Received 10 May 2023, accepted 22 May 2023, date of publication 25 May 2023, date of current version 6 June 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3279862

RESEARCH ARTICLE

An Optimal Nonlinear Type-2 Fuzzy FOPID Control Design Based on Integral Performance Criteria Using FSM

MOHAMMAD M. AL-MOMANI[1], AMNEH AL-MBAIDEEN[1], ABDULLAH I. AL-ODIENAT[1], KHALED MOHAMMAD ALAWASA[1,2], (Member, IEEE), AND SABA F. AL-GHARAIBEH[1]

[1]Department of Electrical Engineering, Mutah University, Mu'tah 61710, Jordan
[2]Department of Electrical and Computer Engineering, Sultan Qaboos University, Muscat 123, Oman

Corresponding author: Mohammad M. Al-Momani (monqedmohammad@gmail.com)

This work was supported by the Scientific Research and Innovation Support Fund (SRISF), Jordan, under Project ENE/1/3/2020.

ABSTRACT A fractional-order fuzzy proportional integral derivative (PID) controller is a controller that combines the benefits of fractional calculus and fuzzy logic with the conventional PID controller. In this paper, a four-stage optimization algorithm is proposed for the design of a Type-2 Fuzzy fractional-order PID controller based on the Fourier Series Method (FSM). Three distinct control structures are introduced: Type-2 fuzzy fractional PD + fractional PI controller, Type-2 fuzzy fractional PID, and Type-2 fuzzy fractional PD + Type-2 fuzzy fractional PI controller. In addition to a modified multi-performance criterion cost function, four integral performance criteria are employed as cost functions for each stage. The suggested algorithm avoids the utilization of the approximation equivalent for the fractional-order system and instead employs FSM. Furthermore, the approach optimizes the nonlinearity within the upper membership function (UMF) and the uncertainty range through the lower membership function, as opposed to arbitrary selection. By considering variations in the membership functions, the outcomes exhibit a superior response compared to previous investigations. The results of the three control structures are compared with the traditional PID controller, and simulation results demonstrate the feasibility of this technique. The findings suggest that by optimizing different integral performance criteria using this design technique, controllers for both integer and fractional-order plants can yield favorable step responses. The proposed algorithm is validated by comparing its step response performance with that of previous research, followed by a discussion on sensitivity analysis and computational requirements.

INDEX TERMS Fourier series method, fractional-order PID controller, type-2 fuzzy controller.

I. INTRODUCTION

A. THE MOTIVATION OF THE PAPER

Various engineering disciplines are currently striving to identify the optimal controller to achieve the most favorable step response for a given plant. Fuzzy logic, which was introduced to the controller field in the last century as part of artificial intelligence applications and tools, has gained significant traction due to its ability to handle uncertainty in inputs, outputs, and decisions [1]. The two most prevalent types of fuzzy controllers are Mamdani and Sugeno, which differ in their output membership function and the definition of fuzzy logic operators [2]. The latest iteration of fuzzy logic is the fuzzy-fuzzy logic, or type-2 fuzzy, which contains an internally undefined area in its input/output membership functions, resulting in a higher level of uncertainty than traditional fuzzy logic [3]. This research employs the Sugeno type-2 fuzzy controller, which differs from its type-1 counterpart in its input membership functions.

Conversely, the fractional-order PID controller offers two additional degrees of solution freedom, effectively rendering the conventional PID controller as a particular point in the

The associate editor coordinating the review of this manuscript and approving it for publication was Yiming Tang.

two-axis FPID space [4]. This study aims to amalgamate the benefits of fuzzy type two and fractional-order control to create distinct type-2 Fuzzy FPID structures.

B. LITERATURE REVIEW

One of the newest research papers in this field is referenced in [5]. This paper presents a novel approach for designing a controller known as the Interval Type-2 Fractional-order Fuzzy PID (IT2FO-FPID) controller, which utilizes fractional-order integrators and differentiators. The authors combine a Takagi-Sugeno-Kang (TSK) type Interval Type-2 Fuzzy Logic Controller (IT2FLC) with the fractional PID controller to evaluate the controller's performance in terms of both unit step response and unit load disturbance. The IT2FO-FPID controller is tested on various benchmark plants with time delays and nonlinearities, and its robustness is analyzed. The design parameters of the controller include fractional-order integrator-differentiator operators and input-output scaling factors.

To optimize these parameters, an optimization algorithm called the Artificial Bee Colony-Genetic Algorithm (ABC-GA) is employed. The optimization process minimizes a weighted sum of the integral of time absolute error (ITAE) and the integral of the square of the control output (ISCO). Simulation results demonstrate that the IT2FO-FPID controller outperforms other controllers in most cases. However, it is important to note that this paper does not take into account the impact of nonlinearity on the fuzzy controller, the lower membership functions of the type-2 fuzzy are chosen arbitrarily, and the Oustaloup approximation is utilized to address the fractional-order controller. Furthermore, the optimization of fuzzy membership functions and set rules is not addressed in this paper, as the focus is solely on optimizing the fractional-order PID parameters. Similar observations can be made regarding the papers referenced as [6], [7], [8], and [9].

In [6], the proposed fuzzy logic controller (FLC) incorporates fractional-order differ-integrations as design variables, along with input-output scaling factors (SF), which are optimized using Genetic Algorithm (GA). The objective function aims to minimize several integral error indices along with the control signal. However, the nonlinearity in the fuzzy controller is ignored, and the Oustaloup approximation is used to create an equivalent for the fractional-order.

Researchers in [7] present the development of a new power system stabilizer (PSS) based on a type-2 fuzzy fractional-order PID controller. The goal is to enhance the dynamic stability of the power system by improving its electromechanical oscillation damping performance. To achieve this, a hybrid algorithm that combines meta-heuristic techniques is utilized for the controller design. All previous articles [5], [6], [7], [8], [9] employ the same control structure, namely the fuzzy fractional PD + fractional PI controller (FFPD-FPI).

The utilization of FFPD-FPI to enhance the dynamic stability of power systems incorporating renewable resources is introduced in [7] and [9]. In [8], the proposed FFPD-FPI controller is applied to a tractor active suspension system. Based on a comprehensive literature review, it is evident that most researchers tend to arbitrarily select the internal membership functions of type-two fuzzy sets. Unfortunately, they often overlook the impact of nonlinearity on these membership functions. Furthermore, in many cases, approximation models are employed to represent fractional-order plants [5], [6], [7], [8], [9], [10].

In reference [12], the authors propose a novel approach for frequency regulation in a multi-micro-grid (MMG) system. The focus is on load frequency control and the integration of renewable energy sources. They introduce a specific fractional-order fuzzy PID (FOFPID) controller designed for efficient frequency control. The parameters of the FOFPID controller are optimized using an enhanced version of the Harris Hawks optimizer (mHHO), which outperforms other commonly used optimization techniques in this field.

In reference [13], researchers present the design of an adaptive fuzzy type-2 fractional-order PID sliding mode controller for trajectory tracking of robotic manipulators within a reduced task space. By combining the flexibility of fractional-order controllers with the effectiveness of PID controllers, the proposed controller addresses uncertainties and provides a fast and accurate response. The selection of a suitable Lyapunov function ensures global asymptotic stability, regardless of the initial conditions. The study aims to enhance the performance of robotic manipulators by enabling precise trajectory tracking in a limited task space.

Reference [14] introduces a family of hybrid interval type-2 fractional-order fuzzy PID (IT2FO-FPID) controllers. These controllers incorporate fractional-order integro-differentiators to enhance their performance. Evaluation is done using time response measures for unit set-point response and unit load disturbance. The controllers are tested on various fractional-order processes, and their robustness is analyzed. The results demonstrate that the inclusion of fractional-order integro-differential operators reduces sensitivity to parameter changes, making the controllers more effective. The design approach considers the fractional-order integro-differential operators and input/output scaling factors as variables for developing efficient and adaptable controllers.

On the other hand, the controllers described in references [15] and [16] utilize optimization techniques for the membership functions. Reference [15] employs linear membership functions, while reference [16] distributes the membership functions non-uniformly around the origin. However, it should be noted that these controllers rely on the type-1 fuzzy PID controller and do not fully exploit the advantages of fractional-order control or incorporate the uncertainty footprint.

C. STUDY GAPS

Previous studies have presented various hybrid controllers based on Fuzzy and FOPID techniques. However, there are certain gaps in the existing literature regarding the design of these controllers. Specifically, when considering the membership functions of the type-1 fuzzy controller (or UMF of type-2 fuzzy), most algorithms focus solely on optimizing the PID or FOPID parameters and employ linear (triangular) membership functions that are uniformly distributed around the operating point [5], [6], [7], [8], [9], [10], [13], [14], [15], [30], [31], [33], [36], [38]. These studies assume that optimizing the PID or FOPID control parameters is sufficient to optimize the entire controller, disregarding the impact of the membership functions. However, the significance of the membership function, particularly its distribution around the origin, on the overall controller performance has been demonstrated in [15] and [16].

In the context of the type-2 fuzzy controller, the lower membership function plays a crucial role in quantifying the level of uncertainty inherent in the controller. However, a significant number of researchers opt to select these membership functions arbitrarily, without a systematic approach or justification [5], [7], [9], [13], [14], [31], [33]. On the other hand, a few researchers have recognized the importance of optimizing the lower membership function [8], [10]. Nevertheless, their optimization efforts have been limited to scenarios where a linear upper membership function with uniform distribution is employed. The exploration and optimization of the lower membership function in conjunction with various upper membership functions remain unexplored areas within the existing literature.

Another notable gap identified in the existing literature pertains to the treatment of fractional-order parameters within FOPID (Fractional-order Proportional Integral Derivative) controllers. A majority of the algorithms proposed in the literature resort to approximating the fractional parameters by mapping them to an integer-transfer function that closely resembles the desired response [5], [6], [7], [8], [9], [10], [13], [14], [30], [31], [33], [36], [37], [38]. This approach overlooks the inherent characteristics and advantages offered by fractional-order dynamics, thereby limiting the true potential and efficacy of FOPID controllers. The need for novel methodologies that effectively handle and incorporate the fractional-order parameters in the design and optimization of FOPID controllers is evident in the current state of research.

The proposed algorithm presented in this study aims to address the three identified gaps in the existing literature. Firstly, it focuses on the optimization of the nonlinearity of the membership functions, considering their significance in controller performance. Secondly, the algorithm seeks to optimize the lower membership function to enhance the controller's performance. Lastly, the proposed algorithm aims to overcome the gap concerning the handling of fractional-order parameters in FOPID controllers. Instead of approximating the fractional parameters to an integer-transfer function, this algorithm embraces the true nature of fractional-order dynamics, allowing for more accurate and effective control using Fourier series method. To provide a comprehensive overview of the progress made in addressing these gaps, Table 1 summarizes the existing research in terms of these three aspects, highlighting the limitations and areas that require further exploration.

D. CONTRIBUTION AND PAPER STRUCTURE

This paper presents a novel design approach for optimizing a nonlinear type-2 fuzzy fractional-order controller, eliminating the need for fractional-order approximation by utilizing the Fourier series method. Additionally, the paper introduces three distinct structures of hybrid type-2 fuzzy and fractional-order PID controllers. Furthermore, the Teach-Learning-Based Optimization (TLBO) algorithm is modified to effectively handle the type-2 fuzzy parameters, the footprint of uncertainty, and the membership nonlinearity.

The results of the various controller structures are compared using a consistent design process. Furthermore, the paper highlights the impact of the nonlinearity in membership functions, the footprint of uncertainty, and the fractional-order on the step response of the plant.

Finally, this algorithm is compared with recently published algorithms for various types of controllers, including high-order processes, time-delay processes, and fractional-order processes. The comparison aims to evaluate the performance and effectiveness of the proposed algorithm in these different scenarios.

This paper is structured into six sections. Section I serves as the introduction, providing an overview of the research topic and its significance. Section II delves into the theoretical background, presenting the fundamental concepts of fractional-order calculus theory and the type-2 fuzzy controller. Additionally, in section II, the overall transfer function of the various controller structures is discussed. In Section III, the paper outlines a proposed four-step design procedure for the controller. The steps are described in detail, providing a comprehensive understanding of the design process. Section IV presents the simulation results obtained from applying the different controller structures to two benchmark plants. The performance and effectiveness of each structure are analyzed and compared. Algorithm validation is presented in Section V, where the step performance of the proposed algorithm is compared with previous research. Sensitivity analysis and computational time response are also discussed, evaluating the robustness and efficiency of the algorithm. Finally, in Section VI, the paper concludes by summarizing the key findings and insights gained from the research. Concluding remarks are provided, highlighting the contributions of the study and potential areas for future research.

II. CONTROLLER MATHEMATICAL MODEL

In this section, we present the linear and nonlinear fuzzy controllers, which are based on the fuzzy type-two system. We also discuss the fundamentals of fractional calculus and provide the transfer function for the different control structures. The Sugeno fuzzy controller is chosen for this project due to its simplicity in the defuzzification process of the fuzzy type-two system.

TABLE 1. Different studies in hybrid fractional-order PID-Fuzzy controllers.

Ref.	year	Controller used		Fractional-order implementation		upper membership function (UMF)				lower membership function (LMF)	
		Fuzzy type	PID	Exact	Approx.	Linear	Nonlinear	Optimized	randomly	Optimized	randomly
[5]	2017	2	FOPID		√	√			√		√
[6]	2012	1	FOPID		√	√			√	-	-
[7]	2019	2	FOPID		√	√			√		√
[8]	2019	2	FOPID		√	√			√	√	
[9]	2019	2	FOPID		√	√			√		√
[10]	2020	2	FOPID		√	√			√	√	
[12]	2023	-	FOPID	√		-	-	-	-	-	-
[13]	2020	2	FOPID		√	√			√		√
[14]	2018	2	FOPID		√	√			√		√
[15]	2017	1	PID			√		√		-	-
[16]	2019	1	PID	-	-		√	√		-	-
[20]	2008	1	-	-	-		√	√			-
[33]	2008	-	FOPID		√	√			√		√
[28]	2017	-	FOPID	√		-	-	-	-	-	-
[34]	2010	-	FOPID	√		-	-	-	-	-	-
[36]	2018	1	FOPID		√	√		-	-	-	-
[37]	2021	1	FOPID		√		√		√	-	-
[38]	2022	1	FOPID		√	√			√	-	-
[42]	2021	-	FOPID	√		-	-	-	-	-	-
[30]	2023	1	FOPID		√	√			√	-	-
[31]	2023	2	FOPID		√	√			√		√
Proposed		2	FOPID	√			√	√		√	

A. SUGENO FUZZY TYPE-2 CONTROLLER

The Sugeno type-2 fuzzy controller can be represented by a linear transfer function if certain conditions are met [15]:

- The triangular membership functions for all inputs have a uniform distribution.
- A complete set of if-then rules follows a linear transition between membership functions.

In this study, all membership functions will be distributed from −1 to 1. The FPID parameters will be sufficient to determine the optimal operation point for the controller. Therefore, there is no need for test data or precise knowledge about the plant. Figure 1 illustrates the membership functions of two inputs and one output for the linear Sugeno fuzzy set. The space between the centers of the inputs' membership functions is uniform, and the step size of the output membership functions is constant. The complete set of if-then rules is presented in Table 2.

Utilizing the aforementioned linear system, the functional correlation between the system's inputs (E(t), E'(t)) and the output (y(t)) can be explicitly expressed as follows [15]:

$$y(t) = \frac{1}{2}E(t) + \frac{1}{2}E'(t) \qquad (1)$$

The nonlinearity inherent in the membership function can be denoted by a factor that exhibits nonlinearity, as indicated by the findings outlined in reference [16].

$$\zeta_j = \frac{\varepsilon_i - \varepsilon_{i-1}}{\varepsilon_{i-1} - \varepsilon_{i-2}} \qquad (2)$$

This nonlinearity factor, denoted as ζ_j, is associated with the jth input (or output) and is linked to the center of the ith membership function, represented by ε_i. Figure 2 illustrates the nonlinear membership functions, wherein it is discernible that the nonlinear factor, ζ, characterizes the ratio of center distances among the membership functions prior to and subsequent to the point in consideration. It is notable that the value of ζ can be less than one, indicating that the membership functions in close proximity to zero shall be broader. Conversely, if ζ surpasses unity, the MF closest to zero will display a more narrow shape. Each membership function may demonstrate an asymmetrical shape, which can substantially affect the system response, as dictated by the nonlinearity factor's definition.

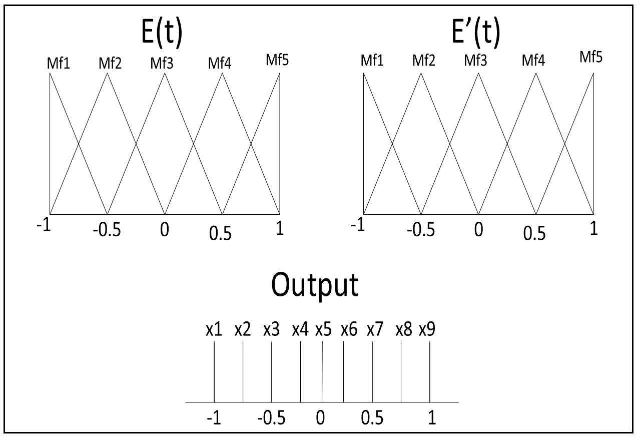

FIGURE 1. Linear membership functions of two-input one-output Sugeno-type fuzzy system.

TABLE 2. Fuzzy set rules of five membership functions.

E(t)/E'(t)	Mf 1	Mf 2	Mf 3	Mf 4	Mf 5
Mf 1	x1	x2	x3	x4	x5
Mf 2	x2	x3	x4	x5	x6
Mf 3	x3	x4	x5	x6	x7
Mf 4	x4	x5	x6	x7	x8
Mf 5	x5	x6	x7	x8	x9

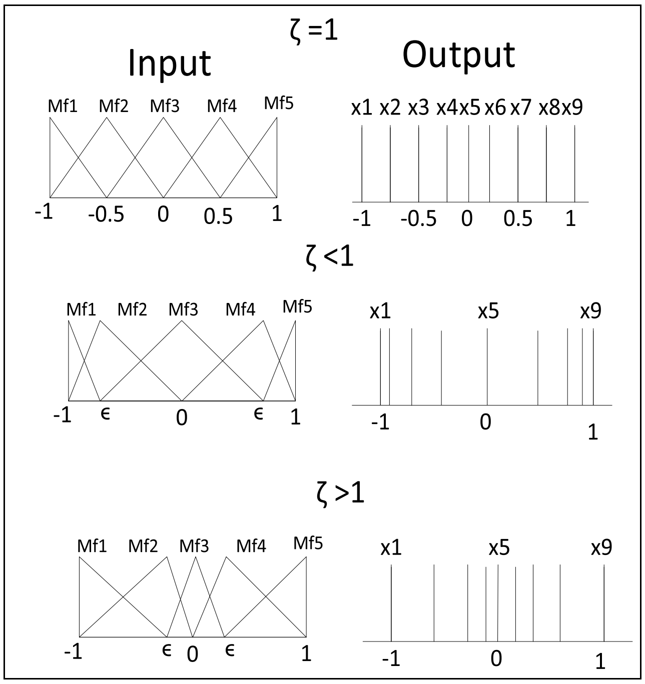

FIGURE 2. Nonlinear membership function; effect of nonlinearity factor on input and output membership functions.

To address uncertainty and imprecision more effectively, the utilization of fuzzy sets, specifically type-2 fuzzy sets, was implemented. The membership degree of the fuzzy sets was assigned to a type-2 membership function denoted as $\mathbf{0} < \mu_{\tilde{A}}(\boldsymbol{x}, \boldsymbol{u}) < \mathbf{1}$ (with $\boldsymbol{x} \in \boldsymbol{X}$ and $\boldsymbol{u} \in \boldsymbol{J_x} \subseteq [\mathbf{0}; \mathbf{1}]$), were defined in two distinct definitions [3].

$$\begin{aligned}\tilde{A} &= \left\{\left(x, \mu_{\tilde{A}}(x)\right) \mid x \in X\right\} \\ &= \left\{x, u, \mu_{\tilde{A}}(x, u) \mid \forall x \in X, \forall u \in J_x \subseteq [0, 1]\right\}\end{aligned} \tag{3}$$

Or

$$\tilde{\boldsymbol{A}} = \left\{\int_{x \in X}\left[\int_{u \in J_x^u \subseteq [0,1]} \boldsymbol{f_u}(\boldsymbol{u}) / \boldsymbol{u}\right] / \boldsymbol{x}\right\} \tag{4}$$

where $\int$: union over x and u. xi and μi represents the centroid of the consequent and the firing level of the ith rule in type-2 fuzzy set.

This definition may be depicted by two membership functions, namely the Upper Membership Function (UMF) and the Lower Membership Function (LMF), as delineated in reference [17]. The FOU (footprint of uncertainty) denotes the interval separating these membership functions and serves as a crucial factor in characterizing the new membership function definition, as depicted in Figure 3.

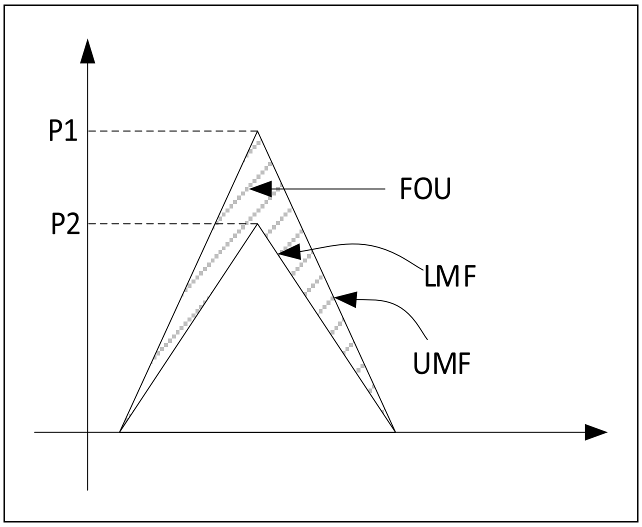

FIGURE 3. Fuzzy-II input membership function; definition of footprint of uncertainty (FOU), upper membership function (UMF) and lower membership function (LMF).

The Footprint of Uncertainty (FOU) is a key feature of type-2 fuzzy sets that serves to capture the inherent uncertainty within a system. This interval quantifies the degree of uncertainty that arises from the imprecision in defining the membership function, and reflects the range of possible values for the membership degree of an element in the fuzzy set.

The incorporation of the FOU in fuzzy control systems provides an opportunity to enhance accuracy by addressing uncertainty more comprehensively. In conventional fuzzy control systems utilizing type-1 fuzzy sets, the membership degree of an element is represented by a single value, which may not fully capture the true uncertainty of the system. By employing type-2 fuzzy sets with a defined FOU, the system can incorporate a more complete understanding of the range of possible values, leading to more accurate and robust control.

The membership function of the output in a type-2 fuzzy Sugeno system is defined as a weighted linear combination of the crisp output values generated by the individual rules of the system. For a given input combination, each rule is evaluated, resulting in a crisp output value. These crisp output values are then weighted according to the corresponding type-2 membership function associated with each rule. The weighted output values are aggregated using a weighted average approach, yielding the final system output.

The type-2 membership functions associated with each rule capture the uncertainty inherent in the crisp output value obtained from that rule. They enable the incorporation of imprecision and uncertainty in the system's output, thereby providing a more comprehensive representation of the overall system behavior.

In a Sugeno type-2 fuzzy system, the defuzzification process is used to convert the fuzzy output into a crisp output that can be used to control the system. The center of gravity (COG) defuzzification method is commonly employed to achieve this objective. In this process, the fuzzy output of the system is initially converted into a type-1 fuzzy set by computing the weighted average of the type-2 membership functions associated with each rule. This calculation is performed according to Equation 5 [19]. As a result, a set of type-1 fuzzy output values is obtained, where each value corresponds to a specific rule.

$$\begin{aligned} y &= \frac{c_L + c_R}{2} \\ C_L &= \frac{\sum_{i=1}^{L} x_i\mu_i + \sum_{i=L+1}^{N} \alpha x_i\mu_i}{\sum_{i=1}^{L} \mu_i + \sum_{i=L+1}^{N} \alpha\mu_i} \\ C_R &= \frac{\sum_{i=1}^{R} \alpha x_i\mu_i + \sum_{i=R+1}^{N} x_i\mu_i}{\sum_{i=1}^{R} \alpha\mu_i + \sum_{i=R+1}^{N} \mu_i} \end{aligned} \tag{5}$$

where xi and μi represents the centroid of the consequent and the firing level of the ith rule in type-2 fuzzy set. In the context of the Sugeno type-2 fuzzy system, the defuzzification process relies on the estimation of switch points denoted as L and R. These switch points are determined using various type-reduction methods [18], [19], [20], such as the Enhanced Karnik Mendel (EKM) method employed in this study [19].

The ratio α, defined as P_2/P_1, represents the proportion of the lower membership function to the upper membership function. The center of gravity defuzzification method, applied to Sugeno type fuzzy type-2 systems, has been demonstrated to effectively address uncertainty and imprecision in control systems.

By utilizing type-2 fuzzy sets in conjunction with the center of gravity defuzzification process, the control system can incorporate a more comprehensive understanding of the system's uncertainty, resulting in improved robustness and accuracy of control.

Towards the conclusion of this section, it is essential to note that the optimization of ζs (for nonlinearity) and αs (for FOU) necessitates two distinct types of parameters. These parameters are incorporated and designed in the third step of the design procedure, which is expounded upon in Section III. Where the first two steps entail the assumption of α and ζ as linear type-1 fuzzy.

B. FRACTIONAL-ORDER PID CONTROLLER

A fractional-order PID (Proportional-Integral-Derivative) controller, denoted as $PI^{\lambda}D^{\mu}$, represents an advanced version of the traditional PID controller that has gained significant attention in recent years. In contrast to the conventional PID controller, which solely employs integer-order differentiation and integration, the fractional-order PID controller incorporates fractional calculus in its design, offering enhanced flexibility and control across various applications [4]. A notable characteristic of the fractional-order PID controller is its capability to accurately model the dynamics of a system, including those exhibiting non-integer order behaviors. The fractional-order parameter of the controller is determined by the system's time constant, enabling additional control capabilities that enhance the system's efficiency and effectiveness. Furthermore, the flexibility of the fractional-order PID controller allows for real-time adjustment of its parameters to accommodate the changing dynamics of the system. This adaptability makes it particularly well suited for systems operating under varying loads or environmental conditions. Therefore, the fractional-order PID controller has emerged as a powerful tool in the field of control systems engineering, owing to its advanced modeling capabilities, increased control flexibility, and adaptability to changing dynamics [4].

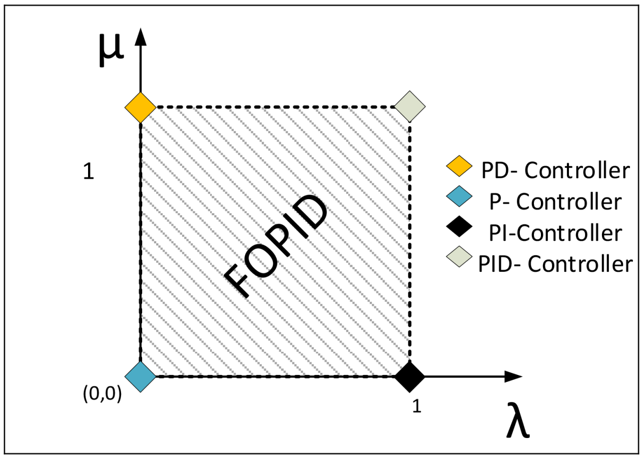

FIGURE 4. The relation between fractional-order PID (FPID) and PID controllers; different values of λ and μ.

This controller has been extensively investigated in both the time and frequency domains [21], [22]. The transfer function of the controller can be represented in the frequency domain by equation 6, while in the time domain, it is expressed by equation 7.

$$G(s) = K_p + K_i s^{-\lambda} + K_d s^{\mu} \tag{6}$$

$$u(t) = K_p e(t) + K_i D^{-\lambda} e(t) + K_d D^{\mu} e(t) \tag{7}$$

where D is a partial differ-integral operator, λ and μ are fractional-orders. Based on these equations, the FPID controller is a generalized form of the PID controller, where the values of λ and μ determine its characteristics. Figure 4 illustrates the relationship between the P, PI, PID, and FPID controllers. The integer controllers are associated with the square corner, representing P, PI, PD, and PID controllers, while the fractional controllers can be positioned at any point inside or outside the square, highlighting their enhanced flexibility.

The partial differ-integral operator $({}_{\alpha}D_t^{r})$ in Equation 7 is a mathematical operator that combines both differentiation and integration within a single operation. It serves as a generalization of traditional differentiation and integration and enables the utilization of non-integer orders of these operations. This operator finds application in diverse fields of science and engineering, including signal processing and fractional calculus [23]. The operator can be applied to a function f(t) to produce a new function that represents the partial derivative of order r and partial integral of order alpha of f(t).

$$ {}_{\alpha}D_t^r = \begin{cases} d^r/dt^r & r > 0 \\ 1 & r = 0 \\ \int_{\alpha}^{t} (d\tau)^r & r < 0 \end{cases} \tag{8} $$

This operator has many properties, including linearity, commutativity, and associativity. Furthermore, it can be inverted, facilitating the computation of anti-derivatives and partial derivatives [23]. The fractional calculus operator can be defined using two common approaches: the Grunwald-Letnikov (GL) method and the Riemann-Liouville (RL) method [24], [25].

The Grunwald-Letnikov operator is a discretized approximation of the fractional derivative. It is defined as the limit of a finite difference approximation as the step size approaches zero. One advantage of this operator is its computational simplicity, but its accuracy is constrained by the selection of the step size. [24]

$$ {}_{\alpha}D_t^r f(t) = \lim_{x\to 0} x^{-r} \sum_{k=0}^{\left[\frac{t-a}{x}\right]} (-1)^k \binom{r}{k} f(t-kx) \tag{9} $$

The Riemann-Liouville operator, on the other hand, provides a continuous definition of the fractional derivative. It extends the concept of integer-order derivative to non-integer orders through the utilization of fractional integration. Unlike the Grunwald-Letnikov operator, the Riemann-Liouville operator offers improved accuracy. However, it is computationally more challenging to compute. The RL definition relies on the gamma function, as expressed below:

$$ {}_{\alpha}D_t^r f(t) = \frac{1}{\Gamma(n-r)} \frac{d^n}{dt^n} \int_{\alpha}^{t} \frac{f(\tau)}{(t-\tau)^{r-n+1}} d\tau \tag{10} $$

for $(n - 1 < r < n)$ and $\Gamma(:)$ is the gamma function.

In fractional calculus, both the Grunwald-Letnikov and Riemann-Liouville operators are employed to extend differentiation and integration to non-integer orders.

C. DIFFERENT CONTROLLER STRUCTURES

This project encompasses three distinct structures: the first structure is the Fuzzy PD-Fuzzy PI controller with fractional-order (FFPD-FPI), as depicted in Figure 5(a); the second structure is the Fractional Fuzzy PID controller (FFPID), illustrated in Figure 5(b); and the third structure is the Fractional Fuzzy PI-Fractional Fuzzy PD controller (FFPI-FFPD) with parallel fuzzy controllers, shown in Figure 5(c).

This research article focuses on the design of hybrid fuzzy-fractional Proportional-Integral-Derivative (PID) structures, which integrate both fuzzy logic and fractional calculus concepts. The primary objective of these structures is to enhance control and stability in diverse systems. The article conducts a comparative analysis to evaluate the performance of three such structures. By comparing the results obtained from different structures, the article aims to identify the structure that exhibits the highest level of stability and control. Additionally, the article offers insights into the effectiveness of these hybrid structures, considering the impacts of the Footprint of Uncertainty (FOU), nonlinearity fuzzy factors, and fractional components.

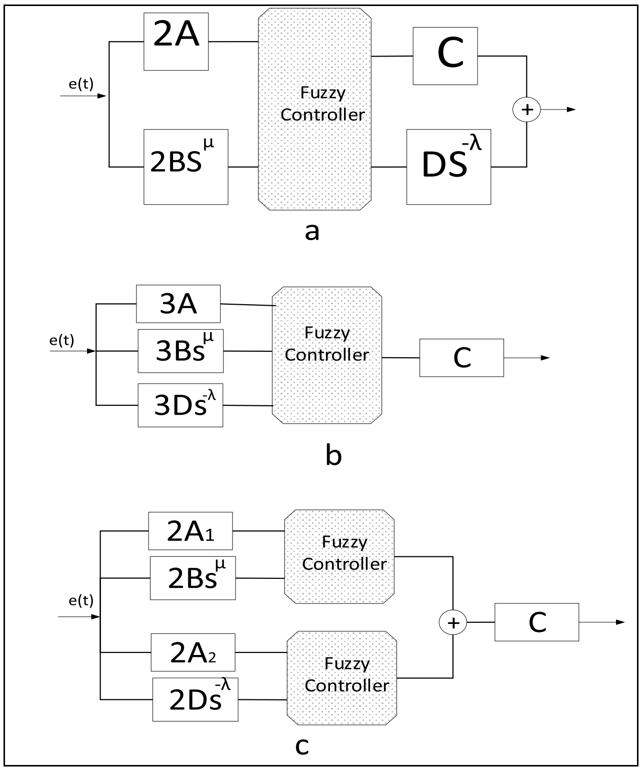

FIGURE 5. Different controller structures. (a) Fractional Fuzzy PD- Fractional PI (FFPD-FPI) controller. (b) Fractional Fuzzy PID (FFPID) controller. (c) Fractional Fuzzy PD-Fractional Fuzzy PI (FFPD-FFPI) controller.

III. PROPOSED DESIGN PROCESS

The design process proposed for hybrid fuzzy-fractional-PID structures comprises four steps outlined in Figure 6. Firstly, the linear type-1 fuzzy controller with PID components is optimized using the "fminsearch" function in MATLAB. Next, the impact of fractional-orders is incorporated into the design. Thirdly, the parameters of the nonlinear fuzzy type-1 controller are optimized using the modified Teaching-Learning-Based Optimization (TLBO) algorithm. Finally, the uncertainty footprint of the fuzzy type-2 membership functions is optimized using the same algorithm. The initial points for each step are determined based on the results obtained from the preceding step.

A. STEP 1: LINEAR TYPE-1 FUZZY PID

At this stage of the study, a linear type-1 fuzzy controller is employed, characterized by input/output membership

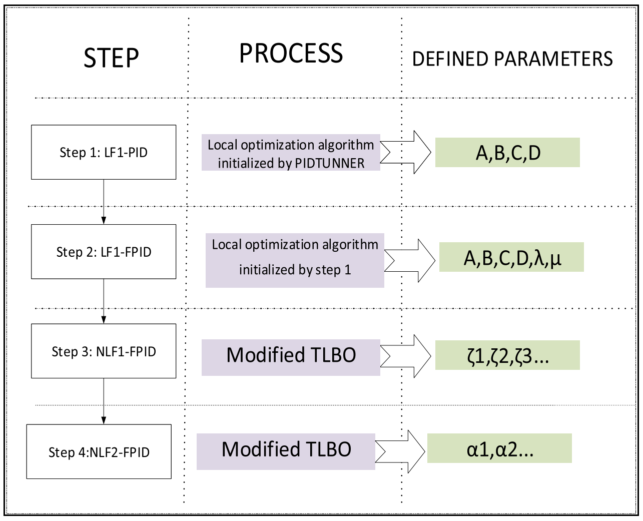

FIGURE 6. **Four-step design process.**

functions depicted in Figure 1. The influence of fractional-order is disregarded in this phase, with λ and μ values set to 1. Fuzzy rules are designed to maintain linearity, as described in Equation 1, and are presented in Table 2. This step serves as a foundation for the subsequent optimization and design of the hybrid fuzzy-fractional-PID structures.

The transfer functions for each of the three structures (FFPD-FPI, FFPID, FFPD-FFPI) are expressed as follows, with ζ, α, λ, and μ all set to 1.

$$C_1(s) = AC + AD/\mathrm{s} + BCs + BD \tag{11}$$
$$C_2(s) = AC + BCs + DC/\mathrm{s} \tag{12}$$
$$C_3(s) = (A_1 + A_2)C + BCs + DC/\mathrm{s} \tag{13}$$

An integral performance criterion is defined in a general form as,

$$J_n = \int_0^\infty \left[t^n e(t)\right]^2 dt \tag{14}$$

where n = 0, n = 1, and n = 2 are utilized to represent the integral squared error (ISE), integral time squared error (ITSE), and integral squared time squared error (IT2SE), respectively. Additionally, the integral absolute error (IAE) is utilized as a cost function. A new multi-performance criterion is also introduced as follows:

$$MPC = 0.4 \times ISE + 0.2 \times ITSE + 0.4 \times IAE \tag{15}$$

The term MPC represents the multi-performance criterion, which is used as one of the objective functions. Therefore, the optimization of control parameters (A, B, C, D) is carried out based on five objective functions (ISE, ITSE, IT2SE, IAE, and MPC). The ISE criterion is evaluated using the Laplace transform of the error signal and Parseval's theorem [26], as given by Equation (16). The solution to this equation can be obtained analytically, as explained in [27].

$$J_0 = \int_0^\infty [e(t)]^2\, dt$$
$$ISE = \frac{1}{2j\pi}\int_{-j\infty}^{j\infty} E(s) * E(-s)\, ds \tag{16}$$

The aim of minimizing the ISE (Integral Squared Error) in Equation 16 is to set the initial values of control parameters A, B, C, and D for the remaining performance criteria. On the other hand, the "PIDRUNNER" tool in MATLAB can be employed for the sake of simplicity to initialize the searching algorithm "FMINSEARCH."

B. STEP 2: LINEAR TYPE-1 FUZZY FPID

In reference [28], it is shown that the step response of the fractional-order transfer function can be represented using the Fourier series. The Fourier series is a mathematical technique used to represent a periodic function as a sum of sine and cosine functions. By employing this method, the step response of the fractional-order transfer function can be decomposed into an infinite sum of sine and cosine terms, enabling its accurate representation. This technique finds widespread application in control systems analysis and design.

$$y(t) = \frac{4}{\pi}\sum_{k=1(odd)}^{\infty}\frac{1}{k}Real(TF(jkw_s))\sin(kw_s t) \tag{17}$$

In the given equation, TF represents the transfer function of the closed-loop control system, and w_s denotes the frequency of the square wave. The values for the step response, square wave frequency (lower frequency), and higher frequency were obtained from [28] to be applied in this study. For a standard second-order underdamped plant, as illustrated in equation 18, the real part of the closed-loop transfer function of the first structure (FFPD-FPI) can be expressed using equation 19.

$$G(s) = \frac{K}{s^2 + 2hw_n s + w_n^2} \tag{18}$$

Here, K represents the open-loop gain, h denotes the damping ratio, and w_n represents the natural frequency of oscillation in (19), as shown at the bottom of the next page.

The use of the equation $(j)^\alpha = \cos((\pi/2)\alpha) + jsin((\pi/2)\alpha)$ can precisely handle the fractional-order without any approximations. Here, M and N are given by:

$$\begin{aligned} M &= KAC + KBCw^\mu \cos\left(\frac{\pi}{2}\mu\right) \\ &\quad + KBDw^{\mu-\lambda}\cos\left(\frac{\pi}{2}(\mu-\lambda)\right) \\ &\quad + KADw^{-\lambda}\cos\left(-\frac{\pi}{2}\lambda\right) \\ N &= KBCw^\mu \sin\left(\frac{\pi}{2}\mu\right) + KBDw^{\mu-\lambda}\sin\left(\frac{\pi}{2}(\mu-\lambda)\right) \\ &\quad + KADw^{-\lambda}\sin\left(-\frac{\pi}{2}\lambda\right),\ and\ w = kw_s \end{aligned}$$

Analogously, the real components of the closed-loop transfer functions for the second and third controller structures can be determined. The values for A, B, C, D, λ, and μ are optimized based on equations 14-15, where e(t) = 1-y(t). In this study, the optimization technique employed is "Fminsearch", with the initial point set as the output of the first step, and λ and μ are both assigned a value of one. The optimized values obtained from this step are utilized as the starting point for the Modified Teaching-Learning-Based Optimization (MTLBO) algorithm in the subsequent steps.

C. STEP 3: NONLINEAR TYPE-1 FUZZY PID

In this step, the focus is on optimizing the nonlinearity present in the input-output membership functions of the fuzzy controller. This nonlinearity is illustrated in Figure 2, where the nonlinearity factors are defined by equation (2). To optimize the nonlinearity factors for all input/output membership functions, a modified Teaching Learning Based Optimization (MTLBO) algorithm is proposed. This algorithm aims to fine-tune the nonlinearity factors for improved performance of the fuzzy controller.

Teaching-Learning Based Optimization (TLBO) is a population-based optimization algorithm inspired by the learning process of students and teachers in a classroom. In the TLBO algorithm, each individual in the population represents a student, and the best individual is considered as the teacher. The algorithm consists of two main phases: the teaching phase and the learning phase.

In the teaching phase, the teacher shares its knowledge with the students by moving each student closer to itself. This is done by updating the student's position using the following equation:

$$x_i(t+1) = x_i(t) + r * (teacher(t) - x_i(t)) \tag{20}$$

where $x_i(t)$ is the position of the ith student at time t, teacher (t) is the position of the teacher at time t, and r is a random number between 0 and 1.

In the learning phase, the students learn from each other by exchanging information. This is done by randomly selecting two students and updating their positions as follows:

$$x_i(t+1) = x_i(t) + r * (x_j(t) - x_k(t)) \tag{21}$$

where $x_j(t)$ and $x_k(t)$ are the positions of the randomly selected students, and r is a random number between 0 and 1.

In this investigation, a new tier of teaching and learning has been introduced to enhance the rate at which the algorithm converges. This tier is referred to as the assistance level and functions within subgroup levels. At the assistance level, the population is sorted according to the cost value and partitioned into subgroups. The most favorable solution for each subgroup serves as a teaching assistant based on the same previous equations, resulting in an acceleration of convergence. The number of subgroups is randomly selected during each iteration to diminish the risk of being confined to local solutions. The fundamental purpose of this revised phase is to increase the convergence speed and elevate the probability of obtaining a global solution, thereby allowing the best solution to be acquired with fewer iterations.

D. STEP 4: NONLINEAR TYPE-2 FUZZY-FPID

In this step, the goal is to optimize the footprint of uncertainty for the type-2 fuzzy controller. The uncertainty footprint pertains to the input membership functions, which are shown in Figure 3. In a Sugeno-type fuzzy controller, the output membership functions for the type-2 fuzzy controller are the same as the type-1 fuzzy membership functions. The effect of the lower membership function is defined by equation (5). In this step, the Modified Teaching Learning Based Optimization (MTLBO) algorithm is utilized to optimize the uncertainty factors ($\alpha 1, \alpha 2, \ldots$).

IV. TEST BENCH PROCESS APPLICATIONS

In this section, two benchmark process plants are chosen to validate the efficacy of the proposed controller design. The first plant is modeled as an integer-order transfer function with three poles, while the second plant is represented by a fractional-order transfer function. These plants are selected for their complexity and practical relevance in the field of process control, providing a robust validation of the proposed controller design approach.

A. EXAMPLE 1: INTEGER ORDER PLANT

The following test bench process plant is an example of a three-pole underdamped integer-order plant [29].

$$G(s) = \frac{9}{(s+1)(s^2+2s+9)} \tag{22}$$

The proposed four-step design procedure is applied to design an optimal controller for this plant in the following subsections. The detailed results of each stage are presented. MATLAB code for this example is attached in Appendix I.

1) STEP ONE: LF1-PID CONTROLLER

In this step, the integral squared error criterion for the first structure (FFPD-FPI) used to initialize the different integral criteria is written in Equation 23. Similarly, the ISE0 for the second and third structures is calculated.

$$\begin{aligned} ISE\_0 = {} & (27ABCD(AC+BD+1)+AB(61-6AC-6BD) \\ & - 27(AC+BD)+84BC+72)/(3(AC+BD)\wedge 2 \\ & + 17(AC+BD)+3AD \\ & + 9BC(AC+BD+1)+14) \end{aligned} \tag{23}$$

$$Real\,(TF) = \frac{M^2 - Mw^2 + Mw_n^2 + N^2 + 2hw_nN}{\left(M-w^2\right)^2 + w_n^4 + 2w_n^2\left(M-w^2\right) + N^2 + 4Nhww_n + 4h^2w^2w_n^2} \tag{19}$$

TABLE 3. Fitness values, LF1-PID.

Performance criteria	Structure 1 (FFPD-FPI)	Structure 2 (FFPID)	Structure 3 (FFPI-FFPD)
ISE	0.3349	0.2861	0.2861
ITSE	0.1469	0.1177	0.1177
IT2SE	0.6641	0.6633	0.6633
IAE	0.8503	0.6960	0.6960
MPC	0.5692	0.4590	0.4590

TABLE 4. Step response information, LF1-PID.

Str.	**PC**	**Tr**	**Ts**	**OS (%)**	Peak time	**ess**
	PID	0.8495	4.2798	0.4602	4.7092	3.573e-6
Structure 1	ISE	0.1044	7.7407	**62.6063**	0.2859	7.239e-4
	ITSE	0.8287	4.0025	**7.1332**	3.4042	7.563e-6
	IT2SE	1.0650	4.0464	4.7601	3.4921	1.393e-5
	IAE	0.4061	4.7843	**10.6426**	0.7958	1.717e-5
	MPC	0.4082	4.8083	**11.4510**	0.8073	1.53e-5
Structure 2,3	ISE	0.1182	10.7515	**55.9498**	0.3172	2.812e-3
	ITSE	0.6387	3.5809	**10.9285**	2.9365	1.934e-6
	IT2SE	1.0007	3.9276	4.7170	3.4390	1.508e-6
	IAE	0.4114	4.9539	**20.2678**	0.9003	7.457e-7
	MPC	0.4115	4.9523	**19.8028**	0.8969	7.795e-7

The initial values of A, B, C, and D are calculated based on PIDTUNER results in MATLAB, as follows:

Structure 1: $A = 1, B = K_i, C = K_p\sqrt{K_p^2 - 4K_iK_d}$, $D = \frac{K_d}{C}$.

Structure 2: $A = K_p, B = K_d, C = 1, D = K_i$.

Structure 3: $A_1 = A_2 = \frac{K_p}{2}, B = K_d, C = 1, D = K_i$.

The optimal values of A, B, C, and D from equation 23 are used to initialize the other performance criteria. Equation 24, as shown at the bottom of the next page, shows the performance criteria of the first structure. Similarly, different performance criteria are found for the second and third structures:

Table 3 shows the fitness values of the various objective functions for the LF1-PID design's first step. The optimal values of [A, B, C, D] for the first structure are [0.71, 0.655, 1.99, 2.15], [1.26, 3.31, 4.25, 1.63], [1.03, 0.42, 0.58, 1.41], [1.06, 0.45, 0.48, 1.11], and [0.71, 0.63, 2.02, 2.27] for IAE, ISE, ITSE, IT2SE, and MPC, respectively. The number of iterations required for IAE, ISE, ITSE, IT2SE, and MPC are 180, 123, 97, 106, and 151, respectively.

It should be noted that the second and third structures have the same transfer function in the first two steps, LF1-PID and LF1-FPID. From Table 3, the fitness values of the second and third structures are lower than that of the first structure for all performance criteria.

Table 4 presents the step response of each structure, including parameters such as rising time, settling time, percentage overshoot, peak time, and steady-state error of LF1-PID controllers. From the table, we can make the following observations regarding the performance of the different controllers:

- The ITSE controllers exhibit fast response times based on their low rising time and settling time values. However, they may not be suitable for applications where overshoot is critical due to their relatively high values in the percent overshoot column.
- The IT2SE and IT2SE controllers have relatively low values in both the rising time (Tr) and settling time (Ts) columns, indicating that they are fast and responsive. They also have relatively low values in the percent overshoot (OS) column, suggesting that they may be a good choice when both speed and overshoot are important.
- The IAE and MPC controllers have relatively low values in the rising time (Tr) and settling time (Ts) columns, indicating that they are generally fast and responsive. However, they have relatively high values in the percent overshoot (OS) column, which means that they may not be the best choice for applications where overshoot is a critical issue.
- The ISE controllers have relatively high values in the percent overshoot (OS) column. However, they have relatively low values in the rise time (Tr) and settling time (Ts) columns, indicating that they are generally fast and responsive.
- In terms of steady-state error (ess), the IAE and MPC of the second and third structures have the lowest values, indicating that they are better at maintaining a steady-state response that tracks the desired value.

The step response of the first structure using the stated criteria is displayed in Figure 7. Based on the graph, it can be observed that there is minimal difference between the responses of the IAE and MPC controllers.

2) STEP TWO, LF1-FPID CONTROLLER

In this particular step, the effect of fractional-order parameters is introduced into the system under consideration. This means that the controllers used to regulate the system have fractional-order parameters that affect their behavior and performance. The performance criteria for the LF1-PID controller in the first structure are presented in equation 25, as shown at the bottom of page 12. Similarly, the performance criteria for the IAE and MPC controllers can also be computed for the first, second, and third structures.

The optimal values of [A, B, C, D, λ, μ] for the first structure are [5.1, 0.97, 18.40, 16.43, 0.99,1.50], [1.9, 0.41, 9.16, 2.37, 1.23, 1.50], [21.31, 4.25, 20.18, 102.15, 0.959, 1.491], [0.64, 0.39, 0.28, 1.50, 1.011, 0.838], and [12.35, 1.46, 36.35, 16.92, 1.50, 1.346], for IAE, ISE, ITSE, IT2SE, and MPC, respectively. The number of iterations for IAE, ISE, ITSE, IT2SE, and MPC are 759, 713, 768, 291, and 653. The fitness value and step information of the various objective functions for the second step design, LF1-FPID, are presented in Tables 5 and 6, respectively.

Upon comparing Table 3 with Table 5, it can be observed that the fractional components have improved the performance of all structures by minimizing their fitness values. Notably, the three structures exhibit enhancements with different criteria. Specifically, for ISE, all structures show lower fitness values in step 2 compared to step 1. Similarly,

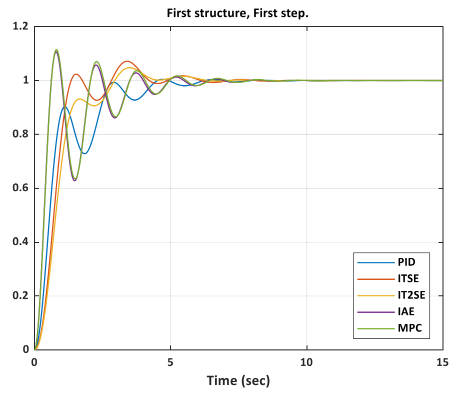

FIGURE 7. Step response of the first structure first step, LF1-PID controller.

step 2 has significantly lower ITSE values for all structures compared to step 1. For IT2SE, step 2 has a much lower fitness value for structure 2 and structure 3, while structure 1 shows similar fitness values for both steps. Moreover, step 2 has substantially lower IAE and MPC values for all structures compared to step 1.

In regards to the settling time (Ts), all structures have demonstrated a decrease in Ts from step 1 to step 2, indicating that the systems in step 2 respond faster. For instance, in Structure 1, Ts decreased from 7.74 seconds to 1.71 seconds for ISE and from 4.00 seconds to 0.15 seconds for ITSE. This trend holds for all other structures. Concerning overshoot (OS), most structures have shown a decline in OS from step 1 to step 2, suggesting that the systems in step 2 exhibit less overshoot. However, some structures have displayed an increase in OS. For example, in Structure 1, ISE has an OS of 62.61% in step 1 and 15.31% in step 2, while ITSE has an OS of 7.13% in step 1 and 46.35% in step 2. With respect to rise time (Tr), most structures have shown a decrease in Tr from step 1 to step 2. However, some structures have exhibited an increase in Tr. For example, in Structure 1, ISE has a Tr of 0.1044 seconds in step 1 and 0.1447 seconds in step 2, while ITSE has a Tr of 0.8287 seconds in step 1 and 0.0142 seconds in step 2. As for peak time, most structures have displayed a decrease in peak time from step 1 to step 2, suggesting that the systems in step 2 exhibit a faster response.

Figure 8 depicts the step response of the second structure in the second step of the optimization technique. The response performance of the MPC is observed to exhibit intermediate behavior between the ISE, ITSE, and IAE criteria.

The influence of the fractional-order on the step response of the first and second structures is demonstrated in Figure 9. The figure illustrates the step response for both structures in the first and second optimization steps.

Based on the figures and tables, it can be concluded that in step 2, which present the effects of fractional parameters, all structures demonstrated faster response times with lower settling times compared to step 1. Furthermore, most structures exhibited a reduction in overshoot, indicating improved performance in step 2, although some structures showed an increase in overshoot. While some structures demonstrated a decrease in rise time, the majority showed an increase. Most structures also showed a decrease in peak time, indicating faster response times in step 2.

TABLE 5. The second step's fitness values, second step (LF1-FPID).

Performance criteria	Structure 1 (FFPD-FPI)	Structure 2 (FFPID)	Structure 3 (FFPI-FFPD)
ISE	0.2041	0.2012	0.2012
ITSE	9.9335e-08	4.8513e-06	4.8513e-06
IT2SE	0.5180	0.0001	0.0001
IAE	0.2041	0.2282	0.2282
MPC	0.1618	0.1724	0.1724

TABLE 6. Step response information, second step (LF1-FPID).

Str.	PC	Tr	Ts	OS (%)	Peak time	ess
Structure 1	PID	0.8495	4.2798	0.4602	4.7092	3.574e-6
	ISE	0.1447	1.7185	15.3147	0.3200	1.441e-3
	ITSE	0.0142	0.1477	46.3546	0.0398	1.581e-6
	IT2SE	1.1240	4.4764	3.6626	3.9414	1.785e-4
	IAE	0.0445	0.2767	29.6724	0.1120	1.837e-5
	MPC	0.0279	0.3727	56.6632	0.0793	1.364e-6
Structure 2,3	ISE	0.1483	0.5404	13.4333	0.3262	7.055e-4
	ITSE	0.0459	0.2068	26.4981	0.1137	4.335e-6
	IT2SE	0.1088	0.5050	21.5182	0.2597	1.557e-6
	IAE	0.1545	0.6667	18.4560	0.3604	1.949e-5
	MPC	0.1545	0.6668	18.4508	0.3604	1.563e-5

3) STEP 3, NLF1-FPID CONTROLLER

In this stage, the impact of nonlinearity in the membership functions of the fuzzy inputs/outputs is taken into consideration. The values of the parameters of the fractional-order PID controller (A, B, C, D, λ, and μ) remain constant, while the nonlinearity factors (ζ_1, ζ_2, ζ_3, and so on) of each input/output membership function are optimized using the Modified Teaching-Learning-Based Optimization (MTLBO) algorithm. The minimum and maximum values of the nonlinearity factors are set at 0.1 and 7, respectively. The initial population is selected uniformly between 0.1 and 7, with a sample size of 10, 12, and 20 for the first, second, and third structures, respectively. An additional sample representing linear fuzzy is added to the initial population.

The number of optimized parameters is the total number of fuzzy inputs and outputs. For example, the first structure, FFPD-FPI, has two inputs and one output. The three factors (ζ_1, ζ_2, ζ_3) in the first structure represent the nonlinearity of

$$j_n = 0.2 \times \sum_{t=0(0.2)}^{20sec} [1 - \sum_{k=1(odd)}^{10001} \frac{k^2 w_s^2 \left(9AD - 9BC - 27AC\right) + 99BC - 99AD + 81AC + 81BD}{k\left((kw_s)^2 + 1\right)\left((kw_s)^4 - 14\left(kw_s\right)^2 + 81\right)} \sin\left(kw_s t\right) t^n]^2 \tag{24}$$

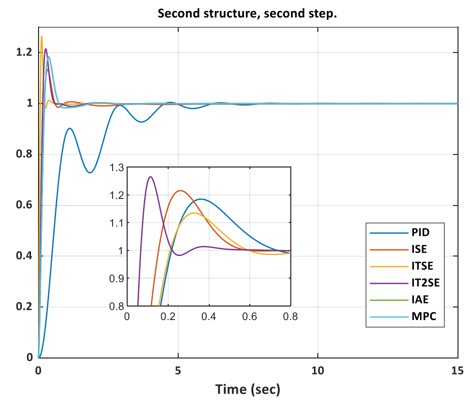

FIGURE 8. Step response of the second structure second step, LF1-FPID.

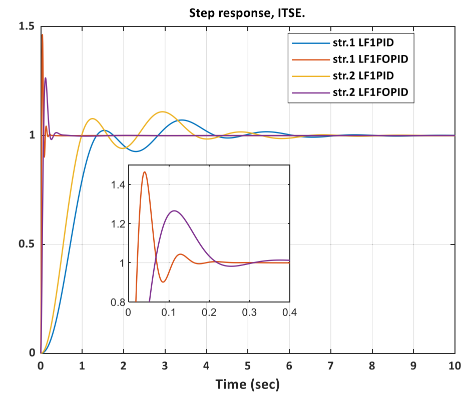

FIGURE 9. Step response of the first and second structures steps one and two, ITSE criteria.

TABLE 7. Fitness values, NLF1-FPID.

Performance criteria	Structure 1 (FFPD-FPI)	Structure 2 (FFPID)	Structure 3 (FFPI-FFPD)
ISE	0.1131	0.0162	0.0329
ITSE	2.9788e-06	6.6876e-06	6.9670e-05
IT2SE	0.5194	1.4098e-06	4.7357e-04
IAE	0.0220	0.0439	0.1051
MPC	0.0202	0.0226	0.0571

TABLE 8. Step response information, NLF1-FPID.

Str.	PC	Tr	Ts	OS (%)	Peak time	ess
	PID	0.8495	4.2798	0.4602	4.7092	3.574e-6
Structure 1	ISE	0.0276	3.0483	27.8268	0.0684	9.935e-4
	ITSE	0.0056	0.0910	64.5724	0.0171	1.672e-7
	IT2SE	1.0974	4.3629	3.3198	3.9277	1.326e-4
	IAE	0.0080	0.0814	47.8338	0.0229	4.124e-6
	MPC	0.0075	0.1677	72.7151	0.0232	6.658e-7
Structure 2	ISE	0.0204	0.1412	31.4108	0.0526	1.099e-4
	ITSE	0.0145	0.1329	36.6215	0.0385	9.828e-7
	IT2SE	0.0319	0.2008	28.6716	0.0794	3.253e-7
	IAE	0.0291	0.1845	29.6927	0.0733	2.433e-6
	MPC	0.0233	0.1561	30.8488	0.0591	2.019e-6
Structure 3	ISE	0.0538	0.9879	21.2973	0.1261	5.365e-4
	ITSE	0.0356	0.1667	29.6510	0.0896	3.424e-6
	IT2SE	0.1088	0.5050	21.5182	0.2597	1.557e-6
	IAE	0.0736	0.7982	19.4116	0.1696	1.731e-5
	MPC	0.0742	0.5604	19.1563	0.1702	1.091e-5

the first input, the second input, and the output, respectively. The second structure has four optimized parameters; the first three parameters represent the nonlinearity factors in the three inputs, while the last parameter represents the nonlinearity in the output membership function. For the third structure, FFPI-FFPD, six optimized parameters are required. The first three parameters represent the nonlinearity in the first fuzzy (FFPI) input 1, input 2, and output membership functions. Similarly, the last three parameters represent the nonlinearity in the second Fuzzy membership functions.

In the optimization process, the nonlinearity factors (ζ_1, ζ_2, and ζ_3) of the input/output membership functions were optimized for the first structure using the MTLBO algorithm. The optimal values of these parameters were found to be [0.100, 6.313, 0.359], [3.9687,1.1636,0.1001], [0.9179, 1.105, 0.9749], [1.343, 5.1266, 0.1010], and [0.1506, 3.536, 0.1021] for ISE, ITSE, IT2SE, IAE, and MPC, respectively. These values were determined based on the fitness value of the objective functions, which is shown in Table 7. The fitness value indicates the performance of the controller in terms of settling time, overshoot, and rise time, represents better performance.

Table 8 shows the step response of each structure for the third step, NLF1-FPID. The values of settling time, overshoot, and rise time were also evaluated and compared to the values obtained in the previous steps. It was found that the optimized values of the nonlinearity factors led to an improvement in the performance of the controllers, as evidenced by the lower settling time, overshoot, and rise time values.

Figure 10 depicts the step response of the second structure using different criteria. The MPC criteria response combines the advantages of ISE for lower overshoot and the advantages of ITSE for lower steady-state error. The impact of the nonlinearity of the membership functions on the step response of the first, second, and third structures is illustrated in Figure 11. The figure shows the step response for all structures in the second and third steps, LF1-FPID and NLF1-FPID. From the

$$
\begin{aligned}
j_n &= 0.2 \times \sum_{t=0(0.2)}^{20sec} \left[\left(1 - \sum_{k=1(odd)}^{10001} \frac{M\left(M - 3k^3 w_s^3 + 9\right) + N(N + 11kw_s - k^3 w_s^3)}{k\left(\left(M - 3k^2 w_s^2 + 9\right)^2 + \left(N + 11kw_s - k^3 w_s^3\right)^2\right)} \sin\left(kw_s t\right) \right) t^n \right]^2 \\
M &= 9AC + 9BCk^{\mu} w^{\mu} \cos\left(\frac{\pi}{2}\mu\right) + 9ADk^{-\lambda} w^{-\lambda} \cos\left(\frac{\pi}{2}\lambda\right) + 9BDk^{\mu-\lambda} w^{\mu-\lambda} \cos\left(\frac{\pi}{2}(\mu - \lambda)\right) \\
N &= 9BCk^{\mu} w^{\mu} \sin\left(\frac{\pi}{2}\mu\right) + 9ADk^{-\lambda} w^{-\lambda} \sin\left(\frac{\pi}{2}\lambda\right) + 9BDk^{\mu-\lambda} w^{\mu-\lambda} \sin\left(\frac{\pi}{2}(\mu - \lambda)\right)
\end{aligned} \tag{25}
$$

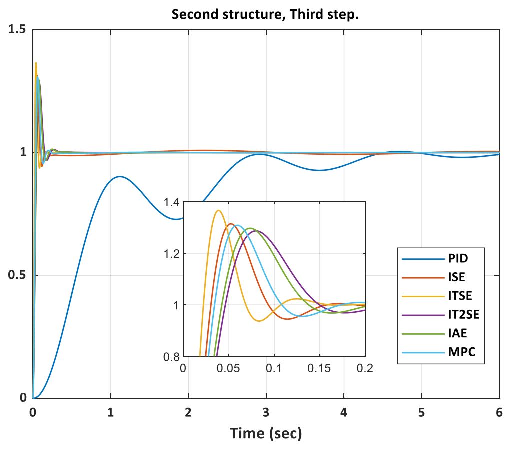

FIGURE 10. Step response of the second structure third step, NLF1-FPID controller.

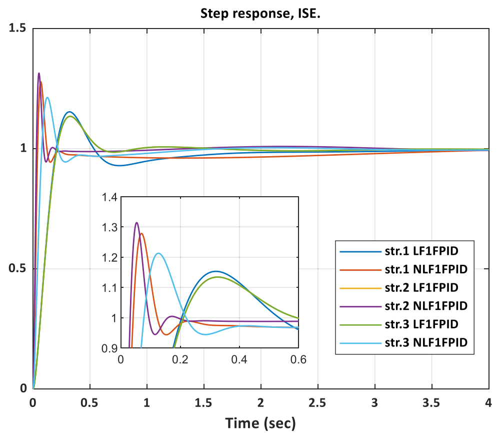

FIGURE 11. Step response of all structures second and third steps, ISE criteria.

figure, it is evident that optimizing the nonlinearity factors enhances the step response of the controller in all three structures. The fitness function used for this response is ISE, and similar observations can be made for other criteria.

Upon comparing Step 2 and Step 3, significant advancements in the performance metrics of the majority of controllers can be observed. Here is a comprehensive analysis:

- Rise Time (Tr): The controllers exhibit a decrease in rise time, which is desirable as it signifies the time taken by the system to reach its steady-state value. However, IT2SE in the third structure displays a minor increase in rise time.
- Settling Time (Ts): The controllers exhibit an improvement in settling time, indicating that the system reaches its steady-state value faster. However, ISE in the first and third structures shows an increase in settling time.
- Percent Overshoot (OS): The percent overshoot has reduced for the majority of the controllers. Nevertheless, there is a considerable increase in OS for ITSE in structure 1.
- Peak Time: The majority of controllers exhibit improvements in peak time, indicating that the system reaches its maximum value faster. However, ISE in structure 2 shows a slight increase in peak time.
- Steady-state error (ess): All controllers exhibit a decrease in steady-state error.

Overall, it is apparent that Step 3 has resulted in significant improvements in the performance metrics of the majority of controllers, demonstrating an increase in the efficiency and effectiveness of the control system.

4) STEP OUR, NLF2-FPID CONTROLLER

In this step, the uncertainty range is taken into account without modifying the values of $A, B, C, D, \zeta_1, \zeta_2 \ldots$ The lower membership function ratios $\alpha_1, \alpha_2, \alpha_3, ..,$ and so forth are fine-tuned using the same method as in the previous step. The output membership functions remain unchanged.

For the FFPD-FPI structure, two lower membership ratios, namely α_1 and α_2, are determined for the two inputs. The second structure requires the optimization of three parameters, and the third structure requires the optimization of four parameters. Tables 9 and 10 present the fitness value and step response of different objective functions for the NLF2-FPID structure in the last step.

Upon comparing the fitness values in Step 3 and Step 4, noticeable differences can be observed in the performance criteria of the three control strategies. The most significant improvement is seen in the ISE criteria for Structure 1 and Structure 3. In Step 4, the ISE values decreased from 0.1131 and 0.0329 to 0.0230 and 0.0309, respectively. This improvement indicates a more effective control mechanism in reducing the deviation between the target set point and the actual output. However, the ITSE and IAE criteria show only a slight improvement for Structure 2, with ITSE values remaining the same for all strategies in both steps. Similarly, the IT2SE criteria demonstrate notable improvement for Structure 3 in Step 4, with a decrease from 0.000473 to 0.000123. However, this progress is not evident for Structure 1 and Structure 2. The MPC criteria show no significant improvement between the two steps for all strategies, indicating limited enhancement in maximum peak control performance.

Figure 12 depicts the step response of the first structure using various performance criteria. It is evident from the figure that the steady-state value can be reached within 0.2 seconds using the MPC criterion. Notably, the IAE criterion delivers the best step performance, as shown in Figure 12. Table 10 further confirms that the IAE criterion achieves the minimum settling time of 0.0814 seconds.

Figure 13 demonstrates the effect of nonlinearity in the membership functions on the step response of the first, second, and third structures (NLF1-FPID and NLF2-FPID). The figure highlights that optimizing the uncertainty footprint of the type-2 fuzzy results in a slight reduction in overshoot across the three structures. Moreover, the second structure

TABLE 9. Fitness values, NLF2PID.

PC	Structure 1 (FFPD-FPI)	Structure 2 (FFPID)	Structure 3 (FFPI-FFPD)
ISE	0.0230	0.0162	0.0309
ITSE	2.9788e-06	6.6876e-06	6.9670e-05
IT2SE	0.5194	1.1206e-06	1.2300e-04
IAE	0.0220	0.0428	0.1051
MPC	0.0202	0.0226	0.0571

TABLE 10. Step response information, NLF2-FPID.

Str.	PC	Tr	Ts	OS (%)	Peak time	ess
	PID	0.8495	4.2798	0.4602	4.7092	3.574e-6
Structure 1	ISE	0.0276	3.0483	27.823	0.0684	9.935e-4
	ITSE	0.0056	0.0910	64.568	0.0171	1.6722-7
	IT2SE	1.0974	4.3629	3.3181	3.9277	1.325-e4
	IAE	0.0080	0.0814	47.8321	0.0229	4.124e-6
	MPC	0.0075	0.1677	72.712	0.0232	6.658e-7
Structure 2	ISE	0.0204	0.1412	31.409	0.0526	1.099e-4
	ITSE	0.0145	0.1329	36.6202	0.0385	9.828e-7
	IT2SE	0.0319	0.2013	28.6401	0.0794	3.239e-7
	IAE	0.0291	0.1869	29.5208	0.0733	2.412e-6
	MPC	0.0233	0.1561	30.8424	0.0591	2.019e-6
Structure 3	ISE	0.0496	1.1274	22.1954	0.1173	5.399e-5
	ITSE	0.0356	0.1667	29.6499	0.0896	3.424e-6
	IT2SE	0.1092	0.4645	19.3042	0.2546	1.428e-6
	IAE	0.0736	0.7982	19.4109	0.1696	1.731e-5
	MPC	0.0742	0.5604	19.1511	0.1702	1.091e-5

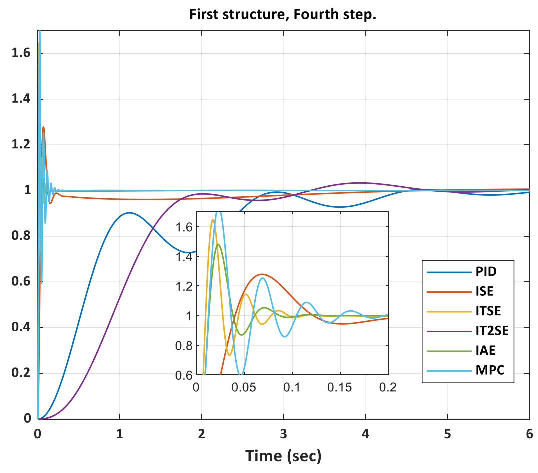

FIGURE 12. Step response of the first structure fourth step, NLF2-FPID.

performs better than the first structure but not as well as the third structure in terms of overshoot.

The advantages of type-2 fuzzy in the optimized nonlinear fuzzy controller are limited. Further comparison is made using the linear fuzzy controller. Figure 14 illustrates the impact of the uncertainty footprint in the linear type-1 fuzzy fractional-order controller (LF1-FPID) and linear type-2 fuzzy fractional-order controller (LF2-FPID) for the three structures. Optimizing the footprint of the uncertainty of the

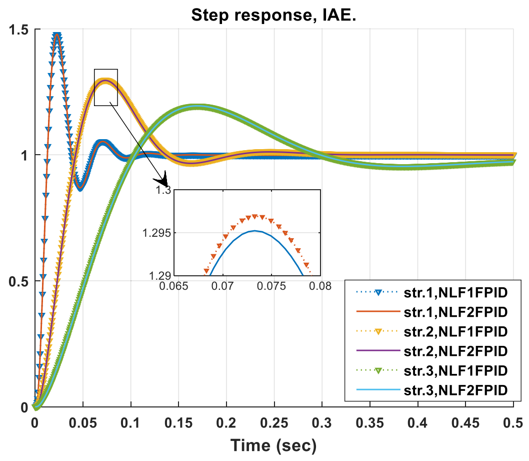

FIGURE 13. Step response of all structures third and fourth steps, IAE criteria.

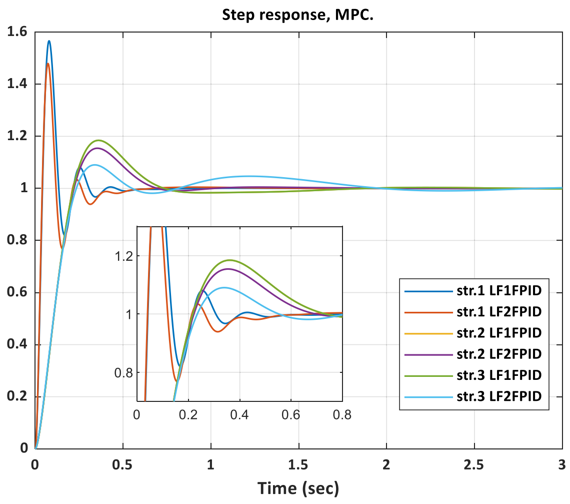

FIGURE 14. Step response of all structures, LF1-FPID, LF2-FPIF, for MPC criteria.

linear fuzzy controllers enhances the step response of all three structures. It is observed that the influence of the type-2 fuzzy footprint is more significant in the linear fuzzy controller compared to the nonlinear fuzzy controller, as depicted in Figures 13 and 14.

In conclusion, it can be stated that achieving good performance is possible using either a nonlinear type-1 fuzzy or a linear type-2 fuzzy controller. There is no need to optimize the uncertainty footprint of the optimized nonlinear fuzzy controller or the nonlinearity of the optimized linear type-2 fuzzy controllers. This observation suggests that the enhancement provided by the nonlinearity in the type-1 fuzzy controller is comparable to the improvement achieved by the linear type-2 fuzzy controller.

B. EXAMPLE 2: FRACTIONAL-ORDER SYSTEM

A fractional-order plant refers to a system or process whose dynamics are described by fractional calculus equations. The dynamics of fractional-order plants can be modeled using fractional-order differential equations, which are similar to ordinary differential equations but involve

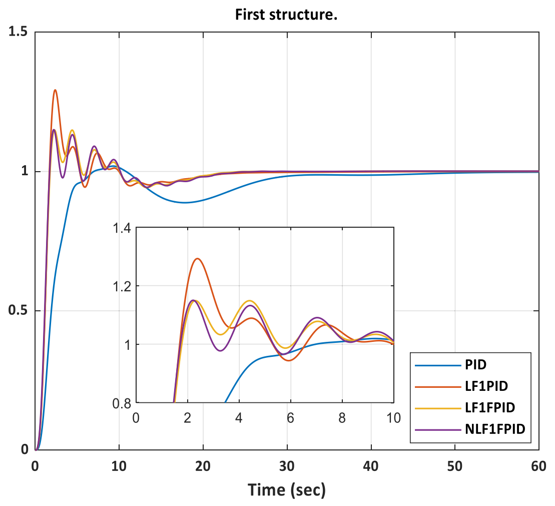

FIGURE 15. Step response of the first structure fractional-order plant, IAE.

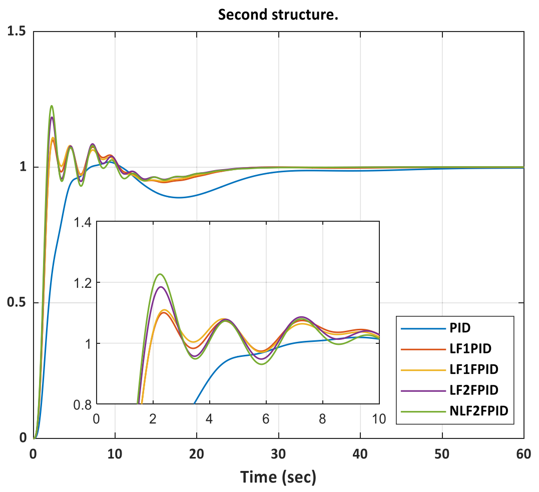

FIGURE 16. Step response of the second structure fractional-order plant, IAE.

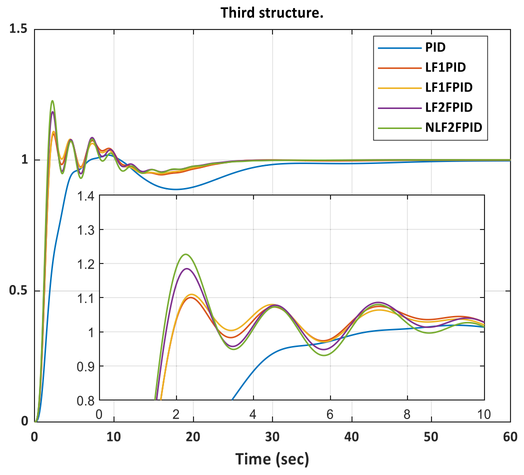

FIGURE 17. Step response of the third structure fractional-order plant, IAE.

fractional-order derivatives. These equations provide a more accurate description of the behavior of many real-world systems that exhibit non-integer order dynamics. Fractional-order plants have a wide range of applications, including chemical and biological processes, electrical circuits, and mechanical systems. Controlling such plants presents new challenges and opportunities, as traditional control techniques designed for integer-order systems may not be effective. Research in fractional calculus and fractional-order systems is an active area, with new theories and applications constantly emerging. The following example represents a fractional-order system [27]:

$$G(s) = \frac{0.4s^{1.12} + 0.3}{s^{5.25} + 4s^{3.77} + 6s^{2.92} + 4s^{1.68} + 1.3s^{1.43} + 0.3} \tag{26}$$

The four-step design procedure is applied to this plant. Figures 15 to 17 display the step response using the integral absolute error (IAE) objective function.

Figure 15 likely depicts a plot showcasing the step response of the first structure of the fractional-order plant for the initial three stages of the control system design. The plot indicates that the last step, denoted as NLF2-FPID, achieves outcomes that are considerably close to the third step of the control system design. This implies that optimizing the footprint of the uncertainty for the nonlinear fuzzy controller of the first structure of the fractional-order plant may not be necessary. Moreover, the plot reveals that the step response of the plant is augmented with each step of the design. For instance, the overshoot is reduced by the second step, attributed to advancements in the fractional-order effect. The integral absolute error (IAE) is also reduced by the third step, credited to advancements in the nonlinear effect. In summary, the plot and statement demonstrate how a systematic approach to control system design can enhance the performance of a fractional-order plant. Employing diverse control techniques and assessing system performance using the IAE criterion can help identify and address performance issues in the system.

In the second and third structures, as demonstrated in Figures 16-17, the optimization process involves the uncertainty footprint for the linear fuzzy controller followed by the optimization of the nonlinearity in the type-2 fuzzy. With each step, the integral absolute error is reduced, but there is an increase in overshoot. Different objective functions yield distinct step responses for this plant. These figures (Figures 15-17) exemplify how the proposed design algorithm can facilitate advancements in control system design for fractional-order plants.

V. ALGORITHM TESTING AND VALIDATION

A. PERFORMANCE RESULT VALIDATION

This section is dedicated to validating the effectiveness of the proposed algorithm through its application to different types of processes. The processes considered for validation include time delay processes, higher order processes, and fractional-order systems. The primary objective is to assess the performance of the algorithm by comparing it with other recently published algorithms, as presented in Table 11.

TABLE 11. Systems parameters and performance of the proposed algorithm and some selected from the previous research.

Fun.	Algorithm	Year	System parameters	System performance			
				Rise time	O.shot	Setl. time	Perf. Crit.
G1	MIGO [33]	2008	Kp=1.54, Ki=5.03, Kd=0.7, λ =1 , μ=1	***	13	15.7	***
	Gude [34]	2010	Kp=0.66, Ki=6.63, Kd=0, λ =1. , μ=1	***	28	51	***
	Tepljakov [39]	2012	Kp=1.93, Ki=0.87, Kd=0.401, λ =0.8 , μ= 0.69	***	12	11.5	***
	KC [41]	2018	Kp=1.09, Ki=13.7, Kd=0, λ =1.15 , μ=1	***	9	41	***
	ZN-FOC [40]	2021	Kp=2.80, Ki=2.37, Kd=0.899, λ =0.7, μ=0.8	***	27	7.2	***
	Proposed	***	FOPID =[0.16,0.09,23.49,2.13,1.01,1] ζ=[1.14,1,0.87]	0.83	18	9.3	ISE=1.6
			FOPID=[0.27,0.12,14.7,1.2,1.,1.05] ζ=[1.04,0.98,0.97]	1.03	7.218	1.07	ITSE=1.3
			FOPID=[0.25,0.1,15.3,1.2,1,1.05] ζ=[1.02,0.98,1.01]	1.17	3.62	5.17	IT2SE= 3.4
			FOPID=[1.5,0.3,1.68,0.13,1.01,0.59] ζ=[1.07,1.05,0.8]	1.22	12.64	1.12	IAE= 2.17
			FOPID=[0.14,0.06,27.7,2.17,1,μ=1.05] ζ=[0.99,1,0.95]	1.01	7.23	5.19	MPC= 1.7
G2	MIGO [33]	2008	Kp=0.91, Ki=1.79, Kd=0, λ =0.9 , μ=1	***	31	33.5	***
	Gude [34]	2010	Kp=0.61, Ki=3.55, Kd=0, λ =1.13 , μ=1	***	6.5	34.3	***
	KC [41]	2018	Kp=0.82, Ki=4.57, Kd=0, λ =1.18 , μ=1	***	6.5	42.8	***
	ZN-FOC [40]	2021	Kp=2, Ki=3.08, Kd=0.59, λ =0.9 , μ= 0.9	***	27	23.3	***
	Proposed	***	FOPID =[0.93,2,2.4,0.65,1.2,1.2], ζ=[1.55,0.65,0.86]	1.07	26.2	12.5	ISE =1.82
			FOPID =[2.9,4.1,1.85,0.48,1,1.5] ζ=[0.94,0.92,0.98]	0.85	11.5	5.54	ITSE= 0.26
			FOPID =[1.7,2.5,1.35,0.47,0.99,1.33] ζ=[0.99,1,0.99]	1.5	4.3	3.6	IT2SE= 2.3
			FOPID =[1.05,1.47,3.23,0.92,1,1.47] ζ=[0.97,1.3,0.74]	0.79	14.9	5.4	IAE= 1.08
			FOPID =[1.41,2.18,4.2,1.04,1,1.5] ζ=[1.02,1.02,0.98]	0.71	16.9	5.02	MPC= 0.69
G3	Monje [11]	2006	***	***	13.5	5.3	***
	critical 1 [42]	2006	Kp=1.03, Ki=1.06, Kd=0.81, λ =1.08 , μ=0.79	***	48.5	30.7	***
	KC [41]	2018	Kp=0.38, Ki=28, Kd=0, λ =0.71 , μ=1	***	13	42	***
	ZN-FOC [40]	2021	Kp=5.9, Ki=2.7, Kd=1.5, λ =0.4 , μ=0.4	***	40	2.3	***
	Proposed	***	FOPID =[1.5,0.7,7.44,0.01,1.01,0.99] ζ=[5.03,0.83,0.1]	0.113	6.44	0.93	ISE =0.13
			FOPID =[2.8,0.64,12.8,3.4,0.99,1.3] ζ=[1.22,1.13,0.56]	0.38	0.57	0.79	ITSE= 0.01
			FOPID =[1.44,1.41,3.51,10.7,1,1] ζ=[1.05,0.9,1.05]	1.7	0.19	3.4	IT2SE= 0.7
			FOPID =[5.3,1.1,6.8,4,0.81,1.22] ζ=[1.42,0.88,0.48]	0.135	2.95	0.67	IAE= 0.23
			FOPID =[0.79,0.44,13.6,3.36,1,1.0] ζ=[3.72,0.37,0.1]	0.14	1.97	0.55	MPC= 0.15
G4	Shaped 1[42]	2006	Kp=0.199, Ki=0.46, Kd=0.08, λ =1.48 , μ=0.98	***	37.8	18.5	***
	Shaped 2[42]	2006	Kp=1.33, Ki=1.77, Kd=-0.41, λ =1.32 , μ= -0.18	***	14.5	10.1	***
	critical 1 [42]	2006	Kp=0.24, Ki=0.56, Kd=-0.13, λ =1.46 , μ=1.003	***	32	17	***
	MIGO [33]	2008	Kp=0.44, Ki=0.3, Kd=0, λ =1.1 , μ=1	***	8	4	***
	Gude [34]	2010	Kp=0.32, Ki=0.29, Kd=0, λ =1.12 , μ=1	***	8.2	5.8	***
	KC [41]	2018	Kp=0.66, Ki=0.35, Kd=0, λ =1.18 , μ=1	***	7.5	6.7	***
	ZN-FOC [40]	2021	Kp=0.76, Ki=0.64, Kd=0.122, λ =0.9 , μ=0.9	***	18	5.5	***
	Proposed	***	FOPID =[0.59,0.15,0.6,3.42,1.1,1.02] ζ=[0.92,1.2,0.92]	0.19	32.8	2.94	ISE =0.43
			FOPID =[0.25,0.04,1.54,8.22,1,1.06] ζ=[1.02,1.08,0.98]	0.25	19.3	1.51	ITSE= 0.03
			FOPID =[15.2,0.37,0.03,0.1,1,1.31] ζ=[1.02,1.02,0.99]	0.49	3.62	1.45	IT2SE= 0.02
			FOPID =[0.78,0.02,0.58,1.9,1.,1.34] ζ=[0.47,2.74,0.56]	0.002	16.0	1.61	IAE= 0.57
			FOPID =[1.93,0.01,0.33,0.84,1,1.5] ζ=[0.96,1.06,1.01]	0.41	7.8	2.23	MPC= 0.5
G5	Set-Point Reg. [37]	2021	Kp=0.381, Ki=5.199, Kd=5.578, λ =1.06 , μ=0.53	0.173	3.724	***	***
	Reg. +20% [37]			0.2187	5.247	***	***
	Reg. -20% [37]			0.3647	0.6526	***	***
	Dist. rejection [37]			0.2205	2.9863	***	***
	Random delay [37]			0.1817	2.9576	***	***
	Proposed	***	FOPID =[0.13,0.1,32.6,30.9,1.05,1] ζ=[1.1,1.04,0.91]	0.07	16.8	3.1	ISE =0.13
			FOPID =[9.7,3.3,0.69,0.79,1.03,1] ζ=[1.13,1.04,0.9]	0.07	18.9	2.13	ITSE= 0.007
			FOPID =[1.11,0.4,5.13,12.1,1,0.99] ζ=[0.97,0.97,1.1]	0.1	22.2	2.33	IT2SE=0.063
			FOPID =[1.3,1.1,3.2,5.81,1.13,0.88] ζ=[0.81,1.3,1.04]	0.08	19.6	1.34	IAE= 0.27
			FOPID=[1.13,0.78,2.62,4.14,1.01,1.03]ζ=[1.03,1.2,0.85]	0.07	19	2.97	MPC=0.16
	Proposed*		FOPID =[1.27,1.1,1.34,1.561,1] ζ=[1.1,1.2,0.71]	0.09	0.2	4.6	ISE =0.14
			FOPID =[0.65,0.47,2.54,3.06,1.02,1.13] ζ=[1.1,1.2,0.74]	0.085	0.58	4.23	ITSE= 0.03
			FOPID =[1.7,0.9,1.34,2.5,1,1] ζ=[1.1,1.1,0.83]	0.27	13.7	3.21	IT2SE=0.27
			FOPID =[0.74,0.58,2.43,3.08,1.02,1.01] ζ=[1.11,1.2,0.7]	0.085	2.4	3.69	IAE= 0.298
			FOPID =[0.788,0.62,2.2,2.68,1.02,1.04] ζ=[1.1,1.21,0.7]	0.086	1.3	4	MPC=0.19

The transfer functions provided below exemplify the diverse nature of plants in various applications:

a) First order Pulse dead time lag-dominant process [33]:

$$G_1(s) = \frac{2.4351}{12.5688s + 1} e^{-1.0787s} \tag{27}$$

b) Higher order Processor [34]:

$$G_2(s) = \frac{1}{(s+1)^4} \tag{28}$$

c) Integrating Time-Delay Process [35]:

$$G_3(s) = \frac{0.55}{s(0.6s+1)} e^{-0.05s} \tag{29}$$

d) First order pulse dead time delay-dominant Process [32]:

$$G_4(s) = \frac{1}{0.2s+1}e^{-0.4s} \tag{30}$$

e) Modelling of Pneumatic Pressure System [37]:

$$G_5(s) = \frac{1}{1.1s^{1.5}+1}e^{-0.105s} \tag{31}$$

Through the comparison with existing algorithms in Table 11, the performance of the proposed algorithm will be assessed in terms of its ability to handle the unique characteristics and challenges posed by these different types of processes. This validation process aims to provide insights into the algorithm's effectiveness and its potential for practical applications within various domains.

The first four processes have been previously tested in [32], and the performance results (specifically, overshoot value and settling time) available in the literature are included in the table for comparison with the proposed algorithm. Additionally, the results of the fractional-order process (G5) are presented in [37].

The system parameters of the proposed algorithm are presented in two vectors: FOPID and ζ. The FOPID vector represents the values of A, B, C, D, λ, and μ in the specified sequence, while the ζ vector represents the nonlinearity factors in the two input membership functions and the output membership function. For the purpose of this comparison, only the first structure (FFPD-FPI) is considered, as it is the most commonly used in the literature. Additionally, the fourth step of the proposed algorithm, which involves optimizing the lower membership function, is not taken into account. This is due to its minimal impact on the designed nonlinear controller, as previously discussed.

The table serves to highlight the robustness of the proposed algorithm. Notably, the ITSE, IT2SE, and MPC criteria demonstrate superior performance compared to the best algorithm presented in the table for the first process (G1), considering both overshoot and settling time. Similarly, for the second process (G2), the IT2SE criterion outperforms other algorithms in terms of overshoot and settling time. In the case of the integrating time delay process (G3), all criteria exhibit improved performance compared to the best algorithm presented. For the first order pulse dead time delay-dominant process (G4), the IT2SE criterion proves to be the most effective controller according to the table's results.

Furthermore, the proposed algorithm is shown to be well-suited for fractional-order processes (G5). The table demonstrates that the rise time achieved by the proposed algorithm, irrespective of the performance criterion used, is superior to the best rise time reported in [37]. To facilitate a fair comparison of overshoot values, the fitness function in the proposed algorithm incorporates the maximum value of the step response. The corresponding results are denoted as Proposed*. These results exhibit slightly longer rise times but with reduced overshoot, surpassing the performance of the approach presented in [37] in terms of both rise time and overshoot.

TABLE 12. Computational time.

Algorithm step	Transfer function	Computational time		
		Max	Min	Ava
Step 1	Integer order	34.64	22.8	25.7
	Fractional-order	44.77	32.5	36.7
Step 2	Integer order	42.8	33.4	38.1
	Fractional-order	45.3	35.7	39.8
Step 3	Integer order	265.3	225.5	245.8
	Fractional-order	271.2	250.9	261.9
Step 4	Integer order	268.6	189.9	215.3
	Fractional-order	278.6	236.7	243.8

B. COMPUTATIONAL PROCESS

The simulation and implementation of this study were conducted using MATLAB 2020b. The MATLAB codes and Simulink models employed in the simulations are provided in the appendix for reference. To measure the computational time required for the execution of the proposed algorithm, the (tic, toc) functions in MATLAB were utilized. The resulting computational times, obtained from an average of 10 trials for the first structure, are presented in the following table.

The optimization process for the first two steps in Table 12 was terminated when the optimal fitness value was achieved. However, the third and fourth steps were stopped due to reaching the maximum number of iterations set for the Modified Teaching-Learning-Based Optimization (MTLBO) algorithm, which was set to 25 in this study. The computational limitations experienced in the third and fourth steps can be attributed to the implementation of nonlinear fuzzy logic and type-2 fuzzy logic, which were simulated using MATLAB Simulink. These simulation constraints could be overcome by representing the nonlinear fuzzy controller and type-2 fuzzy controler in the form of transfer functions, similar to the approach used for type-1 linear fuzzy logic [15]. Furthermore, the computational time required for the second and third structures was longer due to the larger population size used in these steps.

On a contrasting note, the Modified Teaching-Learning-Based Optimization (MTLBO) algorithm introduced a challenge regarding computational time. However, the outcomes achieved after 25 iterations using MTLBO were found to be comparable to those obtained after approximately 50 iterations using the Traditional Teaching-Learning-Based Optimization (TLBO) algorithm. Interestingly, the time taken to complete 25 iterations with MTLBO was approximately 75% of the time required for 50 iterations with TLBO. In simpler terms, the same level of optimization results could be attained with a reduction of 25% in computational time.

The selection of the population size for the Modified Teaching-Learning-Based Optimization (MTLBO) algorithm was determined based on careful consideration. Specifically, population size is a crucial factor that depends on the number of variables and the range of parameters involved in the optimization process. In this study, the population size for the three structures was chosen as 10, 12, and 15, respectively, taking into account the specific characteristics of each structure. It is important to note that the range of nonlinearity factors in this study varied from 0.1 to 7, which represents a wide range of values. It was observed that larger population sizes resulted in longer computational time for each iteration, while smaller population sizes required a greater number of iterations to converge to a solution. Therefore, the selection of an appropriate population size is a trade-off between computational efficiency and convergence speed in the optimization process.

C. STABILITY ANALYSIS

Stability is a fundamental requirement for control systems, ensuring that internal signals remain bounded and preventing loss of control or equipment damage. In linear feedback systems, stability is typically assessed by analyzing the poles of the closed-loop transfer function. However, it is also important to consider the robustness of stability, which refers to the system's ability to tolerate changes in the loop gain without losing stability.

The root locus plot is a useful tool for estimating the range of loop gain values that maintain stability. Robust stability goes beyond loop gain changes and takes into account imperfect plant modeling, where both gain and phase may not be known precisely. Phase variation near the gain crossover frequency, where the open-loop gain is 0dB, is particularly critical. The phase margin quantifies the amount of phase variation needed at the gain crossover frequency to destabilize the system, while the gain margin measures the relative gain variation required for instability. These stability margins provide an estimate of the safety margin for closed-loop stability, with smaller margins indicating greater fragility.

To satisfy the criteria for a feedback control system, several conditions must be met. First, closed-loop stability with an appropriate margin should be ensured. Second, the system should exhibit robustness against process variations and model uncertainty. Additionally, it should be insensitive to time delays and measurement noise. Lastly, the system should demonstrate good step performance, characterized by fast set-point tracing and minimal overshoot.

In the proposed algorithm, these criteria are incorporated into the objective function using three constraints derived from frequency domain analysis: gain margin ($Gm > 2$), phase margin ($Pm > 30$), and phase crossover frequency ($5 < Wcg < 20$). The gain margin represents the amount of gain variation required to achieve unity loop gain at the frequency where the phase angle is $-180°$. The phase margin is the difference between the response phase and $-180°$ when the loop gain is 1.0. The gain crossover frequency is the frequency at which the magnitude is 1.0.

Calculating these parameters involves analyzing the open-loop transfer function using the formula $((jw)^{\alpha} = w^{\alpha}(\cos\alpha + j\sin\alpha))$. The specified criteria ensure acceptable stability margins, sensitivity within the operating frequency range, and desirable performance.

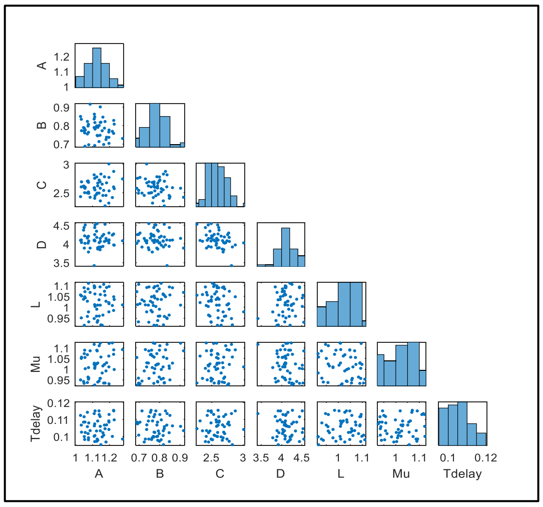

FIGURE 18. Scatter plot of the controller parameters.

In this section, the sensitivity of the system is tested using a fractional-order process described by equation (32) to achieve the worst case. An (MPC) criteria is employed, with controller values specified as FOPID = [1.13, 0.78, 2.62, 4.14, 1.01, 1.03]. These values serve as the centers of the Normal Distribution function, and normalizing coefficients (σ) are set as [0.065, 0.045, 0.151, 0.24] respectively. The fractional-orders (λ and μ) and time delay are uniformly distributed from [0.91, 0.92, 0.094] to [1.12, 1.13, 0.11] respectively, resulting in the generation of 50 random samples. The corresponding values are depicted in Figure18.

The evaluation analysis involves determining the boundaries of the step response based on specific performance criteria. In this study, the selected criteria for the step response are a maximum overshoot of 20%, a rising time of 1 second, and a settling time of 3 seconds, as illustrated in Figure 19.

These criteria provide quantitative measures for assessing the performance of the system's step response. The maximum overshoot refers to the peak value beyond the desired setpoint, which should be limited to 20% to ensure stability and avoid excessive oscillations. The rising time represents the duration required for the system's output to reach and stabilize within a certain tolerance of the setpoint. In this case, a rising time of 1 second is specified to achieve a prompt response. The settling time indicates the duration needed for the system's output to reach and remain within a specified tolerance band around the setpoint, ensuring steady-state operation. In this study, a settling time of 3 seconds is considered as a suitable criterion for achieving stable behavior.

By setting these boundaries for the step response, the performance of the system can be assessed against the desired criteria, enabling a comprehensive evaluation of its dynamic behavior and responsiveness. Figure 19 provides a visual representation of the step response with the specified performance boundaries.

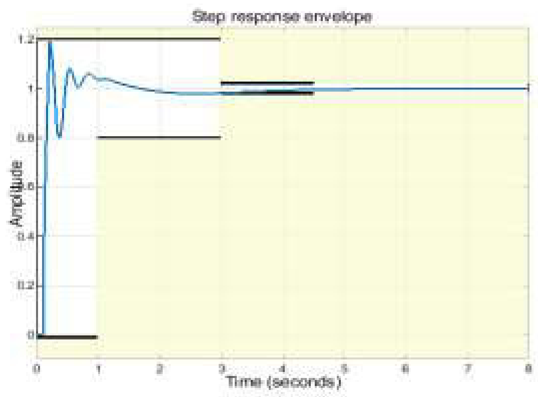

FIGURE 19. Step response envelope used for system evaluation.

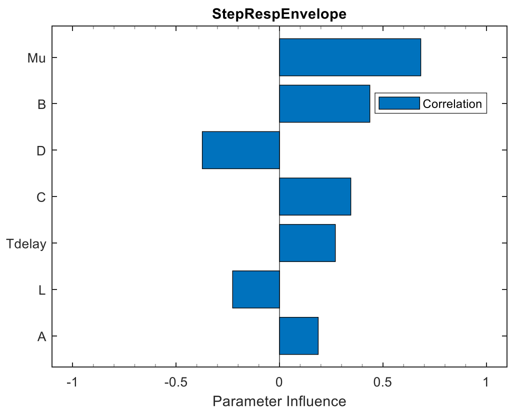

FIGURE 21. Correlation analysis.

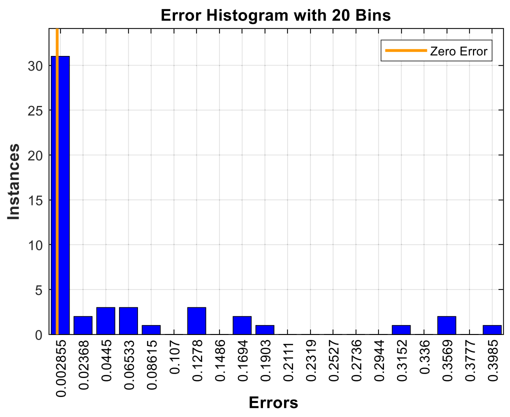

FIGURE 20. Error histogram on the step response envelope.

Figure 20 displays a histogram representing the distribution of errors, calculated based on the envelope of the response. The error is a measure of the deviation between the desired response and the actual output of the system. In this study, the errors for a sample set of fifty are analyzed and presented in the histogram.

Upon examining the histogram, it is observed that one sample out of the fifty has an error of approximately 0.4. Additionally, two samples exhibit errors around 0.36, while only four samples have errors exceeding 0.2 in their envelopes. This information provides insights into the variability and accuracy of the system's performance. By comparing the histogram in Figure 20 with Figure 19, it can be inferred that the proposed algorithm has contributed to a reduction in errors, resulting in improved response accuracy.

To investigate the correlation between the control parameters and the system's output, Figure 21 is presented. This figure depicts the relationship between the control parameters, which are the inputs to the system, and the corresponding output. By analyzing the correlation, it is possible to identify any patterns or dependencies between the control parameters and the resulting output. This information is crucial for understanding the influence of the control parameters on the system's behavior and performance.

Overall, Figures 20 and 21 provide valuable insights into the error distribution and the correlation between the control parameters and the system's output. These findings contribute to the evaluation and understanding of the proposed algorithm's effectiveness and its impact on the system's performance.

Figure 21 presents the correlation analysis of the system parameters, namely A, B, C, D, L, Mu, and Td. The correlation coefficients between these parameters and their influence on the system's output response are as follows: approximately 0.2 for parameter A, around 0.45 for parameter B, approximately 0.35 for parameter C, −0.48 for parameter D, −0.25 for parameter L, approximately 0.7 for parameter Mu, and 0.25 for parameter Td.

These correlation coefficients provide insights into the relationship between the system parameters and the output response. A positive correlation coefficient indicates a direct relationship, meaning that an increase in the parameter's value leads to an increase in the system's output response, while a negative correlation coefficient suggests an inverse relationship, where an increase in the parameter's value results in a decrease in the system's output response.

Based on the correlation analysis presented in Figure 21, it can be concluded that parameter Mu exhibits the highest positive correlation with the system's output response, with a correlation coefficient of approximately 0.7. This indicates that variations in the Mu parameter have a substantial influence on the behavior of the system, and increasing the Mu value is likely to result in an increase in the system's output response.

On the other hand, parameter D shows the highest negative correlation with the system's output response, as evidenced by a correlation coefficient of -0.48. This negative correlation implies that as the value of D increases, the system's output response tends to decrease, and vice versa.

Understanding the correlation between system parameters and the output response is crucial for system analysis and control. It allows for the identification of key parameters that significantly impact the system's behavior and provides valuable insights for system optimization and performance enhancement. The correlation coefficients presented in Figure 21 offer quantitative measures of these relationships, aiding in the development of effective control strategies and decision-making processes.

VI. CONCLUSION AND FUTURE WORK

This research paper introduces a novel four-step algorithm for the design of hybrid type-2 fuzzy fractional-order PID controllers. The proposed algorithm optimizes each component of the controller, including the nonlinearity and the footprint of uncertainty in fuzzy membership functions. Additionally, it addresses the fractional-order aspects of the controllers and processes using the Fourier Series Method, which offers greater accuracy compared to traditional fractional-order approximation methods.

Three different type-2 fuzzy FOPID control structures are presented, and various performance criteria such as ISE, ITSE, IT2SE, IAE, and MPC are employed as fitness functions. The proposed algorithm optimizes the controller parameters in a step-by-step manner. Firstly, the A, B, C, and D parameters, which correspond to Kp, Ki, and Kd in a PID controller, are optimized. Subsequently, the fractional-order parameters are considered in the second step. The third and fourth steps focus on handling the fuzzy parameters, specifically the nonlinearity in the membership functions and the footprint of uncertainty.

The proposed algorithm is applied to both integer order and fractional-order plants, and the influence of each parameter group is thoroughly examined. Furthermore, the algorithm's performance is validated by comparing the controller's response with various algorithms reported in the literature. Stability analysis and computational time analysis are also conducted for the algorithm.

The results demonstrate the robustness and superior performance of the proposed algorithm. However, it should be noted that the computational analysis highlights a limitation in the simulation process due to the utilization of MATLAB Simulink for representing the nonlinearity and footprint of uncertainty in the fuzzy controller. To address this limitation in future work, researchers are encouraged to develop a transfer function in the time or frequency domain for the fuzzy controller, even when operating in different modes. By doing so, the results obtained in the third and fourth steps of the algorithm can be further enhanced. Currently, the optimization algorithm terminates after 25 iterations in the modified TLBO because of this limitation.

The algorithm presented in this paper holds promise for application in various engineering fields, particularly in the context of smart grids and renewable energy integration problems. In the next phase of this project, the authors plan to apply this algorithm to address low-frequency oscillation challenges [43], [44] and implement the results in the Jordan Power system using the available Real-Time Digital Simulator (RTDS) at Mu'tah University.

APPENDIX A: MATLAB CODES

```
clear all
clc
%% Training steps 1+2
global G Gff str cost step Results
Gff = @(x) 9./(x+1)./(x.^2+2.*x+9);
Gf = @(x) 9/(x+1)/(x^2+2*x+9);
s = tf('s');
G = Gf(s);
for str=1:2
for step =1:2 % LF1PIF, LF1FOPID
for cost=1:5 % cost= ISE, ISTE,IST2E, IAE, FFF
Results.str{str}.step{step}.perfor{cost}=optimPID1
;
end
end
end
Results.str{3}=Results.str{2};
Results.PID=pidtune(G, 'PID');
save('ResultsP1', 'Results')
clc
disp('step 1+2 finished')
%% Training steps 3-5
clear all
clc
global str cost step Results
load ('ResultsP1');
num=9;
denum=[1 3 11 9];
s=fotf('s');
for cost=1:5
for step =3:5 % NLF1FPIF, NLF2FPID, LF2FPID
for str=1:3 % cost= ISE, ISTE,IST2E, IAE, FFF
Results.str{str}.step{step}.perfor{cost}=optimPID2
;
save('ResultsP1', 'Results')
end
disp(['step ' num2str(step) ' cost '
num2str(cost) ' are finished'])
end
disp(['step 3-5 str ' num2str(str) ' finished'])
end
%% information Structure 1
clear
num=9;
denum=[1 3 11 9];
load ('ResultsP1');
load_system('str2.slx');
X=[Results.PID.Kp,Results.PID.Kd,1,Results.PID.Ki,
1,1];
fis3=FUZZY3([1 1 1 1 1 1]);
out=sim('str2.slx');
Results.pid.y=out.simout.Data;
Results.pid.t=out.simout.Time;
Results.pid.stepinto=stepinfo(out.simout.Data,out.
simout.Time,1);
load_system('str1.slx');
str=1; step=1;
for cost=1:5
X=[Results.str{str}.step{step}.perfor{cost}.x(end,
:),1,1];
fis2=FUZZY2([1 1 1 1 1 ]);
out=sim('str1.slx');
Results.str{str}.step{step}.perfor{cost}.y=out.sim
out.Data;
Results.str{str}.step{step}.perfor{cost}.t=out.sim
out.Time;
Results.str{str}.step{step}.perfor{cost}.stepinto=
stepinfo(out.simout.Data,out.simout.Time,1);
end
save('ResultsP1', 'Results')
str=1; step=2;
for cost = 1:5
X = Results.str{str}.step{step}.perfor{cost}.x(end,:
);
fis2 = FUZZY2([1 1 1 1 1 ]);
out = sim('str1.slx');
```

```
Results.str{str}.step{step}.perfor{cost}.y = out.sim
out.Data;
Results.str{str}.step{step}.perfor{cost}.t = out.sim
out.Time;
Results.str{str}.step{step}.perfor{cost}.stepinto =
stepinfo(out.simout.Data,out.simout.Time,1);
end
save('ResultsP1', 'Results')
str = 1; step = 3;
for cost = 1:5
X = Results.str{str}.step{1,2}.perfor{cost}.x(end,:);
fis2 = FUZZY2([Results.str{str}.step{step}.perfor{co
st}.x(end,:), 1, 1 ]);
out = sim('str1.slx');
Results.str{str}.step{step}.perfor{cost}.y = out.sim
out.Data;
Results.str{str}.step{step}.perfor{cost}.t = out.sim
out.Time;
Results.str{str}.step{step}.perfor{cost}.stepinto =
stepinfo(out.simout.Data,out.simout.Time,1);
end
save('ResultsP1', 'Results')
str = 1; step = 4;
for cost = 1:5
X = Results.str{str}.step{1,2}.perfor{cost}.x(end,:)
;
fis2 = FUZZY2([Results.str{str}.step{1,3}.perfor{cos
t}.x(end,:),
Results.str{str}.step{step}.perfor{cost}.x(end,:)
]);
out = sim('str1.slx');
Results.str{str}.step{step}.perfor{cost}.y = out.sim
out.Data;
Results.str{str}.step{step}.perfor{cost}.t = out.sim
out.Time;
Results.str{str}.step{step}.perfor{cost}.stepinto =
stepinfo(out.simout.Data,out.simout.Time,1);
end
save('ResultsP1', 'Results')
str = 1; step = 5;
for cost = 1:5
X = Results.str{str}.step{1,2}.perfor{cost}.x(end,:)
;
fis2 = FUZZY2([1,1,1,
Results.str{str}.step{step}.perfor{cost}.x(end,:)
]);
out = sim('str1.slx');
Results.str{str}.step{step}.perfor{cost}.y = out.sim
out.Data;
Results.str{str}.step{step}.perfor{cost}.t = out.sim
out.Time;
Results.str{str}.step{step}.perfor{cost}.stepinto =
stepinfo(out.simout.Data,out.simout.Time,1);
end
save('ResultsP1', 'Results')
%% information Structure 2
load_system('str2.slx');
str = 2; step = 1;
for cost = 1:5
X = [Results.str{str}.step{step}.perfor{cost}.x(end,
:),1,1];
fis3 = FUZZY3([1 1 1 1 1 1 1]);
out = sim('str2.slx');
Results.str{str}.step{step}.perfor{cost}.y = out.sim
out.Data;
Results.str{str}.step{step}.perfor{cost}.t = out.sim
out.Time;
Results.str{str}.step{step}.perfor{cost}.stepinto =
stepinfo(out.simout.Data,out.simout.Time,1);
end
save('ResultsP1', 'Results')
str = 2; step = 2;
for cost = 1:5
X = Results.str{str}.step{step}.perfor{cost}.x(end,:
);
fis3 = FUZZY3([1 1 1 1 1 1]);
out = sim('str2.slx');
Results.str{str}.step{step}.perfor{cost}.y = out.sim
out.Data;
Results.str{str}.step{step}.perfor{cost}.t = out.sim
out.Time;
Results.str{str}.step{step}.perfor{cost}.stepinto =
stepinfo(out.simout.Data,out.simout.Time,1);
end
save('ResultsP1', 'Results')
str = 2; step = 3;
for cost = 1:5
X = Results.str{str}.step{1,2}.perfor{cost}.x(end,:)
;
fis3 = FUZZY3([Results.str{str}.step{step}.perfor{co
st}.x(end,:), 1, 1,1 ]);
out = sim('str2.slx');
Results.str{str}.step{step}.perfor{cost}.y = out.sim
out.Data;
Results.str{str}.step{step}.perfor{cost}.t = out.sim
out.Time;
Results.str{str}.step{step}.perfor{cost}.stepinto =
stepinfo(out.simout.Data,out.simout.Time,1);
end
save('ResultsP1','Results')
str = 2; step = 4;
for cost = 1:5
X = Results.str{str}.step{1,2}.perfor{cost}.x(end,:)
;
fis3 = FUZZY3([Results.str{str}.step{1,3}.perfor{cos
t}.x(end,:),
Results.str{str}.step{step}.perfor{cost}.x(end,:)
]);
out = sim('str2.slx');
Results.str{str}.step{step}.perfor{cost}.y = out.sim
out.Data;
Results.str{str}.step{step}.perfor{cost}.t = out.sim
out.Time;
Results.str{str}.step{step}.perfor{cost}.stepinto =
stepinfo(out.simout.Data,out.simout.Time,1);
end
save('ResultsP1','Results')
str = 2; step = 5;
for cost = 1:5
X = Results.str{str}.step{1,2}.perfor{cost}.x(end,:)
;
fis3 = FUZZY3([1,1,1,1,
Results.str{str}.step{step}.perfor{cost}.x(end,:)
]);
out = sim('str2.slx');
Results.str{str}.step{step}.perfor{cost}.y = out.sim
out.Data;
Results.str{str}.step{step}.perfor{cost}.t = out.sim
out.Time;
Results.str{str}.step{step}.perfor{cost}.stepinto =
stepinfo(out.simout.Data,out.simout.Time,1);
end
save('ResultsP1', 'Results')
%% information structure 3
load ('ResultsP1')
num = 9;
denum = [1 3 11 9];
load_system('str3.slx');
str = 3; step = 1;
for cost = 1:5
X = [Results.str{str}.step{step}.perfor{cost}.x(end
,:),1,1];
if
numel(Results.str{str}.step{step}.perfor{cost})>0
[fis1, fis2] = FUZZY4([1 1 1 1 1 1 1 1 1]);
out = sim('str3.slx');
Results.str{str}.step{step}.perfor{cost}.y = out.sim
out.Data;
Results.str{str}.step{step}.perfor{cost}.t = out.sim
out.Time;
Results.str{str}.step{step}.perfor{cost}.stepinto =
stepinfo(out.simout.Data,out.simout.Time,1);
end
end
save('ResultsP1', 'Results')
str = 3; step = 2;
```

```
for cost = 1:5
X = Results.str{str}.step{step}.perfor{cost}.x(end,:);
if
numel(Results.str{str}.step{step}.perfor{cost})>0
[fis1, fis2] = FUZZY4([1 1 1 1 1 1 1 1 1 1]);
out = sim('str3.slx');
Results.str{str}.step{step}.perfor{cost}.y = out.sim
out.Data;
Results.str{str}.step{step}.perfor{cost}.t = out.sim
out.Time;
Results.str{str}.step{step}.perfor{cost}.stepinto =
stepinfo(out.simout.Data,out.simout.Time,1);
end
end
save('ResultsP1', 'Results')
clear
clc
num = 9;
denum = [1 3 11 9];
load ('ResultsP1');
load_system('str3.slx');
str = 3; step = 3;
for cost = 2:3
X = Results.str{str}.step{1,2}.perfor{cost}.x(end,:);
if
numel(Results.str{str}.step{step}.perfor{cost})>0
[fis1,
fis2] = FUZZY4([Results.str{str}.step{step}.perfor{c
ost}.x(end,:), 1, 1,1,1 ]);
out = sim('str3.slx');
Results.str{str}.step{step}.perfor{cost}.y = out.sim
out.Data;
Results.str{str}.step{step}.perfor{cost}.t = out.sim
out.Time;
Results.str{str}.step{step}.perfor{cost}.stepinto =
stepinfo(out.simout.Data,out.simout.Time,1);
end
end
save('ResultsP1', 'Results')
str = 3; step = 4;
for cost = 2:3
X = Results.str{str}.step{1,2}.perfor{cost}.x(end,:)
;
if
numel(Results.str{str}.step{step}.perfor{cost})>0
[fis1,
fis2] = FUZZY4([Results.str{str}.step{1,3}.perfor{co
st}.x(end,:),
Results.str{str}.step{step}.perfor{cost}.x(end,:)
]);
out = sim('str3.slx');
Results.str{str}.step{step}.perfor{cost}.y = out.sim
out.Data;
Results.str{str}.step{step}.perfor{cost}.t = out.sim
out.Time;
Results.str{str}.step{step}.perfor{cost}.stepinto =
stepinfo(out.simout.Data,out.simout.Time,1);
end
end
save('ResultsP1','Results')
str = 3; step = 5;
for cost = 2:3
X = Results.str{str}.step{1,2}.perfor{cost}.x(end,:);
if
numel(Results.str{str}.step{step}.perfor{cost})>0
[fis1,
fis2] = FUZZY4([1,1,1,1,1,1
Results.str{str}.step{step}.perfor{co
st}.x(end,:) ]);
out = sim('str3.slx');
Results.str{str}.step{step}.perfor{cost}.y = out.sim
out.Data;
Results.str{str}.step{step}.perfor{cost}.t = out.sim
out.Time;
Results.str{str}.step{step}.perfor{cost}.stepinto =
stepinfo(out.simout.Data,out.simout.Time,1);
end
end
save('ResultsP1','Results')
%% Functions
function [C] = optimPID1
global G str step Results cost
% Initial parameters using stability based tuning
if step ==1
PID = pidtune(G,'PID');
switch str
case 1
A = 1;D = PID.Ki;CC = abs(PID.Kp+sqrt(PID.Kp^2-
4*PID.Kd*PID.Ki));B = PID.Kd/CC;
x0 = [A, B,CC, D]; %x1 = [0 0 0 0];x2 = [7 7 7 7];
case 2
A = PID.Kp;B = PID.Kd;D = PID.Ki;CC = 1;
x0 = [A, B,CC, D];% x1 = [0 0 0 0];x2 = [7 7 7 7];
case 3
A1 = PID.Kp/2;A2 = PID.Kp/2;B = PID.Kd;D = PID.Ki;CC = 1;
x0 = [A1,A2, B,CC, D]; %x1 = [0 0 0 0 0];x2 = [7 7 7 7];
end
C = runfmincon1(@obj_FOPID2,x0);
end
if step ==2
x0 = [Results.str{str}.step{1}.perfor{cost}.x(end,:)
,1,1];% x1 = [x1,0.7, 0.7];x2 = [x2, 1.3, 1.3];
C = runfmincon1(@obj_FOPID2,x0);
end
end
function [history] = runfmincon1(obj,x0)
i = 1;
history.x = [];
history.fval = [];
options = optimset('OutputFcn',@outfun,...
'Algorithm','active-set');
fminsearch(obj,x0,options)
function stop = outfun(x,optimValues,state)
stop = false;
switch state
case 'iter'
history.fval = [history.fval; optimValues.fval];
xx = abs(x);
if size(x,2)>4
xx(5:6) = min(1.5,xx(5:6));
end
history.x = [history.x; xx];
i = i+1;
otherwise
end
end
end
function [obj] = obj_FOPID2 (BCD)
global cost
BCD = abs(BCD);
for ii = 1:size(BCD,1)
[t,y] = stepf(BCD(ii,:));
dt = [t(2:end),t(end)]-t;
e = 1-y;
switch cost
case 1 %ISE
Value = sum(e.^2.*dt);
case 2 % ITSE
Value = sum((t.*e).^2.*dt);
case 3 % IT2SE
Value = sum((t.^2.*e).^2.*dt);
case 4 %'IAE'
Value = sum(abs(e).*dt);
case 5% 'MPC'
Value = (0.4*sum(abs(e).*dt)+0.4*sum(e.^2.*dt)+0.2*s
um((t.*e).^2.*dt));
end
obj(ii) = Value+simple_constraint(BCD);
end
end
function [tv,out] = stepf (BCD)
global Gff str
BCD = abs(BCD);
t1 = 0;t2 = 60;dt = 0.2;
wh = 100;ws = 0.01;
kf = (wh/ws);
```

```
k = 1:2:kf+1;
out = [];
ii = 1;
t = t1;
w = k.*ws;
s = sqrt(-1)*w;
G = Gff(s);
switch str
case 1
A = abs(BCD(1));B = abs(BCD(2));
C = abs(BCD(3));D = abs(BCD(4));
if numel(BCD)>4
L = min(abs(BCD(5)),1.5);Mu = min(1.5,abs(BCD(6)));
else
L = 1;Mu = 1;
end
CCG = (A+B.*s.^Mu).*((C+D./s.^L));
case 2
A = abs(BCD(1));B = abs(BCD(2));
C = abs(BCD(3));D = abs(BCD(4));
if numel(BCD)>4
L = min(abs(BCD(5)),1.5);Mu = min(1.5,abs(BCD(6)));
else
L = 1;Mu = 1;
end
CCG = (A+B.*s.^Mu+D./s.^L).*C;
case 3
A1 = abs(BCD(1))/2;A2 = abs(BCD(1))/2;B = abs(BCD(2));
C = abs(BCD(3));D = abs(BCD(4));
if numel(BCD)>5
L = min(abs(BCD(5)),1.5);Mu = min(1.5,abs(BCD(6)));
else
L = 1;Mu = 1;
end
CCG = (A1+A2+B.*s.^Mu+D./s.^L)*C;
end
R = real(G.*CCG./(1+G.*CCG));
I = 0;%imag(G.*CCG./(1+G.*CCG));
II = 1e-3;
while t<t2
out(ii) = sum(R.*sin(w.*t)./k+I.*cos(w.*t)./k)*4/pi;
tv(ii) = t;
ii = ii+1;
t = t+dt;
end
end
function [ceq] = simple_constraint(BCD)
BCD = abs(BCD);
global G str step
switch step
case 1
s = tf('s');
switch str
case 1
CCG = (BCD(1)+BCD(2)*s)*(BCD(3)+BCD(4)/s);
case 2
CCG = (BCD(1)+BCD(2)*s+BCD(4)/s)*BCD(3);
case 3
CCG = (BCD(1)+BCD(2)*s+BCD(4)/s)*BCD(3);
end
case 2
s = fotf('s');
switch str
case 1
CC = (BCD(1)+BCD(2)*s^BCD(6))*(BCD(3)+BCD(4)/s^BCD(5
));
case 2
CC = (BCD(1)+BCD(2)*s^BCD(6)+BCD(4)/s^BCD(5))*BCD(3)
;
case 3
CC = (BCD(1)+BCD(2)*s^BCD(6)+BCD(6)/s^BCD(5))*BCD(3)
;
end
CCG = oustapp(CC,5);
end
TF = CCG*G;
[Gm,Pm,Wcg,Wcp,S] = margin(TF);
if S==0
ceq = 100;
else
ceq = 0;
end
if Wcp>5
ceq = ceq+Wcp-5;%abs(TF(j*omega_c))-1;
end
if Pm<30
ceq = ceq+30-Pm;
end
if Gm<2
ceq = ceq+2-Gm;
end
end
function H = optimPID2
global str step cost Results
switch str
case 1
switch step
case 3
X0 = [1,1,1];
H = runfmincon(@obj_Fuzzy_FOPID_str1,X0);
case 4
H = runfmincon(@obj_Fuzzy_FOPID_str1,[1,1,0,0]);
case 5
H = runfmincon(@obj_Fuzzy_FOPID_str1,[1 1,0,0]);
end
case 2
switch step
case 3
H = runfmincon(@obj_Fuzzy_FOPID_str1,[1,1,1,1]);
case 4
H = runfmincon(@obj_Fuzzy_FOPID_str1,[1,1,1,0,0,0]);
case 5
H = runfmincon(@obj_Fuzzy_FOPID_str1,[1,1,1,0,0,0]);
end
case 3
H = runfmincon(@obj_Fuzzy_FOPID_str1,[1,1,1,1,1,1]);
case 4
H = runfmincon(@obj_Fuzzy_FOPID_str1,[1,1,1,1,0,0,0,
0]);
case 5
H = runfmincon(@obj_Fuzzy_FOPID_str1,[1,1,1,1,0,0,0,
0]);
end
end
function [history] = runfmincon(obj,x0)
i = 1;
history.x = [];
history.fval = [];
options =  optimset('OutputFcn',@outfun,'Maxiter',25);%,...
NewTLBO(obj,x0,options)
function stop = outfun(x,optimValues,state)
stop = false;
switch state
case 'iter'
history.fval = [history.fval; optimValues.fval];
history.x = [history.x; x];
i = i+1;
otherwise
end
end
end
function obj = obj_Fuzzy_FOPID_str1(X)
ii = 1;
global cost step Results str
load_system('str1.slx');
switch str
case 1
switch step
case 3
fis2 = FUZZY2([X(ii,:) 1 1 0 0]);
case 4
fis2 = FUZZY2([Results.str{str}.step{3}.perfor{cost}
.x(end,:), X(ii,:) ]);
case 5
fis2 = FUZZY2([1 1 1 X(ii,:) ]);
end
```

```
assignin ('base','fis2',fis2)
out = sim('str1.slx');
e = 1-out.simout.Data;
t = out.simout.Time;
dt = [t(2:end);20]-t;
case 2
switch step
case 3
fis3 = FUZZY3([X(ii,:) 1 1 1,1,0,0,0,0]);
case 4
fis3 = FUZZY3([Results.str{str}.step{3}.perfo
r{cost}.x(end,:), X(ii,:) ]);
case 5
fis3 = FUZZY3([1, 1, 1,1, X(ii,:) ]);
end
assignin ('base','fis3',fis3)
out = sim('str2.slx');
e = 1-out.simout.Data;
t = out.simout.Time;
dt = [t(2:end);20]-t;
case 3
switch step
case 3
[fis1,fis2] = FUZZY4([X(ii,:), 1, 1, 1, 1,0,0,0,0]);
case 4
[fis1,fis2] = FUZZY4([Results.str{str}.step{3}.perfo
r{cost}.x(end,:), X(ii,:) ]);
case 5
[fis1,fis2] = FUZZY4([1, 1, 1,1,1,1, X(ii,:) ]);
end
assignin ('base','fis1',fis1)
assignin ('base','fis2',fis2)
out = sim('str3.slx');
e = 1-out.simout.Data;
t = out.simout.Time;
dt = [t(2:end);20]-t;
end
switch cost
case 1%'ISE'
Value = sum(e.^2.*dt);
case 2%'ITSE'
Value = sum((t.*e).^2.*dt);
case 3%'IT2SE'
Value = sum((t.^2.*e).^2.*dt);
case 4%'IAE'
Value = sum(abs(e).*dt);
case 5% 'FFF'
Value = 0.4*sum(abs(e).*dt)+0.4*sum(e.^2.*dt)+0.2*su
m((t.*e).^2.*dt);
end
obj = Value;
end
function [FIS1] = FUZZY2(X)
L = X(1:3);
alpha = X(4:5);
beta = X(6:7);
alpha = min(alpha,[1 1]);
alpha = max(alpha,[0.1 0.1]);
beta = min(beta,[1 1]);
beta = max(beta,[0.0 0.0]);
L = min(L,[7 7 7]);
L = max(L,[0.1 0.1 0.1]);
fis1 = sugfis;
L1 = L(1);
L2 = L(2);
Lo = L(3);
p1 = 1/(1+L1);
p2 = 1/(1+L2);
po1 = 1/(Lo^2*(Lo+1)+Lo+1);
po2 = po1*(Lo+1);
po3 = Lo*(po2-po1)+po2;
fis1 = addInput(fis1,[-10e4 10e4],'Name','E');
fis1 = addInput(fis1,[-10e4 10e4],'Name','delE');
% fis1 = addInput(fis1,[0 10*td],'Name','T');
fis1 = addMF(fis1,'E','trimf',[-11e4 -10e4 -
10e4*p1 ],'Name','NB');
fis1 = addMF(fis1,'E','trimf',[-10e4 -10e4*p1
0],'Name','NM');
fis1 = addMF(fis1,'E','trimf',[-10e4*p1 0 10e4*p1
],'Name','Z');
fis1 = addMF(fis1,'E','trimf',[0 10e4*p1
10e4],'Name','PM');
fis1 = addMF(fis1,'E','trimf',[p1*10e4 10e4
11e4],'Name','PB');
fis1 = addMF(fis1,'delE','trimf',[-11e4 -10e4 -
10e4*p2 ],'Name','NB');
fis1 = addMF(fis1,'delE','trimf',[-10e4 -10e4*p2
0],'Name','NM');
fis1 = addMF(fis1,'delE','trimf',[-10e4*p2 0
10e4*p2 ],'Name','Z');
fis1 = addMF(fis1,'delE','trimf',[0 p2*10e4
10e4],'Name','PM');
fis1 = addMF(fis1,'delE','trimf',[p2*10e4 10e4
11e4],'Name','PB');
fis1 = addOutput(fis1,[-10e4 10e4],'Name','U');
fis1 = addMF(fis1,'U','constant', -
10e4,'Name','NBB');
fis1 = addMF(fis1,'U','constant'-
10e4*po3,'Name','NB');
fis1 = addMF(fis1,'U','constant',-
10e4*po2,'Name','NM');
fis1 = addMF(fis1,'U','constant',-
10e4*po1,'Name','N');
fis1 = addMF(fis1,'U','constant', 0.0,'Name','Z');
fis1 =
 addMF(fis1,'U','constant',po1*10e4,'Name','P');
fis1 =
 addMF(fis1,'U','constant',po2*10e4,'Name','PM');
fis1 =
 addMF(fis1,'U','constant',po3*10e4,'Name','PB');
fis1 =
 addMF(fis1,'U','constant',10e4,'Name','PBB');
rules = [...
"E==NB & delE==NB => U = NBB"; ...
"E==NB & delE==NM => U = NB"; ...
"E==NB & delE==Z => U = NM"; ...
"E==NB & delE==PM => U = N"; ...
"E==NB & delE==PB => U = Z"; ...
"E==NM & delE==NB => U = NB"; ...
"E==NM & delE==NM => U = NM"; ...
"E==NM & delE==Z => U = N"; ...
"E==NM & delE==PM => U = Z"; ...
"E==NM & delE==PB => U = P"; ...
"E==Z & delE==NB => U = NM"; ...
"E==Z & delE==NM => U = N"; ...
"E==Z & delE==Z => U = Z"; ...
"E==Z & delE==PM => U = P"; ...
"E==Z & delE==PB => U = PM"; ...
"E==PM & delE==NB => U = N"; ...
"E==PM & delE==NM => U = Z"; ...
"E==PM & delE==Z => U = P"; ...
"E==PM & delE==PM => U = PM"; ...
"E==PM & delE==PB => U = PB"; ...
"E==PB & delE==NB => U = Z"; ...
"E==PB & delE==NM => U = P"; ...
"E==PB & delE==Z => U = PM"; ...
"E==PB & delE==PM => U = PB"; ...
"E==PB & delE==PB => U = PBB"; ...
];
fis1 = addRule(fis1,rules);
FIS1 = convertToType2(fis1);
for i = 1:length(FIS1.Inputs)
for j =
1:length(FIS1.Inputs(i).MembershipFunctions)
FIS1.Inputs(i).MembershipFunctions(j).LowerLag =
beta(i);
FIS1.Inputs(i).MembershipFunctions(j).LowerScale =
alpha(i);%scale(i,j);
end
end
end
function [FIS1] = FUZZY3(X)
L = X(1:4);
alpha = X(5:7);
beta = X(8:10);
alpha = min(alpha,[1 1 1]);
alpha = max(alpha,[0.1 0.1 0.1]);
beta = min(beta,[1 1 1]);
beta = max(beta,[0.0 0.0 0.0]);
L = min(L,[7 7 7 7]);
L = max(L,[0.1 0.1 0.1 0.1]);
fis1 = sugfis;
L1 = L(1);
L2 = L(2);
L3 = L(3);
Lo = L(4);
p1 = 1/(1+L1);
p2 = 1/(1+L2);
p3 = 1/(1+L3);
po5 = 1/(Lo+1-Lo/(Lo+1-Lo/(Lo+1-Lo/(Lo+1-Lo/(Lo+1)))));
po4 = po5/(Lo+1-Lo/(Lo+1-Lo/(Lo+1-Lo/(Lo+1))));
po3 = po4/(Lo+1-Lo/(Lo+1-Lo/(Lo+1)));
po2 = po3/(Lo+1-Lo/(Lo+1));
po1 = po2/(Lo+1);
fis1 = addInput(fis1,[-10e4 10e4],‘Name’,‘E’);
fis1 = addInput(fis1,[-10e4 10e4],‘Name’,‘delE’);
fis1 = addInput(fis1,[-10e4 10e4],‘Name’,‘IE’);
fis1 = addMF(fis1,‘E’,‘trimf’,[-11e4 -10e4 -
10e4*p1 ],‘Name’,‘NB’);
fis1 = addMF(fis1,‘E’,‘trimf’,[-10e4 -10e4*p1
0],‘Name’,‘NM’);
fis1 = addMF(fis1,‘E’,‘trimf’,[-10e4*p1 0 10e4*p1
],‘Name’,‘Z’);
fis1 = addMF(fis1,‘E’,‘trimf’,[0 10e4*p1
10e4],‘Name’,‘PM’);
fis1 = addMF(fis1,‘E’,‘trimf’,[p1*10e4 10e4
11e4],‘Name’,‘PB’);
fis1 = addMF(fis1,‘delE’,‘trimf’,[-11e4 -10e4 -
10e4*p2 ],‘Name’,‘NB’);
fis1 = addMF(fis1,‘delE’,‘trimf’,[-10e4 -10e4*p2
0],‘Name’,‘NM’);
fis1 = addMF(fis1,‘delE’,‘trimf’,[-10e4*p2 0
10e4*p2 ],‘Name’,‘Z’);
fis1 = addMF(fis1,‘delE’,‘trimf’,[0 p2*10e4
10e4],‘Name’,‘PM’);
fis1 = addMF(fis1,‘delE’,‘trimf’,[p2*10e4 10e4
11e4],‘Name’,‘PB’);
fis1 = addMF(fis1,‘IE’,‘trimf’,[-11e4 -10e4 -
10e4*p3 ],‘Name’,‘NB’);
fis1 = addMF(fis1,‘IE’,‘trimf’,[-10e4 -10e4*p3
0],‘Name’,‘NM’);
fis1 = addMF(fis1,‘IE’,‘trimf’,[-10e4*p3 0 10e4*p3
],‘Name’,‘Z’);
fis1 = addMF(fis1,‘IE’,‘trimf’,[ 0 10e4*p3
10e4],‘Name’,‘PM’);
fis1 = addMF(fis1,‘IE’,‘trimf’,[p3*10e4 10e4
11e4],‘Name’,‘PB’);
fis1 = addOutput(fis1,[-10e4 10e4],‘Name’,‘U’);
fis1 = addMF(fis1,‘U’,‘constant’, -
10e4,‘Name’,‘mf1’);
fis1 = addMF(fis1,‘U’,‘constant’, -
10e4*po5,‘Name’,‘mf2’);
fis1 = addMF(fis1,‘U’,‘constant’, -
10e4*po4,‘Name’,‘mf3’);
fis1 = addMF(fis1,‘U’,‘constant’, -
10e4*po3,‘Name’,‘mf4’);
fis1 = addMF(fis1,‘U’,‘constant’, -
10e4*po2,‘Name’,‘mf5’);
fis1 = addMF(fis1,‘U’,‘constant’, -
10e4*po1,‘Name’,‘mf6’);
fis1 = addMF(fis1,‘U’,‘constant’,0, ‘Name’,‘mf7’);
fis1 =
addMF(fis1,‘U’,‘constant’, po1*10e4,‘Name’,‘mf8’);
fis1 =
addMF(fis1,‘U’,‘constant’, po2*10e4,‘Name’,‘mf9’);
fis1 =
addMF(fis1,‘U’,‘constant’, po3*10e4,‘Name’,‘mf10’);
fis1 =
addMF(fis1,‘U’,‘constant’, po4*10e4,‘Name’,‘mf11’);
fis1 =
addMF(fis1,‘U’,‘constant’, po5*10e4,‘Name’,‘mf12’);
fis1 =
addMF(fis1,‘U’,‘constant’, 10e4,‘Name’,‘mf13’);
rules = [...
```

```
"IE==NB & E==NB & delE==NB => U = mf1"; ...
"IE==NB & E==NB & delE==NM => U = mf2"; ...
"IE==NB & E==NB & delE==Z => U = mf3"; ...
"IE==NB & E==NB & delE==PM => U = mf4"; ...
"IE==NB & E==NB & delE==PB => U = mf5"; ...
"IE==NB & E==NM & delE==NB => U = mf2"; ...
"IE==NB & E==NM & delE==NM => U = mf3"; ...
"IE==NB& E==NM & delE==Z => U = mf4"; ...
"IE==NB& E==NM & delE==PM => U = mf5"; ...
"IE==NB& E==NM & delE==PB => U = mf6"; ...
"IE==NB& E==Z & delE==NB => U = mf3"; ...
"IE==NB& E==Z & delE==NM => U = mf4"; ...
"IE==NB& E==Z & delE==Z => U = mf5"; ...
"IE==NB& E==Z & delE==PM => U = mf6"; ...
"IE==NB& E==Z & delE==PB => U = mf7"; ...
"IE==NB& E==PM & delE==NB => U = mf4"; ...
"IE==NB& E==PM & delE==NM => U = mf5"; ...
"IE==NB& E==PM & delE==Z => U = mf6"; ...
"IE==NB& E==PM & delE==PM => U = mf7"; ...
"IE==NB& E==PM & delE==PB => U = mf8"; ...
"IE==NB& E==PB & delE==NB => U = mf5"; ...
"IE==NB& E==PB & delE==NM => U = mf6"; ...
"IE==NB& E==PB & delE==Z => U = mf7"; ...
"IE==NB& E==PB & delE==PM => U = mf8"; ...
"IE==NB& E==PB & delE==PB => U = mf9"; ...
"IE==NM& E==NB & delE==NB => U = mf2"; ...
"IE==NM& E==NB & delE==NM => U = mf3"; ...
"IE==NM& E==NB & delE==Z => U = mf4"; ...
"IE==NM& E==NB & delE==PM => U = mf5"; ...
"IE==NM& E==NB & delE==PB => U = mf6"; ...
"IE==NM& E==NM & delE==NB => U = mf3"; ...
"IE==NM& E==NM & delE==NM => U = mf4"; ...
"IE==NM& E==NM & delE==Z => U = mf5"; ...
"IE==NM& E==NM & delE==PM => U = mf6"; ...
"IE==NM& E==NM & delE==PB => U = mf7"; ...
"IE==NM& E==Z & delE==NB => U = mf4"; ...
"IE==NM& E==Z & delE==NM => U = mf5"; ...
"IE==NM& E==Z & delE==Z => U = mf6"; ...
"IE==NM& E==Z & delE==PM => U = mf7"; ...
"IE==NM& E==Z & delE==PB => U = mf8"; ...
"IE==NM& E==PM & delE==NB => U = mf5"; ...
"IE==NM& E==PM & delE==NM => U = mf6"; ...
"IE==NM& E==PM & delE==Z => U = mf7"; ...
"IE==NM& E==PM & delE==PM => U = mf8"; ...
"IE==NM& E==PM & delE==PB => U = mf9"; ...
"IE==NM& E==PB & delE==NB => U = mf6"; ...
"IE==NM& E==PB & delE==NM => U = mf7"; ...
"IE==NM& E==PB & delE==Z => U = mf8"; ...
"IE==NM& E==PB & delE==PM => U = mf9"; ...
"IE==NM& E==PB & delE==PB => U = mf10"; ...
"IE==Z& E==NB & delE==NB => U = mf3"; ...
"IE==Z& E==NB & delE==NM => U = mf4"; ...
"IE==Z& E==NB & delE==Z => U = mf5"; ...
"IE==Z& E==NB & delE==PM => U = mf6"; ...
"IE==Z& E==NB & delE==PB => U = mf7"; ...
"IE==Z& E==NM & delE==NB => U = mf4"; ...
"IE==Z& E==NM & delE==NM => U = mf5"; ...
"IE==Z& E==NM & delE==Z => U = mf6"; ...
"IE==Z& E==NM & delE==PM => U = mf7"; ...
"IE==Z& E==NM & delE==PB => U = mf8"; ...
"IE==Z& E==Z & delE==NB => U = mf5"; ...
"IE==Z& E==Z & delE==NM => U = mf6"; ...
"IE==Z& E==Z & delE==Z => U = mf7"; ...
"IE==Z& E==Z & delE==PM => U = mf8"; ...
"IE==Z& E==Z & delE==PB => U = mf9"; ...
"IE==Z& E==PM & delE==NB => U = mf6"; ...
"IE==Z& E==PM & delE==NM => U = mf7"; ...
"IE==Z& E==PM & delE==Z => U = mf8"; ...
"IE==Z& E==PM & delE==PM => U = mf9"; ...
"IE==Z& E==PM & delE==PB => U = mf10"; ...
"IE==Z& E==PB & delE==NB => U = mf7"; ...
"IE==Z& E==PB & delE==NM => U = mf8"; ...
"IE==Z& E==PB & delE==Z => U = mf9"; ...
"IE==Z& E==PB & delE==PM => U = mf10"; ...
"IE==Z& E==PB & delE==PB => U = mf11"; ...
"IE==PM& E==NB & delE==NB => U = mf4"; ...
"IE==PM& E==NB & delE==NM => U = mf5"; ...
"IE==PM& E==NB & delE==Z => U = mf6"; ...
```

```
"IE==PM& E==NB & delE==PM => U = mf7"; ...
"IE==PM& E==NB & delE==PB => U = mf8"; ...
"IE==PM& E==NM & delE==NB => U = mf5"; ...
"IE==PM& E==NM & delE==NM => U = mf6"; ...
"IE==PM& E==NM & delE==Z => U = mf7"; ...
"IE==PM& E==NM & delE==PM => U = mf8"; ...
"IE==PM& E==NM & delE==PB => U = mf9"; ...
"IE==PM& E==Z & delE==NB => U = mf6"; ...
"IE==PM& E==Z & delE==NM => U = mf7"; ...
"IE==PM& E==Z & delE==Z => U = mf8"; ...
"IE==PM& E==Z & delE==PM => U = mf9"; ...
"IE==PM& E==Z & delE==PB => U = mf10"; ...
"IE==PM& E==PM & delE==NB => U = mf7"; ...
"IE==PM& E==PM & delE==NM => U = mf8"; ...
"IE==PM& E==PM & delE==Z => U = mf9"; ...
"IE==PM& E==PM & delE==PM => U = mf10"; ...
"IE==PM& E==PM & delE==PB => U = mf11"; ...
"IE==PM& E==PB & delE==NB => U = mf8"; ...
"IE==PM& E==PB & delE==NM => U = mf9"; ...
"IE==PM& E==PB & delE==Z => U = mf10"; ...
"IE==PM& E==PB & delE==PM => U = mf11"; ...
"IE==PM& E==PB & delE==PB => U = mf12"; ...
"IE==PB& E==NB & delE==NB => U = mf5"; ...
"IE==PB& E==NB & delE==NM => U = mf6"; ...
"IE==PB& E==NB & delE==Z => U = mf7"; ...
"IE==PB& E==NB & delE==PM => U = mf8"; ...
"IE==PB& E==NB & delE==PB => U = mf9"; ...
"IE==PB& E==NM & delE==NB => U = mf6"; ...
"IE==PB& E==NM & delE==NM => U = mf7"; ...
"IE==PB& E==NM & delE==Z => U = mf8"; ...
"IE==PB& E==NM & delE==PM => U = mf9"; ...
"IE==PB& E==NM & delE==PB => U = mf10"; ...
"IE==PB& E==Z & delE==NB => U = mf7"; ...
"IE==PB& E==Z & delE==NM => U = mf8"; ...
"IE==PB& E==Z & delE==Z => U = mf9"; ...
"IE==PB& E==Z & delE==PM => U = mf10"; ...
"IE==PB& E==Z & delE==PB => U = mf11"; ...
"IE==PB& E==PM & delE==NB => U = mf8"; ...
"IE==PB& E==PM & delE==NM => U = mf9"; ...
"IE==PB& E==PM & delE==Z => U = mf10"; ...
"IE==PB& E==PM & delE==PM => U = mf11"; ...
"IE==PB& E==PM & delE==PB => U = mf12"; ...
"IE==PB& E==PB & delE==NB => U = mf9"; ...
"IE==PB& E==PB & delE==NM => U = mf10"; ...
"IE==PB& E==PB & delE==Z => U = mf11"; ...
"IE==PB& E==PB & delE==PM => U = mf12"; ...
"IE==PB& E==PB & delE==PB => U = mf13"; ...
];
fis1 = addRule(fis1,rules);
FIS1 = convertToType2(fis1);
for i = 1:length(FIS1.Inputs)
for j =
1:length(FIS1.Inputs(i).MembershipFunctions)
FIS1.Inputs(i).MembershipFunctions(j).LowerLag =
 beta(i);
FIS1.Inputs(i).MembershipFunctions(j).LowerScale =
 alpha(i);%scale(i,j);
end
end
end
function [FIS1, FIS2] = FUZZY4(X)
L = X(1:3);alpha = X(7:8);beta = X(11:12);
alpha = min(alpha,[1 1]);
alpha = max(alpha,[0.1 0.1]);
beta = min(beta,[1 1]);
beta = max(beta,[0.0 0.0]);
L = min(L,[7 7 7]);
L = max(L,[0.1 0.1 0.1]);
fis1 = sugfis;
L1 = L(1);
L2 = L(2);
Lo = L(3);
p1 = 1/(1+L1);
p2 = 1/(1+L2);
po1 = 1/(Lo^2*(Lo+1)+Lo+1);
po2 = po1*(Lo+1);
po3 = Lo*(po2-po1)+po2;
fis1 = addInput(fis1,[-10e4 10e4],'Name','E');
fis1 = addInput(fis1,[-10e4 10e4],'Name','delE');
fis1 = addMF(fis1,'E','trimf',[-11e4 -10e4 -
10e4*p1 ],'Name','NB');
fis1 = addMF(fis1,'E','trimf',[-10e4 -10e4*p1
0],'Name','NM');
fis1 = addMF(fis1,'E','trimf',[-10e4*p1 0 10e4*p1
],'Name','Z');
fis1 = addMF(fis1,'E','trimf',[0 10e4*p1
10e4],'Name','PM');
fis1 = addMF(fis1,'E','trimf',[p1*10e4 10e4
11e4],'Name','PB');
fis1 = addMF(fis1,'delE','trimf',[-11e4 -10e4 -
10e4*p2 ],'Name','NB');
fis1 = addMF(fis1,'delE','trimf',[-10e4 -10e4*p2
 0],'Name','NM');
fis1 = addMF(fis1,'delE','trimf',[-10e4*p2 0
10e4*p2 ],'Name','Z');
fis1 = addMF(fis1,'delE','trimf',[0 p2*10e4
10e4],'Name','PM');
fis1 = addMF(fis1,'delE','trimf',[p2*10e4 10e4
11e4],'Name','PB');
fis1 = addOutput(fis1,[-10e4 10e4],'Name','U');
fis1 = addMF(fis1,'U','constant',-
10e4, 'Name','NBB');
fis1 = addMF(fis1,'U','constant',-
10e4*po3, 'Name','NB');
fis1 = addMF(fis1,'U','constant',-
10e4*po2, 'Name','NM');
fis1 = addMF(fis1,'U','constant',-
10e4*po1, 'Name','N');
fis1 = addMF(fis1,'U','constant',0.0, 'Name','Z');
fis1 =
 addMF(fis1,'U','constant',po1*10e4, 'Name','P');
fis1 =
 addMF(fis1,'U','constant',po2*10e4, 'Name','PM');
fis1 =
 addMF(fis1,'U','constant',po3*10e4, 'Name','PB');
fis1 =
 addMF(fis1,'U','constant',10e4,'Name','PBB');
rules = [...
"E==NB & delE==NB => U = NBB"; ...
"E==NB & delE==NM => U = NB"; ...
"E==NB & delE==Z => U = NM"; ...
"E==NB & delE==PM => U = N"; ...
"E==NB & delE==PB => U = Z"; ...
"E==NM & delE==NB => U = NB"; ...
"E==NM & delE==NM => U = NM"; ...
"E==NM & delE==Z => U = N"; ...
"E==NM & delE==PM => U = Z"; ...
"E==NM & delE==PB => U = P"; ...
"E==Z & delE==NB => U = NM"; ...
"E==Z & delE==NM => U = N"; ...
"E==Z & delE==Z => U = Z"; ...
"E==Z & delE==PM => U = P"; ...
"E==Z & delE==PB => U = PM"; ...
"E==PM & delE==NB => U = N"; ...
"E==PM & delE==NM => U = Z"; ...
"E==PM & delE==Z => U = P"; ...
"E==PM & delE==PM => U = PM"; ...
"E==PM & delE==PB => U = PB"; ...
"E==PB & delE==NB => U = Z"; ...
"E==PB & delE==NM => U = P"; ...
"E==PB & delE==Z => U = PM"; ...
"E==PB & delE==PM => U = PB"; ...
"E==PB & delE==PB => U = PBB"; ...
];
fis1 = addRule(fis1,rules);
FIS1 = convertToType2(fis1);
for i = 1:length(FIS1.Inputs)
for j =
1:length(FIS1.Inputs(i).MembershipFunctions)
FIS1.Inputs(i).MembershipFunctions(j).LowerLag =
beta(i);
FIS1.Inputs(i).MembershipFunctions(j).LowerScale =
alpha(i);%scale(i,j);
end
end
L = X(4:6);alpha = X(9:10);beta = X(13:14);
```

```
alpha = min(alpha,[1 1]);
alpha = max(alpha,[0.1 0.1]);
beta = min(beta,[1 1]);
beta = max(beta,[0.0 0.0]);
L = min(L,[7 7 7]);
L = max(L,[0.1 0.1 0.1]);
fis2 = sugfis;
L1 = L(1);
L2 = L(2);
Lo = L(3);
p1 = 1/(1+L1);
p2 = 1/(1+L2);
po1 = 1/(Lo^2*(Lo+1)+Lo+1);
po2 = po1*(Lo+1);
po3 = Lo*(po2-po1)+po2;
fis2 = addInput(fis2,[-10e4 10e4],'Name','E');
fis2 = addInput(fis2,[-10e4 10e4],'Name','delE');
fis2 = addMF(fis2,'E','trimf',[-11e4 -10e4 -
10e4*p1 ],'Name','NB');
fis2 = addMF(fis2,'E','trimf',[-10e4 -10e4*p1
 0],'Name','NM');
fis2 = addMF(fis2,'E','trimf',[-10e4*p1 0 10e4*p1
],'Name','Z');
fis2 = addMF(fis2,'E','trimf',[0 10e4*p1
10e4],'Name','PM');
fis2 = addMF(fis2,'E','trimf',[p1*10e4 10e4
11e4],'Name','PB');
fis2 = addMF(fis2,'delE','trimf',[-11e4 -10e4 -
10e4*p2 ],'Name','NB');
fis2 = addMF(fis2,'delE','trimf',[-10e4 -10e4*p2
0],'Name','NM');
fis2 = addMF(fis2,'delE','trimf',[-10e4*p2 0
10e4*p2 ],'Name','Z');
fis2 = addMF(fis2,'delE','trimf',[0 p2*10e4
10e4],'Name','PM');
fis2 = addMF(fis2,'delE','trimf',[p2*10e4 10e4
11e4],'Name','PB');
fis2 = addOutput(fis2,[-10e4 10e4],'Name','U');
fis2 = addMF(fis2,'U','constant', -
10e4,'Name','NBB');
fis2 = addMF(fis2,'U','constant', -
10e4*po3,'Name','NB');
fis2 = addMF(fis2,'U','constant', -
10e4*po2,'Name','NM');
fis2 = addMF(fis2,'U','constant', -
10e4*po1,'Name','N');
fis2 = addMF(fis2,'U','constant',0.0, 'Name','Z');
fis2 =
addMF(fis2,'U','constant', po1*10e4,'Name','P');
fis2 =
addMF(fis2,'U','constant', po2*10e4,'Name','PM');
fis2 =
addMF(fis2,'U','constant', po3*10e4,'Name','PB');
fis2 =
addMF(fis2,'U','constant',10e4, 'Name', 'PBB');
rules = [...
"E==NB & delE==NB => U = NBB"; ...
"E==NB & delE==NM => U = NB"; ...
"E==NB & delE==Z => U = NM"; ...
"E==NB & delE==PM => U = N"; ...
"E==NB & delE==PB => U = Z"; ...
"E==NM & delE==NB => U = NB"; ...
"E==NM & delE==NM => U = NM"; ...
"E==NM & delE==Z => U = N"; ...
"E==NM & delE==PM => U = Z"; ...
"E==NM & delE==PB => U = P"; ...
"E==Z & delE==NB => U = NM"; ...
"E==Z & delE==NM => U = N"; ...
"E==Z & delE==Z => U = Z"; ...
"E==Z & delE==PM => U = P"; ...
"E==Z & delE==PB => U = PM"; ...
"E==PM & delE==NB => U = N"; ...
"E==PM & delE==NM => U = Z"; ...
"E==PM & delE==Z => U = P"; ...
"E==PM & delE==PM => U = PM"; ...
"E==PM & delE==PB => U = PB"; ...
"E==PB & delE==NB => U = Z"; ...
"E==PB & delE==NM => U = P"; ...
"E==PB & delE==Z => U = PM"; ...
"E==PB & delE==PM => U = PB"; ...
"E==PB & delE==PB => U = PBB"; ...
];
fis2 = addRule(fis2,rules);
FIS2 = convertToType2(fis2);
for i = 1:length(FIS2.Inputs)
for j = 1:length(FIS2.Inputs(i).MembershipFunctions)
FIS2.Inputs(i).MembershipFunctions(j).LowerLag =
beta(i);%0.0;
FIS2.Inputs(i).MembershipFunctions(j).LowerScale =
alpha(i);%scale(i,j);
end
end

end
```

APPENDIX B
MATLAB SIMULINK

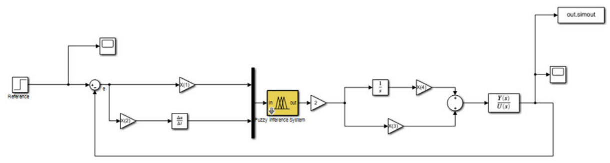

FIGURE 22. Structure 1.

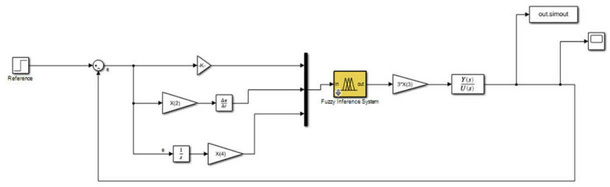

FIGURE 23. Structure 2.

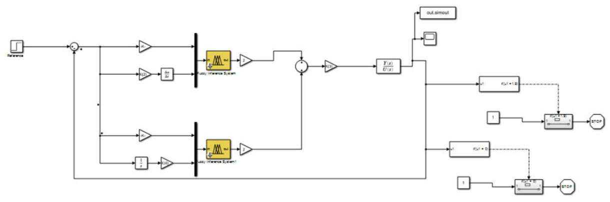

FIGURE 24. Structure 3.

REFERENCES

[1] D. Driankov, H. Hellendoorn, and M. Reinfrank, *An Introduction to Fuzzy Control*. Berlin, Germany: Springer, 2013.

[2] H. Zhang and D. Liu, *Fuzzy Modeling and Fuzzy Control*. Berlin, Germany: Springer, 2007.

[3] J. M. Mendel and R. I. B. John, "Type-2 fuzzy sets made simple," *IEEE Trans. Fuzzy Syst.*, vol. 10, no. 2, pp. 117–127, Apr. 2002, doi: 10.1109/91.995115.

[4] P. Shah and S. Agashe, "Review of fractional PID controller," *Mechatronics*, vol. 38, pp. 29–41, Sep. 2016, doi: 10.1016/j.mechatronics.2016.06.005.

[5] A. Kumar and V. Kumar, "A novel interval type-2 fractional order fuzzy PID controller: Design, performance evaluation, and its optimal time domain tuning," *ISA Trans.*, vol. 68, pp. 251–275, May 2017, doi: 10.1016/j.isatra.2017.03.022.

[6] S. Das, I. Pan, S. Das, and A. Gupta, "A novel fractional order fuzzy PID controller and its optimal time domain tuning based on integral performance indices," *Eng. Appl. Artif. Intell.*, vol. 25, no. 2, pp. 430–442, Mar. 2012, doi: 10.1016/j.engappai.2011.10.004.

[7] H. K. Abdulkhader, J. Jacob, and A. T. Mathew, "Robust type-2 fuzzy fractional order PID controller for dynamic stability enhancement of power system having RES based microgrid penetration," *Int. J. Electr. Power Energy Syst.*, vol. 110, pp. 357–371, Sep. 2019, doi: 10.1016/j.ijepes.2019.03.027.

[8] R. Mohammadikia and M. Aliasghary, ‘‘Design of an interval type-2 fractional order fuzzy controller for a tractor active suspension system,’’ *Comput. Electron. Agricult.*, vol. 167, Dec. 2019, Art. no. 105049, doi: 10.1016/j.compag.2019.105049.

[9] P. K. Ray, S. R. Paital, A. Mohanty, Y. S. E. Foo, A. Krishnan, H. B. Gooi, and G. A. J. Amaratunga, ‘‘A hybrid firefly-swarm optimized fractional order interval type-2 fuzzy PID-PSS for transient stability improvement,’’ *IEEE Trans. Ind. Appl.*, vol. 55, no. 6, pp. 6486–6498, Nov. 2019, doi: 10.1109/tia.2019.2938473.

[10] J. Z. Shi, ‘‘A fractional order general type-2 fuzzy PID controller design algorithm,’’ *IEEE Access*, vol. 8, pp. 52151–52172, 2020, doi: 10.1109/access.2020.2980686.

[11] C. A. Monje, B. M. Vinagre, V. Feliu, and Y. Chen, ‘‘On auto-tuning of fractional order PI$^{\lambda}$ D$^{\mu}$ controllers,’’ *IFAC Proc. Volumes*, vol. 39, no. 11, pp. 34–39, Jan. 2006, doi: 10.3182/20060719-3-pt-4902.00005.

[12] G. Sahoo, R. K. Sahu, S. Panda, N. R. Samal, and Y. Arya, ‘‘Modified Harris Hawks optimization-based fractional-order fuzzy PID controller for frequency regulation of multi-micro-grid,’’ *Arabian J. Sci. Eng.*, Feb. 2023, doi: 10.1007/s13369-023-07613-2.

[13] A. T. Azar, F. E. Serrano, and A. Koubaa, ‘‘Adaptive fuzzy type-2 fractional order proportional integral derivative sliding mode controller for trajectory tracking of robotic manipulators,’’ in *Proc. IEEE Int. Conf. Auto. Robot Syst. Competitions (ICARSC)*, Apr. 2020, pp. 183–187, doi: 10.1109/ICARSC49921.2020.9096163.

[14] A. Kumar and V. Kumar, ‘‘Performance analysis of optimal hybrid novel interval type-2 fractional order fuzzy logic controllers for fractional order systems,’’ *Exp. Syst. Appl.*, vol. 93, pp. 435–455, Mar. 2018, doi: 10.1016/j.eswa.2017.10.033.

[15] C.-T. Chao, N. Sutarna, J.-S. Chiou, and C.-J. Wang, ‘‘Equivalence between fuzzy PID controllers and conventional PID controllers,’’ *Appl. Sci.*, vol. 7, no. 6, p. 513, Jun. 2017, doi: 10.3390/app7060513.

[16] C.-T. Chao, N. Sutarna, J.-S. Chiou, and C.-J. Wang, ‘‘An optimal fuzzy PID controller design based on conventional PID control and nonlinear factors,’’ *Appl. Sci.*, vol. 9, no. 6, p. 1224, Mar. 2019, doi: 10.3390/app9061224.

[17] Q. Liang and J. M. Mendel, ‘‘Interval type-2 fuzzy logic systems: Theory and design,’’ *IEEE Trans. Fuzzy Syst.*, vol. 8, no. 5, pp. 535–550, Oct. 2000, doi: 10.1109/91.873577.

[18] N. N. Karnik and J. M. Mendel, ‘‘Centroid of a type-2 fuzzy set,’’ *Inf. Sci.*, vol. 132, nos. 1–4, pp. 195–220, 2001, doi: 10.1016/s0020-0255(01)00069-x.

[19] D. Wu and J. M. Mendel, ‘‘Enhanced Karnik–Mendel algorithms,’’ *IEEE Trans. Fuzzy Syst.*, vol. 17, no. 4, pp. 923–934, Aug. 2009, doi: 10.1109/tfuzz.2008.924329.

[20] K. Duran, H. Bernal, and M. Melgarejo, ‘‘Improved iterative algorithm for computing the generalized centroid of an interval type-2 fuzzy set,’’ in *Proc. Annu. Meeting North Amer. Fuzzy Inf. Process. Soc.*, New York, NY, USA, May 2008, pp. 1–5, doi: 10.1109/NAFIPS.2008.4531244.

[21] I. Podlubny, ‘‘Fractional-order systems and PI$^{\lambda}$ D$^{\mu}$-controllers,’’ *IEEE Trans. Autom. Control*, vol. 44, no. 1, pp. 208–214, Jan. 1999, doi: 10.1109/9.739144.

[22] W. Krajewski and U. Viaro, ‘‘A method for the integer-order approximation of fractional-order systems,’’ *J. Franklin Inst.*, vol. 351, no. 1, pp. 555–564, Jan. 2014, doi: 10.1016/j.jfranklin.2013.09.005.

[23] F. Mainardi. (2010). *Fractional Calculus and Waves in Linear Viscoelasticity: An Introduction to Mathematical Models*. [Online]. Available: http://ci.nii.ac.jp/ncid/BB03081382

[24] I. Podlubny, *Fractional Differential Equations—An Introduction to Fractional Derivatives, Fractional Differential Equations, to Methods of Their Solution and Some of Their Applications*. Amsterdam, The Netherlands: Elsevier, 1999, doi: 10.1016/s0076-5392(99)x8001-5.

[25] I. Petras, ‘‘The fractional-order controllers: Methods for their synthesis and application,’’ 2000, *arXiv:math/0004064*.

[26] A. Yuce, F. N. Deniz, N. Tan, and D. P. Atherton, ‘‘Obtaining the time response of control systems with fractional order PID from frequency responses,’’ in *Proc. 9th Int. Conf. Electr. Electron. Eng. (ELECO)*, Bursa, Turkey, Nov. 2015, pp. 832–836, doi: 10.1109/ELECO.2015.7394522.

[27] D. P. Atherton, N. Tan, and A. Yuce, ‘‘Methods for computing the time response of fractional-order systems,’’ *IET Control Theory Appl.*, vol. 9, no. 6, pp. 817–830, Apr. 2015, doi: 10.1049/iet-cta.2014.0354.

[28] F. N. Deniz, A. Yuce, N. Tan, and D. P. Atherton, ‘‘Tuning of fractional order PID controllers based on integral performance criteria using Fourier series method,’’ *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 8561–8566, Jul. 2017, doi: 10.1016/j.ifacol.2017.08.1417.

[29] K. J. Åström, H. Panagopoulos, and T. Hagglund, ‘‘Design of PI controllers based on non-convex optimization,’’ *Automatica*, vol. 34, no. 5, pp. 585–601, May 1998, doi: 10.1016/s0005-1098(98)00011-9.

[30] R. Anuja, T. S. Sivarani, and M. G. Nisha, ‘‘Fuzzy fractional order proportional integral derivative controller design for higherorder time delay processes,’’ *Int. J. Uncertainty, Fuzziness Knowl.-Based Syst.*, vol. 31, no. 02, pp. 327–349, Apr. 2023, doi: 10.1142/s0218488523500174.

[31] S. Shekhar and A. Kumar, ‘‘Fractional order interval type-2 fuzzy logic controller,’’ in *Studies in Fuzziness and Soft Computing*. Berlin, Germany: Springer, 2023, pp. 29–42, doi: 10.1007/978-3-031-26332-3_3.

[32] C. I. Muresan, I. Birs, C. Ionescu, E. H. Dulf, and R. De Keyser, ‘‘A review of recent developments in autotuning methods for fractional-order controllers,’’ *Fractal Fractional*, vol. 6, no. 1, p. 37, Jan. 2022, doi: 10.3390/fractalfract6010037.

[33] Y. Chen, T. Bhaskaran, and D. Xue, ‘‘Practical tuning rule development for fractional order proportional and integral controllers,’’ *J. Comput. Nonlinear Dyn.*, vol. 3, no. 2, pp. 1–8, Jan. 2008, doi: 10.1115/1.2833934.

[34] J. J. Gude and E. Kahoraho, ‘‘Modified Ziegler–Nichols method for fractional PI controllers,’’ in *Proc. IEEE 15th Conf. Emerging Technol. Factory Automat.*, Bilbao, Spain, Sep. 2010, pp. 1–5, doi: 10.1109/ETFA.2010.5641074.

[35] G. E. Santamaria, I. Tejado, B. M. Vinagre, and C. A. Monje, ‘‘Fully automated tuning and implementation of fractional PID controllers,’’ in *Proc. ASME Int. Design Eng. Tech. Conf. Comput. Inf. Eng. Conf.*, San Diego, CA, USA, Jan. 2009, pp. 1275–1283, doi: 10.1115/DETC2009-87399.

[36] M. Al-Dhaifallah, N. Kanagaraj, and K. S. Nisar, ‘‘Fuzzy fractional-order PID controller for fractional model of pneumatic pressure system,’’ *Math. Problems Eng.*, vol. 2018, pp. 1–9, Feb. 2018, doi: 10.1155/2018/5478781.

[37] L. Liu, D. Xue, and S. Zhang, ‘‘General type industrial temperature system control based on fuzzy fractional-order PID controller,’’ *Complex Intell. Syst.*, vol. 2021, pp. 1–13, Jun. 2021, doi: 10.1007/s40747-021-00431-9.

[38] M. Al-Dhaifallah, ‘‘Construction and evaluation of a control mechanism for fuzzy fractional-order PID,’’ *Appl. Sci.*, vol. 12, no. 14, p. 6832, Jul. 2022, doi: 10.3390/app12146832.

[39] E. Tepljakov and Y. J. P. Belikov, ‘‘Development of analytical tuning methods for fractional-order controllers,’’ in *Proc. 6th IKTDK Inf. Commun. Technol. Doctoral School Conf.*, 2012, pp. 93–96.

[40] C. I. Muresan and R. De Keyser, ‘‘Revisiting Ziegler–Nichols. A fractional order approach,’’ *ISA Trans.*, vol. 129, pp. 287–296, Oct. 2022, doi: 10.1016/j.isatra.2022.01.017.

[41] R. De Keyser, C. I. Muresan, and C. M. Ionescu, ‘‘Autotuning of a robust fractional order PID controller,’’ *IFAC-PapersOnLine*, vol. 51, no. 25, pp. 466–471, 2018, doi: 10.1016/j.ifacol.2018.11.181.

[42] D. Valério and J. S. da Costa, ‘‘Tuning-rules for fractional PID controllers,’’ *IFAC Proc. Volumes*, vol. 39, no. 11, pp. 28–33, Jan. 2006, doi: 10.3182/20060719-3-pt-4902.00004.

[43] A. Odienat, M. M. Al Momani, K. Alawasa, and S. F. Gharaibeh, ‘‘Low frequency oscillation analysis for dynamic performance of power systems,’’ in *Proc. 12th Int. Renew. Eng. Conf. (IREC)*, Amman, Jordan, Apr. 2021, pp. 1–6, doi: 10.1109/IREC51415.2021.9427818.

[44] M. M. Almomani, A. Odienat, S. F. Al-Gharaibeh, and K. Alawasa, ‘‘The impact of wind generation on low frequency oscillation in power systems,’’ in *Proc. IEEE PES/IAS PowerAfrica*, Nairobi, Kenya, Aug. 2021, pp. 1–5, doi: 10.1109/PowerAfrica52236.2021.9543283.

MOHAMMAD M. AL-MOMANI received the B.Sc. degree (Hons.) in electrical power system and machinery from Yarmouk University, Irbid, Jordan, in 2017, and the M.Sc. degree (Hons.) in smart grids in power systems from Mutah University, Karak, Jordan, in 2021. He currently holds the position of a Protection Engineer within the Protection and Metering Section, National Electrical Power Company, Amman, Jordan. Currently, he is actively seeking an opportunity to pursue his Ph.D. study at a prestigious university. His research interests include the monitoring and control of smart grids and power system dynamics and stability, with an emphasis on the context of low-inertia power systems and microgrid stability.

AMNEH AL-MBAIDEEN received the Ph.D. degree from The University of Sheffield, U.K., in 2011. She is currently an Associate Professor of electrical power engineering with Mutah University, Karak, Jordan.

ABDULLAH I. AL-ODIENAT received the Ph.D. degree from Tashkent State University, Tashkent, in 1998. He is currently a Full Professor of electrical power engineering with Mutah University, Karak, Jordan, where he is currently the Dean of scientific research. His research interests include power system stability, smart grids, impacts of the integration of renewable resources with the power grid, and wide area monitoring, protection, and control of power systems.

KHALED MOHAMMAD ALAWASA (Member, IEEE) received the B.Sc. degree in electrical engineering from Mu'tah University, Al-Karak, Jordan, in 2000, the M.Sc. degree in power engineering with business from Strathclyde University, Glasgow, U.K., in 2008, and the Ph.D. degree in electrical engineering from the University of Alberta, Edmonton, AB, Canada, in 2014. He worked a protection and control engineer with the National Electric Power Company (NEPCO), Amman, Jordan, from 2001 to 2007, and a power system specialist with Magna IV, Edmonton, Canada, from 2012 to 2013. In 2014, he jointed Mutah university as an Assistant Professor, since 2019, he has been prompted to an Associate Professor at Mutah University. Since Aug 2021. He has been joined the Department of Electrical and Computer Engineering at Sultan Qaboos university. His main research interests are renewable energy integration, energy storage integration, power electronics, real time digital simulations (RTDs) and hardware-in-the-loop (HIL), smart grids PMUS, and WAMs, power system stability and control, power quality and power disturbance analytics, power System protection and fault analysis, Energy management and Energy Efficiency.

SABA F. AL-GHARAIBEH received the B.Sc. degree in electrical power system and machinery from Yarmouk University, Irbid, Jordan, in 2017, and the M.Sc. degree in smart grids in power systems from Mutah University, Karak, Jordan, in 2021. Her research interests include the monitoring and control of smart grids and the applications of artificial intelligence in power systems. Currently, she is actively seeking a favorable opportunity to pursue her Ph.D. study at a reputable university.