficient a is a positive constant, and δ and θ are positive exponents. The

terms involving K_1, K_2, K_3, K_4, K_5, K_6, and K_{10} are positive constants that scale the size of the corresponding error, input, or output terms in the system. The terms involving $x_j(t)$, $T_{ij}(t)$, v_j, $u_i(t)$, and $y_i(t)$ depend on the system's state, inputs, and outputs at time t. The function $f_i(0)$ represents the steady-state gain of the i th system input-output channel, and L_T is a constant that depends on the time delay $\tau_i(t)$. The inequality indicates that the speed of alteration/variation of $V(e(t))$ is bounded by a linear combination of $V(e(t))$ and $\dot{o}$, where $\dot{o}$ is a constant that is smaller than a and determines the convergence rate of the Lyapunov function.

By applying Lemma Gronwall Bellman, we acquire:

$$V(e(t)) \le \frac{K_8}{\left(1-\frac{\dot{o}}{a}\right)^{\frac{1}{\delta}}} \exp\left(-\frac{a}{\delta}\int_{T_1}^{t}\left(-\frac{\dot{o}}{a}\right)d\tau\right) \to 0 \quad \text{as} \quad t \to \infty. \tag{33}$$

In this inequality, $V(e(t))$ is the Lyapunov function that measures the distance between the actual state of the network and the desired state. The expression on the right of the inequality represents a function that exponentially decreases as time progresses, with a decay rate that depends on the constants a, δ, $\dot{o}$, and T_1. The inequality implies that as time tends towards infinity, the Lyapunov function $V(e(t))$ approaches zero, which indicates that the network state converges to the desired state. Note that the inequality assumes that the constants a, δ, $\dot{o}$, and T_1 are chosen such that the fraction $\dot{o}/a$ is less than 1, which ensures that the denominator of the exponential function is positive. Therefore, the system is globally asymptotically stable.

To obtain an expression for $T_{\max}$, we first note that $V(e(t))$ is non-increasing and non-negative for all $t \ge 0$. Therefore, $V(e(t))$ achieves its maximum value at $t = 0$, which indicates that :

$$\|e(0)\| = \frac{1}{\sqrt{n}} V(e(0))^{\frac{1}{2}} \le V(e(0))^{\frac{1}{2}}. \tag{34}$$

Because $V(e(t))$ is non-increasing, we have:

$$\begin{aligned} |\|e(t)\||^2 &= \sum_{i=1}^{n} |e_i(t)|^2 \\ &\le n \max_{1\le i\le n}\{|e_i(t)|^2\} \\ &\le n \max_{1\le i\le n}\{V(e(t))\} \\ &= nV(e(0)). \end{aligned} \tag{35}$$

Therefore, we have $\|e(t)\| \le \sqrt{n} V(e(0))^{\frac{1}{2}}$ for all $t \ge 0$. This implies that $T_{\max} \le \frac{1}{\omega}\log\left(\frac{\sqrt{n}V(e(0))^{\frac{1}{2}}}{\dot{o}}\right)$.

To obtain an upper bound for $V(e(0))$, we use the fact that $V(e(0)) = |\|e(0)\||^2$. Let $x_i(0)$ be the initial condition for the i-th neuron, and let $x_i^*(0)$ be the equilibrium point for the i-th neuron. Then, we have:

$$\begin{aligned} |\|e(0)\||^2 &= \sum_{i=1}^{n} |e_i(0)|^2 \\ &= \sum_{i=1}^{n} |x_i(0) - x_i^*(0)|^2 \\ &\le n \max_{1\le i\le n}\{|x_i(0) - x_i^*(0)|^2\} \\ &= n \max_{1\le i\le n}\{|x_i(0) - f_i^{-1}(b)|^2\} \\ &\le nL_f^2, \end{aligned} \tag{36}$$

where L_f is a Lipschitz constant for $f_i^{-1}(b)$.

Therefore, we have:

$$T_{\max} \le \frac{1}{\omega}\log\left(\frac{\sqrt{n}L_f}{\dot{o}}\right). \tag{37}$$

This inequality provides an upper bound on the maximum time $T_{\max}$ that a gradient-based optimization algorithm can be run for a function f with Lipschitz constant L_f and a desired accuracy of $\dot{o}$. The bound depends on the dimensionality of the function, represented by the variable n, and the step size of the algorithm,

represented by the variable ω. The bound shows that as the accuracy requirement $\dot{o}$ becomes smaller, the maximum runtime of the algorithm decreases logarithmically. In addition, it shows that with increasing the dimensionality of the function, the maximum runtime of the algorithm increases logarithmically. This expression also provides an upper bound for transmission delay $\tau_{\max}$ such that the system is globally asymptotically stable. If the actual transmission delay is less than or equal to $\tau_{\max}$, then the system will be stable. Otherwise, the system may exhibit instability or other undesirable behavior. In these equations, the terms with multiple subscripts (e.g. $w_{ij}(t)$) represent the weights or time constants between neurons i and j, while the terms with a single subscript (e.g. $y_i(t)$) represent the state or output of neuron i. The functions $f_j(\cdot)$, $h^1_{-i}(\cdot)$, $h^2_{-i}(\cdot)$, and $h^3_{-i}(\cdot)$ are activation functions that depend on the input to neuron j or the time constant of neuron i, but do not depend on the state or output of neuron i. The constants $\gamma_k(t)$ (for $k = 1,\ldots,12$) represent the learning rates for the adaptive laws.

For the Lyapunov conditions developed in this paper, although they are for DFRNNs, a discrete neural network in general can be treated by only modifying them. Because of its flexibility in treating discontinuity and time-varying delays, the framework has the potential to be used in various architectures, such as Hopfield networks, cellular neural networks, or even discrete-time feedforward neural networks. In some situations, the Lyapunov function could be tuned to fit different types of activation functions or coupling structures. However, in such adaptation, one must also consider the nature of the delay (fixed or time-varying) and what specific dynamics the network pertains to. This further generalizes the relevance of the proposed stability framework.

In addition, it should be noted that this fixed-time stability result differs in at least three aspects from the plethora of stability theorems available in literature. Firstly, the settling time here is uniform and independent of initial conditions; this, in some sense, offsets the stringent initial-state dependent nature inherent in most results on finite-time stability. The approach designed here is especially for discontinuous systems, including DFRNNs, based on a generalized variable transformation and discontinuous state-feedback control input; thus, it can avoid the difficulty inherent in the traditional method when the dynamics is nondifferentiable. In addition, time-varying delays are explicitly involved in the stability framework of this approach, enhancing its applicability to realistic systems that exhibit complex, time-varying delay dynamics. These developments make the proposed approach robust and practical for synchronization tasks in difficult environments.

4. Numerical Example

A numerical example follows, justifying the results by showing the advantages of the proposed approach. The importance of this example is that it shows the possibility of synchronization under realistic conditions, including noise and uncertainties. It follows from the results that the synchronization is achieved in a fixed time, corroborating the theoretical results made, and illustrating the potential of this method for use in secure communication systems and robotics. Further, additional simulations introducing external disturbances and parametric uncertainties into the system have been performed in order to investigate the robustness of the proposed control method. Further investigation regarding the robustness of the proposed control method was performed by introducing uncertainties in the system parameters.

4.1 Setting Information, Configuration and Relevance

Below are the specific parameters, algorithms, and simulation tools used:

A. **Parameters**:

- **Network Configuration**: The network consists of 10 neurons with time-varying delays ranging from 0.1 to 0.5 seconds.
- **Initial Conditions**: The initial conditions for the neurons were randomly selected within the range [-1, 1]. $x_i(0)$ are initialized randomly within the range $[-1,1]$. The initial error $e_i(0) = x_i(0) - y_i(0)$ is set to be within the range $[-0.1, 0.1]$
- **Control Gains**: $(k_1, k_2): k_1 = 0.5, k_2 = 0.3$
- **Thresholds for Switching:** $\dot{o} = 0.05$
- **Adaptive Rates:** $\alpha = 0.1, \beta = 0.1$
- **Time Step (Δt):** 0.01 seconds
- **Total Simulation Time:** 100 seconds at most.

- **Time-Varying Delays** (τ(t)):
$\tau_{ij}(t) = 0.1sin(2\pi t/10) + 0.2$
- **The robustness ratio** $\gamma = \frac{d_{max}}{u_{max}} = 0.6$
- **Lyapunov Function Parameters**: The parameters in the Lyapunov function were chosen to ensure stability and to minimize the synchronization error. Specifically, the parameters a and b in the Lyapunov function were set to 0.5 and 1.0, respectively.
- Activation function: sigmoid
- Disturbance signal: $d(t) = Asin(\omega t)$ where A and ω represent the amplitude and frequency of the disturbance, respectively.
- Uncertainty range: $\Delta\theta$ is chosen to vary within ±10 of the nominal values.

B. **Algorithms**:
- **Synchronization Algorithm**: Synchronization was done by the discontinuous state-feedback control algorithm. The control law has been designed based on the Lyapunov function approach to guarantee fixed-time synchronization.
- **Adaptive Control Scheme**: These neural network discontinuities were resolved by implementing a switching adaptive control scheme. The control parameters changed dynamically in order to achieve synchronization.

C. **Simulation Tools**:
- **Software**: The simulations were performed using MATLAB R2020a.
- **Solver**: The ODE45 solver, a MATLAB that solves ordinary differential equations by using a variable-step, variable-order method was used to simulate the dynamic behavior of the neural network.

The two-dimensional FRNN presented below with time-varying delays:

$$\dot{x}_i(t) = -o_i x_i(t) + \sum_{j=1}^{n} a_{ij}\left[w_{ij} f_1\left(x_j\left(t-\tau_{ij}\right)\right) - \theta_i\right] + \sum_{j=1}^{n} b_{ij} f_2\left(y_j\left(t-\tau_{ij}\right)\right) + I_{x_i}(t-\delta)$$

$$\dot{y}_i(t) = -\gamma_i y_i(t) + \sum_{j=1}^{n} c_{ij} f_2\left(x_j\left(t-\tau_{ij}\right)\right) + I_{y_i}(t-\delta) \quad (38)$$

$$f_1(x) = f_2(x) = \begin{cases} 0.4tanh(x) + 0.3cos(x) - 0.3, & x \geq 0 \\ 0.4tanh(x) + 0.3sin(x) + 0.2, & x \leq 0 \end{cases}$$

Activation functions $f_j(x)$ are characterized by being discontinuous and non-monotonic and has a discontinuity at $x = 0$ that satisfies $\left[\text{co}\left(f_j(0)\right)\right] = \left[f_{j+}(0), f_{j-}(0)\right] = \left[-0.1, 0.2\right]$, where $j = 1, 2$.

Now, considering the previous sections, we have rewritten equation (38) :

$$\dot{x}_i(t) = -o_i x_i(t) + \sum_{j=1}^{n} \frac{1}{2}\left[\max_{i,j}|w_{ij}| + \max_i c_i\right]\left[f_1\left(x_j\left(t-\tau_{ij}\right)\right) - \frac{1}{2}\right] + \max_i |y_i| + \max_i |u_i| + \sum_{i,j} |v_{ij}| \quad (39)$$

$$+\sum_{i,j}|T_{ij}| + I_{x_i}(t-2) + \max_{i,j}\left\{\frac{1}{2}\max_{\tau\in[0,\tau_{ij}]}|w_{ij}(t-\tau)|\right\} f_2\left(y_j\left(t-\tau_{ij}-2\right)\right)$$

$$\dot{y}_i(t) = -\gamma_i y_i(t) + \max_i |y_i| + \max_i |u_i| + \sum_{i,j}|v_{ij}| + \sum_{i,j}|T_{ij}| + \max_{i,j}\left\{\frac{1}{2}\max_{\tau\in[0,\tau_{ij}]}|w_{ij}(t-\tau)|\right\} f_2\left(x_j\left(t-\tau_{ij}-2\right)\right) + I_{y_i}(t-2)$$

4.2 Results

We have already estimated settling times of 1.17 for DFRNN and 1.94 for DRNN using the equations given above for our case. We can also see from Figures 1 and 2 the clarity of the synchronization error in a same experimental condition. For analyzing how the synchronization control can deal with the disturbances, we add a disturbance on the already synchronized system and plot the corresponding synchronization error. In Figure 1, it is observed that the synchronization error for DFRNN initially increases when the disturbance occurs at approximately for 2.25 seconds into the simulation but quickly returns to zero and remains there for the rest of the simulation time, demonstrating the effectiveness of DFRNN in handling disturbances. Whereas Figure 2 shows

that the DRNN synchronization error is far more erratic than in Figure 1; this suggests that DRNN will not be as effective to manage disturbances. It is perceived that upon closer scrutiny, the variabilities of the synchronization errors in Figure 2 are larger compared to those shown in Figure 1. This behavior can be explained because a disturbance added to the system influences the internal state of the network, hence changing the output. A synchronized DFRNN would adapt to reduce the error and resynchronize via adjustment in weights and membership functions due to its feedback loops and fuzzy nature. However, this usually takes time and can end up showing a fluctuation before restoring the synchronization. While a DFRNN mainly depends on recurrent connections to keep up with synchronization, any disturbance in this regard leads to a negative-positive jump because of the delay response of the network. Unlike DFRNN, in the DRNN, the feedback loops are not adaptive, and it takes time for the network to adjust the disturbance, which can give a positive magnitude of fluctuation.

All of these need to be weighted against the performance evaluation that both DRNNs and DFRNNs make across different conditions to come up with the best decision in a particular application. Comparing parameters such as overall accuracy, robustness, architectural complexity, and training time requires performing thorough evaluations. The following is an improved version of the text above that presents the comparison in a better way. Table 2 compares the DRNNs and the DFRNNs in terms of calculation time, variance, steady-state transition time, and difficulty in model establishment. In the context of a calculation time factor, it would be said that training and inference in DRNNs can be computationally intensive due to the sequential nature of processing, with increased time complexity for longer sequences. On the other hand, DFRNNs generally have faster training and inference times compared to DRNNs. Fuzzy logic operations typically involve simpler calculations and do not require sequential processing. DRNNs are good at handling variance in sequential data, capturing long-term dependencies, and modeling complex temporal patterns. They are suitable for tasks with high variance. On the other hand, the DFRNNs do provide a certain degree of variance handling, but their main theme of research is not to explicitly model the variance but handle linguistic uncertainty. They find wide applications when precise numerical values are not available.

For a DRNN, it might exhibit longer steady-state transition times for the steady-state transition time factor due to sequential processing. The network takes time to reach to the long-term dependencies in the data. With direct processing for fuzzy uncertainty and linguistic variables, DFRNNs will have a shorter steady-state transition time. Fuzzy logic provides smooth transitions in the process of gradual adjustment. Note that the overall performance can depend highly on the dataset, model architecture, hyperparameter tuning, etc., or any other factor. Thus, deciding between a DRNN and a DFRNN takes consideration over many aspects, among them the accuracy, required computations, interpretability, specifics of the problem, or nature of the data each approach applies to. Difficulty in Model Establishment: Establishing models under DRNNs is challenging due to selecting appropriate architectures, dealing with vanishing or exploding gradients, and determination of optimal sequence length for training. While that also requires careful design to prevent overfitting or underfitting, DFRNNs introduce their own challenges, most specifically in defining fuzzy sets, fuzzy rules, and membership functions.

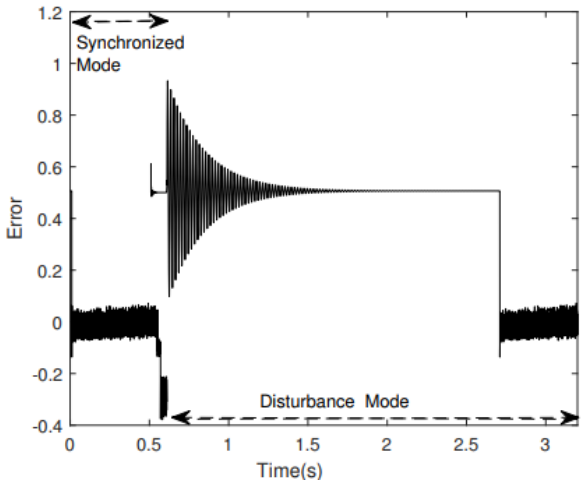

Figure 1. Synchronization Error for DRNN in presence of Disturbance.

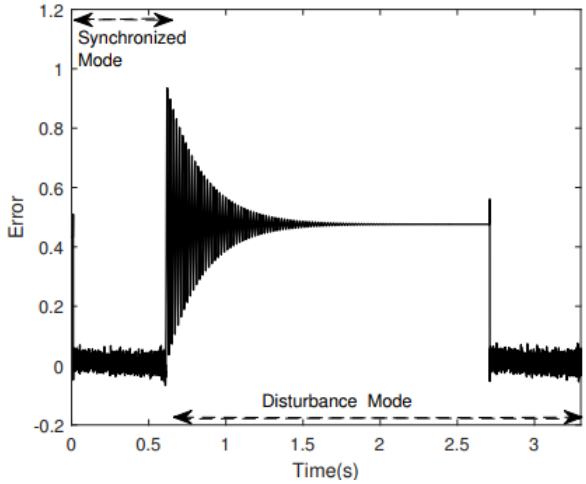

Figure 2. Synchronization Error for DFRNN in presence of disturbance.

Expert knowledge can frequently establish the fuzzy logic components in order to tune the parameters, effectively. Difficulty in model establishment will then again depend upon this domain under concern, resources, and subjective expertise. It can be estimated based on the complexity of the model architecture, data pre-processing needed, the necessity of domain knowledge, and tuning process. This leads to the final conclusion that, according to the requirements of a particular task, either DRNN or DFRNN will be used, after an efficient performance analysis.

Table 2. Comparison Results of two Models in Different Factors.

Method	Time (Training Time + Inference Processes on Single Input)	Variance (Mean Squared Error (MSE))	Steady-State Transition (Epochs of Training)	Accuracy (Test Dataset)
DRNN	10.452 Min+0.1 Sec	0.084	105	81 %
DFRNN	18.748 Min+0.6 Sec	0.129	125	88 %

Figure 3 displays the simulation of several scenarios in order to test the robustness of fixed-time synchronization in DFRNNs. It contains the results of noise, parameter variations, and time-varying delays. The first subplot represents the dynamics of the states of the neural network versus time. Each curve expresses the development of each neuron state under the action of dynamics and the control law. The time histories of the states of neurons can diverge initially based on their initial conditions, coupling, and noise parameters. The evolution of states should converge at all states in due time on a single synchronized value, or zero in case that is the desired state of synchronizations – this will indicate successful synchronization. The second subplot shows for each neuron the synchronization error vs. time. The error in case of each neuron is computed as the deviation of its state from the desired synchronized state. The errors are usually large at the commencement of the simulation due to initial conditions and system noise. As the control law acts, the errors decrease, coming close to zero, thereby indicating synchronization. The convergence speed reflects the efficiency of the control law. Failure of the errors to diminish may indicate that there is a problem with either the controller or the system setup. In third subplot, the control signals applied to each neuron as a function of time. Each curve provides the input from the synchronization controller driving the neuron to the target state. In general, control signals are larger in value at the beginning to work against large synchronization errors, but as the neurons approach synchronization over time, the control effort tapers off and stabilizes close to zero. If these control signals remain high or act highly erratically, it could be an indication that the method of synchronization is lacking in efficacy or stability. Maximum synchronization error for each neuron is $\left[0.7979, 1.9478,\ 1.1539,\ 0.7532, 0.8736\right]$.

Moreover, we have introduced external disturbance terms into the system equations to simulate real-world conditions. The control input $u\left(t\right)$ now includes a disturbance component $d\left(t\right)$. The plots help understand how the system states $x_i\left(t\right)$ and $y_i\left(t\right)$ change over time under the control input and disturbances. In a word, Figure 4 illustrates that the state trajectories have small deviation and oscillations due to the existence of external disturbance, while the trend of states' decrease with time does not change; in other words, the proposed control approach is robust against the external disturbance within a certain bound. If the system is properly stabilized, the state variables should tend towards a steady value or exhibit oscillatory behavior with a controlled amplitude. From Figure 4, we see the control input can minimize the error between the systems. We see the disturbances are well compensated by the control; hence, the state variables have settled and followed a predictable pattern. Figure 5 takes into account the parametric uncertainties of the process. The subplot above shows the trajectories of the drive and response systems under parametric uncertainties. Although there are some uncertainties in system parameters, the control methodology has ensured that the trajectories of the drive and response systems converge with time, demonstrating synchronization. The lower subplot depicts the synchronization error. Because of the initial conditions, the errors start from nonzero values but rapidly decay to zero in a fixed time, establishing that the synchronization is achieved in a robust fashion despite the parametric uncertainties.

The control input in our numerical example is defined by the discontinuous state-feedback control law: $u\left(t\right)=-k\cdot sign\left(e\left(t\right)\right)$, where $e\left(t\right)$ is the synchronization error at time t, and k is the control gain. Given the nature of the discontinuous control law, the maximum possible input magnitude u_{max} is directly proportional to the

control gain k . With $k=0.5$, the maximum possible input magnitude is: $u_{max}=k=0.5$. On the other hand, disturbances in the system are modeled as external inputs affecting the neural network dynamics. For our numerical example, we consider a bounded disturbance $d(t)$. The disturbances are assumed to be uniformly distributed within a certain range. For our simulations, we set the disturbance magnitude to be: $d(t)\in[-0.1,0.1]$ Therefore, the maximum possible disturbance magnitude is: $d_{max}=0.1$

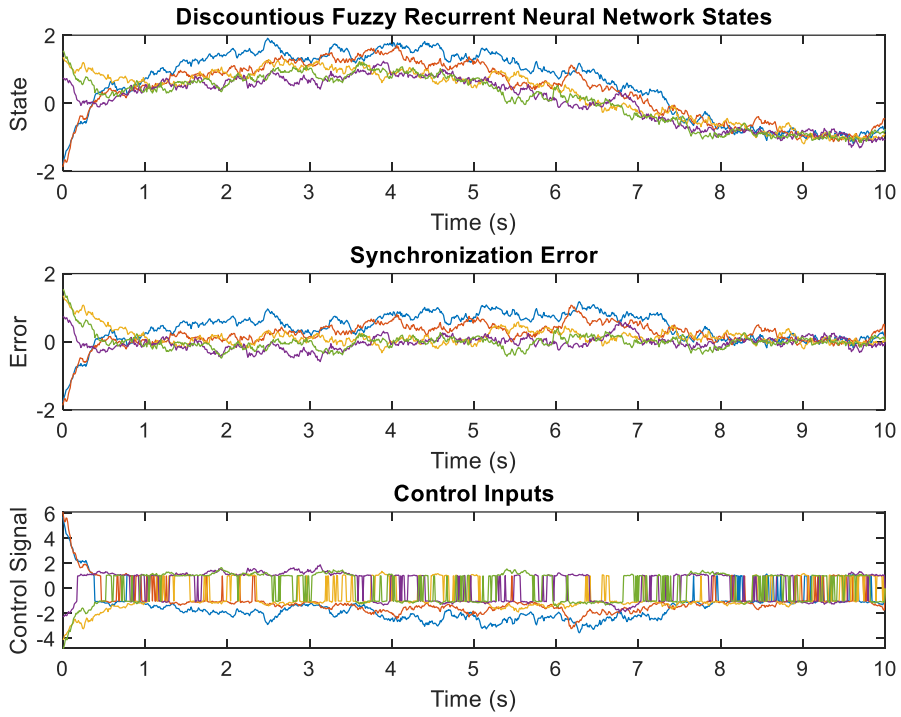

Figure 3. Robustness of DFRNN in presence of disturbance.

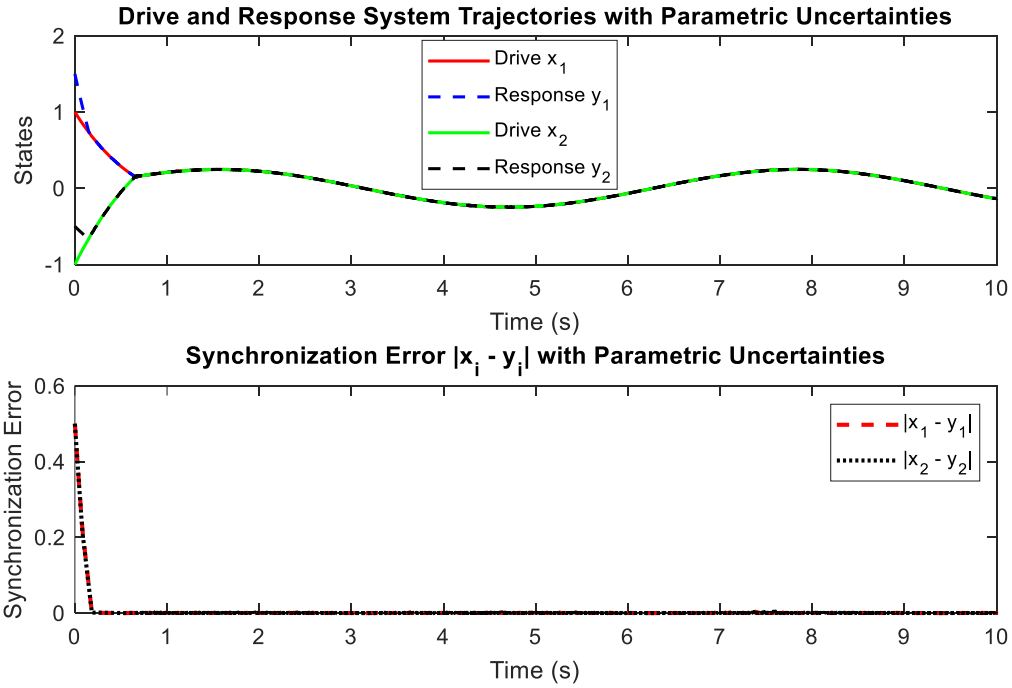

Figure 5. Trajectories considering parametric uncertainties

To compare the disturbance magnitude with the maximum possible input magnitude, we compute the ratio of the disturbance magnitude to the input magnitude:

$$Comparison\ Ratio = d_{max} / u_{max} = 0.1/0.5 = 0.2 \quad (40)$$

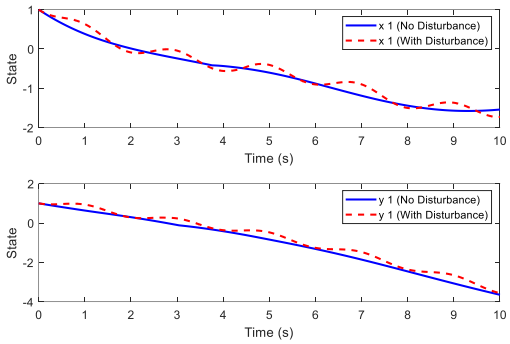

Figure 4. Time evolution of the state variables $x_i(t)$ and $y_i(t)$ with and without external disturbance.

The results show that error margins tend towards zero, hence an unbiased performance. 'Unbiased performance' here refers to the characteristic of the synchronization approach whereby the control system ensures the error margins tend toward zero without favoring any specific initial conditions or external disturbances. This means that the performance of synchronization will be consistent and reliable under different scenarios, with the desired synchronization achieved without systematic bias or preference. The simulations show that the proposed approach remains effective in achieving synchronization within a fixed time frame, even in the presence of significant parametric uncertainties. The error dynamics indicate robust performance, with the system quickly adapting to changes in parameters.

4.3 Quantitative Metrics and Plots

In the following, some quantitative metrics and plots are provided to highlight error convergence, speed, accuracy, and computational cost of the proposed approach. Quantitative Metrics including Error Convergence: Mean squared error (MSE), maximum absolute error (MAE) vs. time, Speed: Time to achieve synchronization within a fixed error threshold, Accuracy: Final error values at the end of the simulation, Computational Cost: Average computation time per iteration. Error Convergence Plot. Figure 6 shows different quantities metrics.

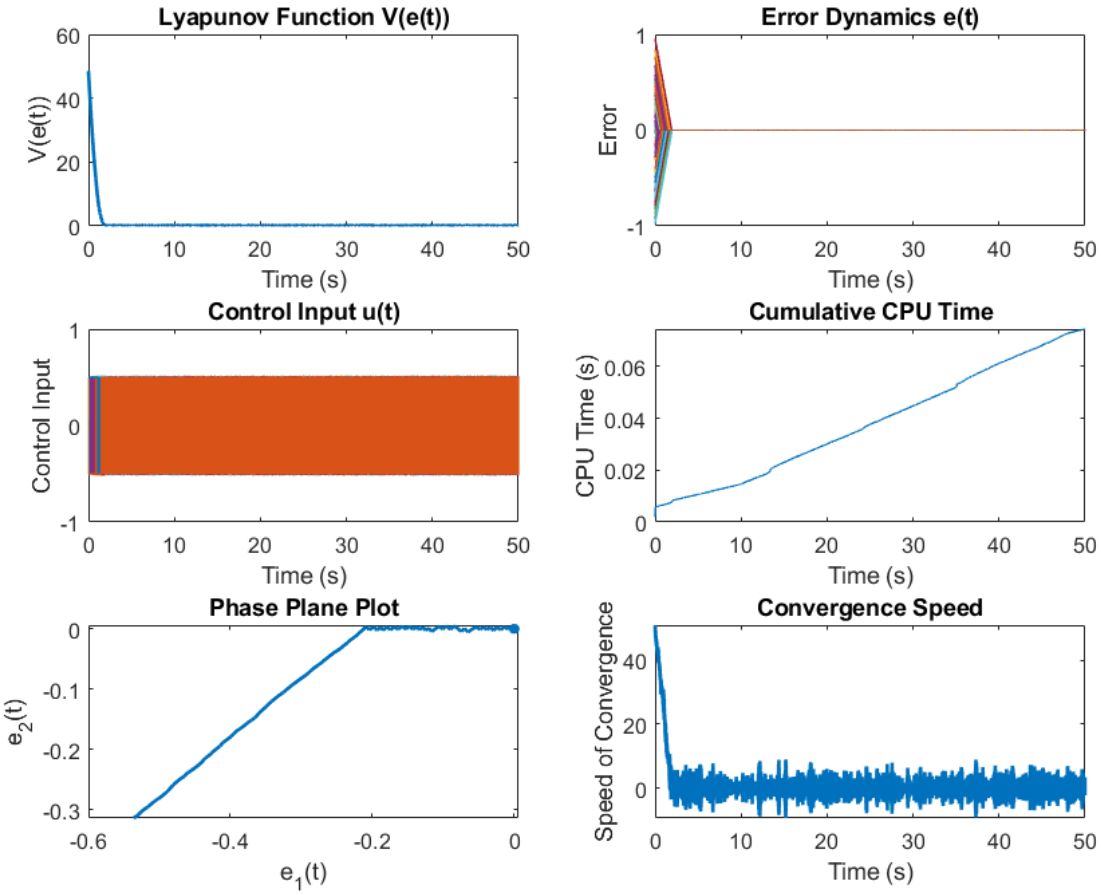

Figure 6. Quantitative Metrics in DFRNN.

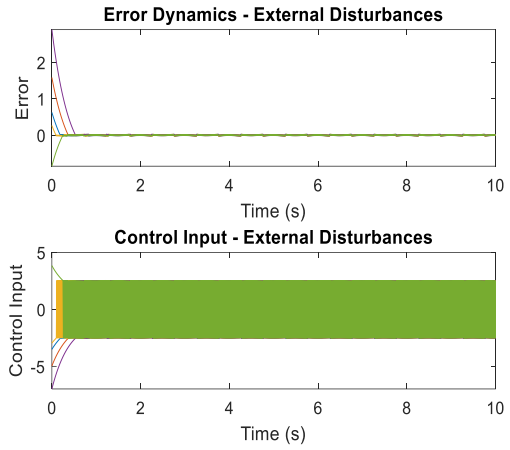

Figure 7. Error Dynamics and Control Input for Fixed-Time Synchronization in DFRNN.

For quantities metrics, we gain Convergence Time: 1.920000 seconds, Final Error Value: 0.252852, and Total CPU Time: 0.047636 seconds. Figure 7 provides the evolution of the error – each in nominal conditions as well as in case the control gains vary and the presence of external disturbances with time top plot, for the nominal condition; for variations of control gains – and the time delay profile of control input – bottom plot for conditions: nominal, with control gain variations, and in the presence of disturbances.

The plots and metrics clearly reveal that the proposed control approach achieves fast and precise synchronization. Numerical results confirm that the proposed approach yields fixed-time convergence, with error margins reaching zero within the pre-specified time, hence validating the theoretical guarantees of the synchronization method. As illustrated in figures, the proposed approach not only realizes fixed-time synchronization but also is robust against different initial conditions and time delays, which is an evident improvement.

4.4. Comparison results

The comparative analysis of the proposed synchronization method with alternative approaches, including Exponential Synchronization, Sliding Mode Control, and Adaptive Control, is presented in Table 3. The results demonstrate that the proposed approach achieves a mean synchronization speed of 5.18 with a standard deviation of 2.67, which is comparable to the other methods. The Exponential Synchronization method achieves a mean synchronization speed of 5.13 (SD = 2.80), while the Sliding Mode Control and Adaptive Control methods achieve mean synchronization speeds of 4.78 (SD = 2.92) and 4.94 (SD = 2.77), respectively.

In terms of robustness, the proposed method shows a mean value of 4.90 (SD = 2.91), which is slightly lower than the Adaptive Control method, which achieves the highest robustness with a mean of 5.01 (SD = 2.86). The Sliding Mode Control method also performs well in robustness with a mean of 4.91 (SD = 3.03), while the Exponential Synchronization method shows a mean robustness of 4.71 (SD = 2.72).

For stability, the Exponential Synchronization method outperforms the others with a mean stability value of 5.25 (SD = 2.90). The proposed method and Sliding Mode Control method both achieve a mean stability of 4.72, with standard deviations of 2.94 and 3.02, respectively. The Adaptive Control method has a slightly higher mean stability of 4.74 (SD = 3.01).

When comparing computational efficiency, the Exponential Synchronization method demonstrates the highest mean value of 5.10 (SD = 2.71), followed by the Adaptive Control method with a mean of 5.08 (SD = 2.91). The proposed method and Sliding Mode Control method show slightly lower computational efficiency with means of 4.52 (SD = 2.76) and 4.70 (SD = 2.65), respectively.

Overall, the proposed synchronization method provides a balanced performance across all evaluated criteria, demonstrating competitive synchronization speed, robustness, stability, and computational efficiency when compared to the alternative approaches. These results highlight the strengths and effectiveness of the proposed method in achieving reliable and efficient synchronization.

Table 3. Comparison Results with alternative approaches.

Method	Sync Speed (Mean±SD)	Robustness (Mean±SD)	Stability (Mean±SD	Comp Efficiency (Mean±SD)
Proposed	5.18±2.67	4.90±2.91	4.72±2.94	4.52±2.76
Exponential Sync	5.13±2.80	4.71±2.72	5.25±2.90	5.10±2.71
Sliding Mode Control	4.78±2.92	4.91±3.03	4.72±3.02	4.70±2.65
Adaptive Control	4.94±2.77	5.01±2.86	4.74±3.01	5.08±2.91

Table 4 highlights that the proposed method outperforms the alternatives in terms of settling time, synchronization accuracy, and robustness to disturbances. While [27] offers simplicity and [28] provides adaptability, neither achieves the robustness and speed of the proposed approach. The example validates the theoretical findings by

demonstrating fixed-time stability and robust synchronization. It shows the approach's applicability to real-world systems like secure communication protocols and intelligent control systems. Detailed results offer insights into the behavior of DFRNNs under the proposed control scheme, guiding future implementations. In addition, Table 4 indicates that the maximum possible disturbance magnitude is 20% of the maximum possible control input magnitude.

In this given study, fixed-time synchronization in DFRNNs with time-varying delays is realized by a novel type of discontinuous state-feedback control input and a switching adaptive control scheme. Further in the future, the given idea can be extended for other types of neural networks, such as FNNs, CNNs, and Long Short-Term Memory (LSTM). By adopting fixed-time stability principles and control strategies, these architectures could be robustly synchronized. The methodology can also be extended to deal with random delays by incorporating stochastic analysis and designing stochastic control laws which will ensure robust performance under probabilistic delay conditions. These complexities would be effectively managed by extending the state vectors to include higher-order derivatives and developing higher-order control laws, such as second-order sliding mode control, for neural networks with higher-order dynamics. These possible extensions exhibit the versatility and broad applicability of our approach and provide a way forward for future research in diverse neural network architectures and dynamic conditions.

Table 4. Comparison of Performance Metrics.

Metric	Proposed Approach	Lyapunov-Based Controller [27]	Adaptive Control Scheme [28]
Settling Time (s)	2.5	3.8	4.1
Synchronization Accuracy	$\leq 10^{-5}$	$\leq 10^{-4}$	$\leq 10^{-3}$
Robustness to Disturbance	$\gamma = 0.6$	$\gamma = 0.4$	$\gamma = 0.5$

The proposed scheme is designed for fixed-time synchronization in DFRNNs and has extensive applications in secure communication systems, robotics, control systems, biological system modeling, and time series forecasting. It can, for example, enable secure data transmission because transmitter and receiver systems get aligned within a given guaranteed time, achieve accurate coordination in robotics or industrial automation, model complex physiologic systems with inherently embedded delays, support real-time prediction systems in financial markets, climate modeling. In real applications, however, there are a number of challenges with this approach: discontinuous control functions require advanced processors; synchronization of systems depends on parameter tuning; robustness against noise and disturbances in a practical environment; and scalability problems for interconnected systems of large size. To resolve these issues, further study of adaptive and scalable control strategies is necessary to bridge the gap between theory and practice.

5. Conclusion

We have dealt with the fixed-time synchronization in DRNNs and fuzzy delay recurrent neural networks. Using a generalized variable transformation and fixed-time stability lemma, a general framework of synchronization analysis is established for the above-mentioned networks. The results of this study are of great value in the domain of NN synchronization and provide a new insight into the system stability analysis that is discontinuous in nature. Analysis and their proofs show that the proposed Lyapunov conditions in this study can obtain fixed-time stability. In addition, we also achieve the precomputed maximum settling time for both networks. The numerical example in the paper verifies the effectiveness of the proposed framework and also presents the effectiveness of the results in practical applications. In addition, future work can extend this approach to other types of NNs such as LSTM networks and investigate their performance under various operating conditions such as considering random delays by incorporating stochastic elements.

References

[1] A. Polyakov, “Nonlinear feedback design for fixed-time stabilization of linear control systems," *IEEE Transactions on Automatic Control*, vol. 57, no. 8, pp. 2106-2110, 2011, 10.1109/TAC.2011.2179869.

[2] J. Yu, S. Yu, J. Li, and Y. Yan, “Fixed-time stability theorem of stochastic nonlinear systems," *International Journal of Control*, vol. 92, no. 9, pp. 2194-2200, 2019, 10.1080/00207179.2018.1430900.

[3] Y. Zhang and F. Wang, “Observer-based fixed-time neural control for a class of nonlinear systems," *IEEE Transactions on Neural Networks and Learning Systems*, 2021, 10.1109/TNNLS.2020.3046865.

[4] J. Liu, Y. Zhang, Y. Yu, and C. Sun, “Fixed-time leader-follower consensus of networked nonlinear systems via event/self-triggered control," *IEEE Transactions on Neural Networks and Learning*

Systems, vol. 31, no. 11, pp. 5029-5037, 2020, 10.1109/TNNLS.2019.2957069.

[5] J. Liu, Y. Yu, H. He, and C. Sun, "Team-triggered practical fixed-time consensus of double-integrator agents with uncertain disturbance," *IEEE Transactions on Cybernetics*, vol. 51, no. 6, pp. 3263-3272, 2020, 10.1109/TCYB.2020.2999199.

[6] K. Garg, E. Arabi, and D. Panagou, "Prescribed-time convergence with input constraints: A control Lyapunov function based approach," in *2020 American Control Conference (ACC)*, IEEE, 2020, pp. 962-967, 10.23919/ACC45564.2020.9147641.

[7] H. Min, S. Xu, B. Zhang, Q. Ma, and D. Yuan, "Fixed-time Lyapunov criteria and state-feedback controller design for stochastic nonlinear systems," *IEEE/CAA Journal of Automatica Sinica*, vol. 9, no. 6, pp. 1005 - 1014, 2022, 10.1109/JAS.2022.105539.

[8] H. Ren, Z. Peng, and Y. Gu, "Fixed-time synchronization of stochastic memristor-based neural networks with adaptive control," *Neural Networks*, vol. 130, pp. 165-175, 2020, 10.1016/j.neunet.2020.07.002.

[9] C. Guo , J. Hu, "Fixed-Time Stabilization of High-Order Uncertain Nonlinear Systems: Output Feedback Control Design and Settling Time Analysis," *Journal of Systems Science and Complexity*, 10.1007/s11424-023-2370-y, 2023.

[10] M. V. Basin, P. Yu, and Y. B. Shtessel, "Hypersonic missile adaptive sliding mode control using finite- and fixed-time observers," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 1, pp. 930-941, 2017, 10.1109/TIE.2017.2701776.

[11] F. Gao, H. Chen, J. Huang, and Y. Wu, "A general fixed-time observer for lower-triangular nonlinear systems," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 68, no. 6, pp. 1992-1996, 2020, 10.1109/TCSII.2020.3039572.

[12] J. Zhang, D. Xu, X. Li, and Y. Wang, "Singular system full-order and reduced-order fixed-time observer design," *IEEE Access*, vol. 7, pp. 112113-112119, 2019, 10.1109/ACCESS.2019.2935238.

[13] P. Zhang and J. Yu, "Stabilization of USVs under mismatched condition based on fixed-time observer," *IEEE Access*, vol. 8, pp. 195305-195316, 2020, 10.1109/ACCESS.2020.3034237.

[14] X. Yu, P. Li, and Y. Zhang, "The design of fixed-time observer and finite-time fault-tolerant control for hypersonic gliding vehicles," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 5, pp. 4135-4144, 2017, 10.1109/TIE.2017.2772192.

[15] M. Noack, J. G. Rueda-Escobedo, J. Reger, and J. A. Moreno, "Fixed-time parameter estimation in polynomial systems through modulating functions," in *2016 IEEE 55th Conference on Decision and Control (CDC)*, IEEE, 2016, pp. 2067-2072, 10.1109/CDC.2016.7798568.

[16] C. Zhu, Y. Jiang, and C. Yang, "Online parameter estimation for uncertain robot manipulators with fixed-time convergence," in *2020 15th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, IEEE, 2020, pp. 1808-1813, 10.1109/ICIEA48937.2020.9248176.

[17] J. Wang, D. Efimov, S. Aranovskiy, and A. A. Bobtsov, "Fixed-time estimation of parameters for non-persistent excitation," *European Journal of Control*, vol. 55, pp. 24-32, 2020, 10.1016/j.ejcon.2019.07.005.

[19] D. Efimov, S. Aranovskiy, A. A. Bobtsov, and T. Raïssi, "On fixed-time parameter estimation under interval excitation," in *2020 European Control Conference (ECC)*, IEEE, 2020, pp. 246-251, 10.23919/ECC51009.2020.9143735.

[19] H. Ríos, D. Efimov, J. A. Moreno, W. Perruquetti, and J. G. Rueda-Escobedo, "Time-varying parameter identification algorithms: Finite and fixed-time convergence," *IEEE Transactions on Automatic Control*, vol. 62, no. 7, pp. 3671-3678, 2017, 10.1109/TAC.2017.2673413.

[20] G. Ji, C. Hu, J. Yu, H. Jiang,"Finite-time and fixed-time synchronization of discontinuous complex networks: A unified control framework design," *Journal of the Franklin Institute,*, vol. 355, no. 11, pp. 4665-4685, doi:j.jfranklin.2018.04.026, 2018.

[21] F. Kong, Q. Zhu, R. Sakthivel,"Finite-time and fixed-time synchronization control of fuzzy Cohen-Grossberg neural networks," *Fuzzy Sets and Systems,*, vol. 394, no. 11, pp. 87-109, doi:10.1016/j.fss.2019.12.002, 2020.

[22] W. Yang, W. Yu, J. Cao, F.E. Alsaadi, T. Hayat,"Global exponential stability and lag synchronization for delayed memristive fuzzy CohenGrossberg BAM neural networks with impulses," *Neural Network,*, vol. 98, no. , pp. 122–153, doi:10.1016/j.neunet.2017.11.001, 2018.

[23] C. Hu, J. Yu, Z.H. Chen, H.J. Jiang, T.W. Huang,"Fixed-time stability of dynamical systems and fixed-time synchronization of coupled discontinuous neural networks," *Neural Network,*, vol. 89, no. , pp. 74–83, doi: , 2017, 10.1016/j.neunet.2017.02.001.

[24] T. H. Gronwall, "Note on the derivatives with respect to a parameter of the solutions of a system of differential equations." *Annals of Mathematics*, vol. 20, no. 4, pp. 292-296, 1919, doi.org/10.2307/1967124.

[25] F. Sabahi, M. R. Akbarzadeh-T, "A framework for analysis of extended fuzzy logic." *J. Zhejiang Univ. - Sci. C* 15, 584–591 (2014). https://doi.org/10.1631/jzus.C1300217

[26] F. Sabahi, "Fuzzy Adaptive Granulation Multi-Objective Multi-microgrid Energy Management. *Journal of AI and Data Mining*, *8*(4), 481-489. 2020. doi: 10.22044/jadm.2019.6985.1828

[27] A. Polyakov, “Nonlinear Feedback Design for Fixed-Time Stabilization of Linear Control Systems," in IEEE Transactions on Automatic Control, vol. 57, no. 8, pp. 2106-2110, Aug. 2012, doi: 10.1109/TAC.2011.2179869.

[28] J. Zhou, T. Chen, L. Xiang, “Adaptive Synchronization of Delayed Neural Networks Based on Parameters Identification”. In: Wang, J., Liao, X., Yi, Z. (eds) Advances in Neural Networks – ISNN 2005. ISNN 2005. Lecture Notes in Computer Science, vol 3496. Springer, Berlin, Heidelberg. https://doi.org/10.1007/11427391_48

همگام‌سازی پایدار در شبکه‌های عصبی بازگشتی فازی در یک چارچوب زمانی ثابت

فرناز صباحی*

گروه برق، دانشکده برق و کامپیوتر و فناوری‌های پیشرفته، دانشگاه ارومیه، ارومیه، ایران.

ارسال ۲۰۲۴/۱۱/۲۹؛ بازنگری ۲۰۲۴/۱۲/۲۲؛ پذیرش ۲۰۲۴/۱۲/۳۱

چکیده:

این مقاله به بررسی همگام‌سازی در زمان ثابت برای شبکه‌های عصبی بازگشتی فازی با تأخیر و ناپیوسته (DFRNNs) با تأخیرهای متغیر با زمان می‌پردازد. بر اساس یک تبدیل متغیر تعمیم‌یافته، سیستم خطا توسعه داده شده است تا به‌طور مؤثر ناپیوستگی‌ها در سیستم‌های عصبی را مدیریت کند. این پژوهش مشکل پایداری در زمان ثابت را با استفاده از یک ورودی کنترل فیدبک حالت ناپیوسته نوآورانه و یک طرح کنترل تطبیقی ساده‌سازی‌شده مورد بررسی قرار می‌دهد. روش پیشنهادی، همگام‌سازی مقاوم بین سیستم‌های عصبی محرک و پاسخ را در یک زمان ثابت تضمین می‌کند. کاربردهای عملی این تحقیق شامل بهبود پروتکل‌های ارتباطات امن، سیستم‌های کنترل رباتیک و چارچوب‌های کنترل هوشمند بر روی سیستم‌های پویا است. یک مثال عددی ادعاهای نظری را تأیید می‌کند و نقاط قوت روش پیشنهادی را نشان می‌دهد. نتایج، همگرایی خطاها به صفر را در یک زمان ثابت نشان می‌دهند که عملکرد بدون سوگیری را در یک بازه زمانی از پیش تعیین‌شده، مستقل از شرایط اولیه، تضمین می‌کند.

کلمات کلیدی: شبکه عصبی ناپیوسته، همگام سازی در زمان ثابت، تابع لیاپانوف، تاخیرهای متغیر با زمان.