Data-Driven Interval Type-2 Fuzzy Inference System Based on the Interval Type-2 Distending Function

József Dombi and Abrar Hussain

***Abstract*—Fuzzy type-2 modeling techniques are increasingly being used to model uncertain dynamical systems. However, some challenges arise when applying the existing techniques. These are the following: 1) a large number of rules are required to complete cover the whole input space; 2) a large number of parameters associated with type-2 membership functions have to be determined; 3) the identified fuzzy model is usually difficult to interpret due to the large number of rules; and 4) designing a fuzzy type-2 controller using these models is a computationally expensive task. To overcome these limitations, a procedure is proposed here to identify the fuzzy type-2 model directly from the data. This model is called the distending-function-based fuzzy inference system (DFIS). This model consists of rules and interval type-2 distending functions. First, a few key rules are identified from the data, and later, more rules are added until the error is less than the threshold. The proposed procedure is used to model the altitude controller of a quadcopter. The performance of the DFIS model is compared with that of various fuzzy models. Furthermore, a simplified procedure based on the rules is presented to design a computationally low-cost type-2 controller. The effectiveness of the controller is shown by regulating the height of a quadcopter in the presence of noisy sensory data. The performance of this controller is compared with that of various other controllers. Finally, the proposed type-2 controller was implemented on a Parrot Mambo quadcopter to demonstrate its real-time performance.**

***Index Terms*—Arithmetic-based control, fuzzy type-2 modeling, Parrot mini-drone Mambo, type-2 distending function (T2DF).**

I. Introduction

FUZZY theory has found numerous practical applications in the fields of engineering, operational research, and statistics [1], [2], [3]. The fuzzy inference engine consists of a fuzzifier, a rule base system, fuzzy operators, and defuzzification. These rules describe the dependencies between the input and output variables in the form of IF–THEN statements. In this article, we considered a special case where the expert knowledge in the form of linguistic rules is not available or it is poorly described. For example:

1) if the system under consideration is so complex that it cannot be described using linguistic fuzzy rules;
2) experts who can describe the system using the linguistic fuzzy rules are not available.

In this case, data-driven fuzzy modeling and control is the only feasible option [4], [5], [6]. There are two main types of fuzzy inference systems: Mamdani systems [7] and Sugeno systems [8]. Fuzzy modeling involves the identification of fuzzy rules and parameter values from the data for either of these two types of inference systems. The data-driven identification of a fuzzy model can be divided into two parts, namely, qualitative and quantitative identification. Qualitative identification focuses on the number and description of fuzzy rules, while quantitative identification is concerned with the identification of parameter values. These parameters belong to membership functions and fuzzy operators. In one of the latest papers, Duţu et al. [9] investigated the qualitative identification in detail for Mamdani-like fuzzy systems. A parameterized rule learning technique called selection reduction was introduced. The number of rules was optimized by dropping some rules based on the rule redundancy index. This technique is called precise and fast fuzzy modeling approach. Ying and Lin [10] presented the first ever self-learning fuzzy model for discrete event systems. Stochastic gradient descent optimization was used to learn the event transition matrix and the parameters of multidimensional Gaussian fuzzy sets. The theoretical developments were supported by MATLAB simulations for learning the automaton's parameter values. Quite recently, Mendel et al. [11] determined the relationship between explainable artificial intelligence (AI) and fuzzy rule-based systems. It was established that the shape of antecedent membership function, linguistic approximations, and similarity are essential for the explainable AI.

In Sugeno-type systems, soft computing methodologies like evolutionary algorithms, genetic algorithms (GAs), and particle swarm optimization are used for qualitative identification [12], [13], [14], [15]. Neural networks are mostly used in quantitative identification. These lead to the development of adaptive neuro-fuzzy inference systems (ANFISs) [16]. ANFIS is the most popular and widely used technique for fuzzy modeling. Different variants of the ANFIS model have been developed using various types of optimization techniques [17], [18], [19]. Harifi et al. [20] proposed a new type of ANFIS model by using a novel emperor penguin colony (EPC) optimization method. EPC is a recently introduced, nature-inspired, and population-based metaheuristic

Manuscript received 6 April 2022; revised 23 July 2022 and 17 October 2022; accepted 15 November 2022. Date of publication 25 November 2022; date of current version 30 June 2023. This work was supported in part by the Ministry of Innovation and Technology of Hungary through the National Research, Development and Innovation Fund under Project TKP2021-NVA-09, and in part by the European Union under Project RRF-2.3.1-21-2022-00004 within the framework of the Artificial Intelligence National Laboratory. *(Corresponding author: Abrar Hussain.)*

The authors are with the Department of Computer Algorithms and Artificial Intelligence, Institute of Informatics, University of Szeged, 6720 Szeged, Hungary (e-mail: dombi@inf.u-szeged.hu; hussain@inf.u-szeged.hu).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TFUZZ.2022.3224793.

Digital Object Identifier 10.1109/TFUZZ.2022.3224793

algorithm. It was shown that the ANFIS model based on EPC can solve various prediction and classification problems with high accuracy compared with the existing ANFIS-type models. However, all these abovementioned methods have some limitations related to the qualitative identification, quantitative identification, and rule interpretability.

Type-2 fuzzy systems (T2FSs) were developed to handle the uncertainty in type-1 fuzzy sets [21], [22]. There are always uncertainties associated with the real-world physical systems. The source of these uncertainties can be the variations in the parameters with time or external noise/disturbances acting on the system. As the membership value of the type-1 system is crisp, these uncertainties cannot be modeled by the type-1 fuzzy models. Therefore, the T2FSs are employed in practice to accurately model uncertainties. However, the computational complexity associated with the quantitative and qualitative identification steps of the data-driven type-2 fuzzy modeling techniques is very high. T2FSs are based on type-2 membership functions (T2MF). The T2MF contains the footprint of uncertainty (FOU) between the upper membership function (UMF) and the lower membership function (LMF). Interval type-2 fuzzy systems (IT2FSs) have been developed to reduce the computational complexity [23]. The T2FS has superior properties such as: 1) better handling of uncertainties [24]; 2) smooth controller response [25]; 3) adaptivity [25]; and 4) reduction in the number of fuzzy rules [26]. Abiyev and Kaynak [27] proposed a novel type-2 Takagi–Sugeno–Kang fuzzy neural system to control the time-varying processes. The parameters of the rules were updated using the fuzzy clustering and gradient-based algorithm. T2FSs have been successfully used in various control system applications [28], [29], data mining [30], and time-series predictions [31]. Naderipour et al. [32] used the T2FS for community detection in large social networks. The methodology was based on possibilistic c-means clustering with two-layer graphs. Cuevas et al. [33] developed a type-2 control system for controlling an omnidirectional autonomous robot in unstructured environment with unknown obstacles. The recent trends in theoretical and practical implementation of T2FS are summarized in [34]. The design of the IT2FS consists of the following:

1) fuzzification of the inputs using the T2MF;
2) calculation of the rule firing strengths;
3) implication and aggregation to produce rule outputs. These operations produce a type-2 fuzzy set;
4) type reduction to convert type-2 fuzzy sets into type-1 fuzzy sets;
5) defuzzification to get a final crisp output value.

The type-reduction step is performed using the so-called Karnik–Mendel (KM) iterative algorithm [35]. This algorithm defines two switching points for the lower and upper firing strengths. Using these points, the algorithm generates two type-1 fuzzy sets. These sets are then defuzzified to get a crisp output. This approach has some drawbacks, such as: 1) the choice of T2MFs; 2) the computational complexity of the type-reduction step; 3) difficulties in the optimization process; and 4) controller design complexity. Quite recently, several techniques have been proposed to tackle these problem [36], [37], [38]. Mendel and Bonissone [11] proposed an efficient methodology for optimal type reduction under stability considerations for fuzzy-model-based control techniques. The methodology was membership function dependent and was based on deep reinforcement learning. Le [39] presented a self-evolving functional-link type-2 fuzzy neural network (SEFIT2FNN). It uses the particle swarm optimization method to adjust the learning rate of the adaptive law. The adaptive law tunes the parameters of the type-2 fuzzy neural network. The SEFIT2FNN has been shown to successfully control the antilock braking system under various road conditions. Mohammadzadeh et al. [40] moved a step further by proposing an interval type-3 fuzzy system (IT3FS). The membership values in the IT3FS are not crisp values, but those in IT2FS are crisp. An online fractional-order learning system has been presented to tune the consequent parameters. It was shown using the simulation studies that the accuracy of the proposed IT3FS is much higher than the IT2FS and the type-1 systems. The results were compared with those of other recently published techniques, including SEFIT2FNN. However, the computational complexity of this technique is high, and the antecedent parameters are not tunable. In another recent paper, Wei et al. [41] developed a self-organizing interval type-2 fuzzy controller to regulate the bispectral index during general anesthesia. The input–output data scaling factor and the FOU were optimized using a joint surrogate model and genetic programming. The performance of the model is generally better than that of the GA-based type-1 and type-2 fuzzy models. The model also perfectly handled noise and interpatient variability. Pratama et al. [42] proposed an evolving type-2 classifier, where fuzzy rules can be pruned, merged, and grow automatically. The antecedent part comprised of a Gaussian function and the consequent part consisted of a Chebyshev polynomial. Quite recently, Le et al. [43] have developed a multilayer interval type-2 fuzzy controller (MIT2FC), where the network parameters are learnt using gradient techniques. The learning rates are optimized using the Jaya algorithm. The proposed methodology was shown to greatly reduce the number of rules. Numerical experiments were conducted to control the trajectory of a quadcopter in the presence of external white noise. However, these existing approaches have some limitations. These limitations are briefly described as follows.

A. Limitations in the Existing Fuzzy Type-2 Modeling Techniques

1) *Qualitative identification:* Qualitative identification suffers from the so-called flat structure (curse of dimensionality) problem of the rule base [30], i.e., if the number of input variables increases, then an exponentially large number of rules are required to accurately model the system. This is due to the fact that the support area of the most frequently used membership function (triangular, trapezoidal) covers a limited area of the input space. To cover the input space completely, a huge number of rules are required. If we have two input variables, each with six categories, then the number of rules required to cover the whole input space will be 36. Each rule is applicable only

within a specific area, and its strength is zero outside. If the training data of the system do not fully span the input and output spaces, then this will cause serious problems when modeling the system. If the input falls in these uncovered areas, then the identified rule base does not generate any action. Even if some sort of interpolation technique is applied, the computational complexity will increase [44]. Therefore, a global fuzzy model requires a large number of rules and the number of rules depends exponentially on the number of input variables, and this will lead to a huge complexity in data-driven fuzzy models.

2) *Quantitative identification:* The computational complexity of the quantitative part of the identified fuzzy model also increases with the number of rules. As the number of rules increases, the number of parameters of the T2MF and operators also grows exponentially. Computing these parameters will then increase the computational cost of the quantitative model.
3) *Choice of the T2MF:* The choice of the T2MF and its systematic connection with the type of uncertainty are not clear. Different type-1 membership functions can be combined to generate T2MFs. However, it is not clear which type of membership functions should be used for a particular type of uncertainty.
4) *Interpretability:* In most cases, the interpretability of the identified fuzzy rule base is not clear. It is easier to interpret a few rules and get an insight into the working model. However, if the number of rules grows exponentially, then for a given set of input values, it is not possible to predict the response of the model and analyze its performance. The model tends to behave more like a black box in these situations.
5) *Optimization process:* Although type-2 fuzzy logic systems require fewer rules compared with type-1 fuzzy systems, the number of parameters is comparatively large. Therefore, optimizing a large number of parameter values is not an easy task.

B. Limitations in the Existing Fuzzy Type-2 Control Techniques

These limitations are related to the computational complexity of the fuzzy type-2 controllers. Using the rule base, an interval type-2 fuzzy controller can be designed using well-established techniques [45]. Most of these techniques use the type-reduction step. The type-reduction step is based on the KM algorithm, which is computationally expensive. Owing to its iterative nature, it is ill-suited for online applications. There are some alternative solutions that reduce the computational burden, but these are approximations [46]. These techniques also include the implication and aggregation steps. These steps further add to the computational complexity of the type-2 fuzzy controller.

C. Unique Features of the Proposed Approach

Here, we propose solutions to remove some of these limitations. We present a novel technique for fuzzy type-2 modeling and control. The proposed fuzzy model (distending-function-based fuzzy inference system—DFIS) consists of rules and T2MFs. The rules are based on the Dombi conjunctive operator. A procedure for designing a fuzzy type-2 controller using the rules is also presented. The controller can handle various types of uncertainties (e.g., sensor noise). The proposed methodology has the following unique contributions.

1) The complexity associated with the qualitative identification of the data-driven techniques is reduced. This is achieved by using a unique interval T2MF called the interval type-2 distending function (T2DF) [47]. It has symmetric and asymmetric forms. It can completely cover the whole input space with a few rules, and this helps overcome the flat structure issue.
2) The computational burden of the quantitative identification is greatly reduced. The interval T2DF has only a few parameters. Most of these parameters are kept fixed and a few are varied during the training process.
3) The proposed methodology could be used to model the real-world physical systems. Different types of uncertainties can be modeled using the parameters of the interval T2DF. Therefore, most forms of the uncertainties in fuzzy systems can be represented using the interval T2DF.
4) The methodology identifies a model with high interpretability. Our approach identifies a few important fuzzy rules. We have also developed a rule reduction algorithm, which can further reduce the number of rules, and it results in an interpretable model.
5) Because only a few parameters are varied during the design process, the optimization is simple and fast.
6) We presented an arithmetic-based interval type-2 fuzzy controller. The type-reduction, implication, and aggregation steps are not involved in the design procedure. Therefore, the controller is computationally efficient.

The rest of this article is organized as follows. In Section II, we briefly introduce the interval T2DF and its properties. In Section III, we explain the proposed controller design approach and rule reduction algorithm. In Section IV, we describe the benchmark system, simulation results, and hardware implementation and discuss the results. Finally, Section V concludes this article.

II. Interval T2DF

Zadeh [48] proposed various membership functions, and one of them has the following form:

$$\mu(x) = \frac{1}{1 + \left(\frac{x-a}{b}\right)^2}. \tag{1}$$

Based on $\mu(x)$, we defined a more general parametric function, which models a soft equality, and it is called the distending function (DF). This type of membership function is closely related to the operator systems, and in our case (i.e., the DF), it is associated with the Dombi operators. The DF can be derived from the Kappa function of the Dombi operator [49]. The DF has four parameters, namely, ν, ε, λ, and c.

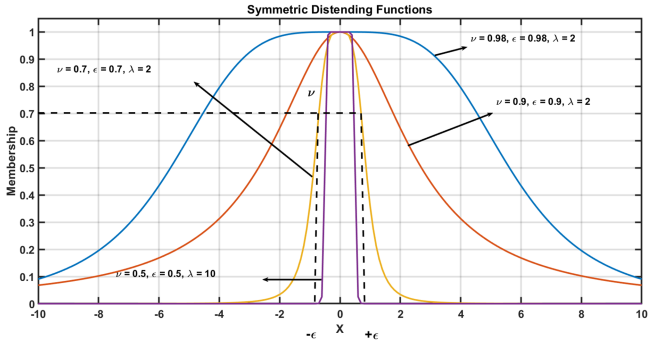

Fig. 1. Various shapes of symmetric DFs (here $c = 0$).

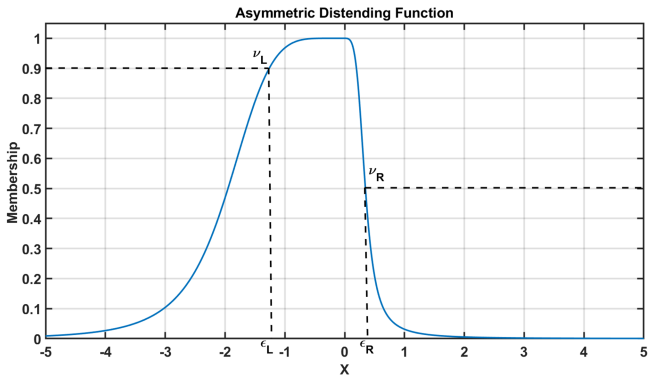

Fig. 2. Asymmetric DF ($\nu_L = 0.5, \varepsilon_L = 0.5, \lambda_L = 5, \nu_R = 0.8, \varepsilon_R = 0.7, \lambda_R = 5, \lambda = 5, c = 0$).

It has two forms: 1) symmetric and 2) asymmetric. The symmetric DF (shown in Fig. 1) is symmetric around $x - c$, and it is defined as [50]

$$\delta^{(\lambda)}_{\varepsilon,\nu}(x-c) = \frac{1}{1+\frac{1-\nu}{\nu}\left|\frac{x-c}{\varepsilon}\right|^{\lambda}} \tag{2}$$

where $\nu \in (0,1)$, $\varepsilon > 0$, $\lambda \in (1,+\infty)$, and $c \in \mathbb{R}$. $\delta^{(\lambda)}_{\varepsilon,\nu}(x-c)$ is denoted by $\delta_s(x)$.

The asymmetric type of the DF (shown in Fig. 2) is given by

$$\delta_A(x-c) = \frac{1}{1+\frac{1-\nu_R}{\nu_R}\left|\frac{x-c}{\varepsilon_R}\right|^{\lambda_R}\frac{1}{1+e^{-\lambda^*(x-c)}}+\frac{1-\nu_L}{\nu_L}\left|\frac{x-c}{\varepsilon_L}\right|^{\lambda_L}\frac{1}{1+e^{\lambda^*(x-c)}}} \tag{3}$$

where $\nu_R, \nu_L \in (0,1)$, $\varepsilon_R, \varepsilon_L > 0$, $\lambda_L, \lambda_R \in (1,+\infty)$, $c \in \mathbb{R}$, $\lambda^* \in (1,+\infty)$, and $\lambda^* >> \lambda_L, \lambda_R$. Here, c is the centre point, i.e., $\delta_A(c) = 1$.

A. Construction of the Interval T2DF

The values of the DF parameters ($\nu, \varepsilon, \lambda$, and c) may be uncertain. As a result, these parameters can take various values around their nominal values, within the uncertainty bound (Δ). By varying the parameter values within Δ, various DFs are obtained. The DF with the highest grade values is called the UMF and that with the lowest values is called the LMF. The

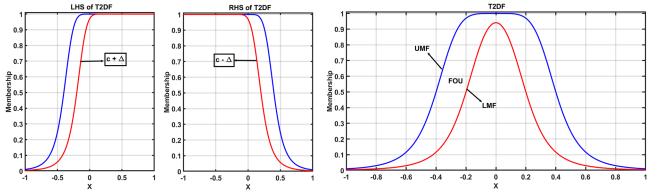

Fig. 3. Uncertain peak value interval T2DF with the FOU.

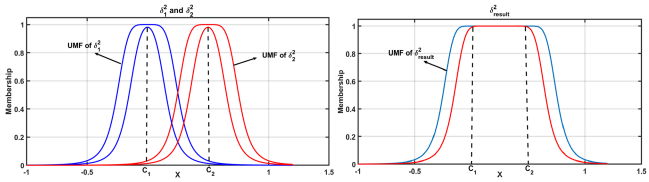

Fig. 4. Combining two interval T2DFs (δ_1^2 and δ_2^2) to get a single interval T2DF (δ^2_{result}).

UMF, LMF, and various DFs in between can be combined to form an interval T2DF [47]. If the peak value of the DF becomes uncertain, then it can be represented using the interval T2DF with an uncertain "c" value, as shown in Fig. 3.

Various interval T2DFs belonging to the same fuzzy variable can be combined to form a single interval T2DF. The support of the resultant interval T2DF will be approximately the same as the combined support of the individual interval T2DFs. The UMF of the interval T2DF consists of the left-hand side (LHS) and right-hand side (RHS) (the same is true for the LMF). The LHS and RHS are given by Dombi and Hussain [47]:

$$\bar{\delta}^2_L(x-c) = \frac{1}{1+\frac{1-\nu}{\nu}\left|\frac{x-c}{\varepsilon}\right|^{\lambda}\frac{1}{1+e^{(\lambda^*(x-c))}}} \tag{4}$$

$$\bar{\delta}^2_R(x-c) = \frac{1}{1+\frac{1-\nu}{\nu}\left|\frac{x-c}{\varepsilon}\right|^{\lambda}\frac{1}{1+e^{(-\lambda^*(x-c))}}}. \tag{5}$$

The LHS and RHS of the UMF and LMF can be combined using the Dombi conjunctive operator to get a single interval T2DF. Consider two interval T2DFs δ_1^2 and δ_2^2. The LHS of δ_1^2 and RHS of δ_2^2 can be combined using the Dombi conjunctive operator [51]. This produces a resultant interval T2DF δ^2_{result}, as shown in Fig. 4. Combining various interval T2DFs helps to reduce the number of fuzzy rules. This leads to a decrease in the computational complexity of the identified fuzzy model.

B. Interval T2DF in a Higher Dimension

Consider n different interval T2DFs in n different dimensions given by $\delta^{2(\lambda_1)}_{1(\epsilon_1,\nu_1)}(x_1-c_1)$, $\delta^{2(\lambda_2)}_{2(\epsilon_2,\nu_2)}(x_2-c_2)\ldots,\delta^{2(\lambda_n)}_{n(\epsilon_n,\nu_n)}(x_n-c_n)$. If we apply the Dombi conjunctive operator on these n interval T2DFs, then the result will also be an interval T2DF $\delta^2(x_1,x_2,\ldots,x_n)$ in n dimensions (shown in Fig. 5). And

$$\bar{\delta}^2(x_1,x_2,\ldots,x_n) = \frac{1}{1+\sum_{i=1}^{n}\frac{1-\nu_i}{\nu_i}\left|\frac{x_i-c_i}{\epsilon_i}\right|^{\lambda_i}} \tag{6}$$

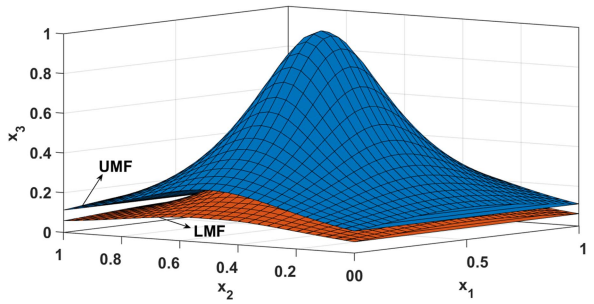

Fig. 5. Interval T2DF in the x_3 dimension.

$$\underline{\delta}^2(x_1, x_2, \ldots, x_n) = \frac{1}{1 + \sum_{i=1}^{n} \frac{1-\nu_i}{\nu_i} \left(\left| \frac{x_i - (c_i - \Delta)}{\epsilon_i} \right|^{\lambda_i} \sigma_{Li} + \left| \frac{x_i - (c_i + \Delta)}{\epsilon_i} \right|^{\lambda_i} \sigma_{Ri} \right)} \tag{7}$$

$$\text{where } \sigma_{Li} = \frac{1}{1 + e^{-\lambda_i (x_i - (c_i - \Delta))}} \tag{8}$$

$$\text{and} \quad \sigma_{Ri} = \frac{1}{1 + e^{\lambda_i (x_i - (c_i + \Delta))}}. \tag{9}$$

Here, Δ is the upper bound on the uncertainty in the c value, $\bar{\delta}^2(x_1, x_2, \ldots, x_n)$ is the UMF, and $\underline{\delta}^2(x_1, x_2, \ldots, x_n)$ is the LMF of $\delta^2(x_1, x_2, \ldots, x_n)$. (See the supplemental material for the proof.) The proposed design approach is explained in the next section.

III. Data-Driven Fuzzy Modeling

Here, we assume that expert knowledge in the form of linguistic rules is not available. However, the batch offline (input and output) data of a healthy process are available. Using these data, we will drive a DFIS model composed of rules and interval T2DFs. A rule base for a multi-input multi-output (MIMO) system can be written as

$$\begin{aligned} &\text{if } x_1 \text{ is } U_1^i \text{ and } \ldots\ldots \text{ and } x_n \text{ is } U_n^i \\ &\text{then } y_1 \text{ is } V_1^i;\ \ldots\ldots;\ y_m \text{ is } V_m^i, \end{aligned} \tag{10}$$

where $x_1, x_2, \ldots, x_n$ are the input variables and $y_1, y_2, \ldots, y_m$ are the m output variables, and the corresponding fuzzy subsets are $U_1, U_2, \ldots, U_n$ and $V_1, V_2, \ldots, V_m$, respectively. The index i represents the rule number, and there are l fuzzy rules. A MIMO system given by (10) with m independent outputs can always be replaced by m multi-input single-output systems of the form

$$\text{if } x_1 \text{ is } U_1^i \text{ and } \ldots\ldots \text{ and } x_n \text{ is } U_n^i \text{ then } y_s \text{ is } V_s^i \tag{11}$$

where $s = 1, \ldots, m$ are the m outputs. For simplicity, we will consider the case where $s = 1$, and we propose a methodology for generating a crisp control signal using the input and output training data. The methodology can be generalized to m independent outputs.

Let us now assume that the input and output databases have the following form:

$$U = \begin{bmatrix} a_1^1 & a_2^1 & \ldots & a_n^1 \\ a_1^2 & a_2^2 & \ldots & a_n^2 \\ \vdots & \vdots & \vdots & \vdots \\ a_1^l & a_2^l & \ldots & a_n^l \end{bmatrix}, \quad V = \begin{bmatrix} b^1 \\ b^2 \\ \vdots \\ b^l \end{bmatrix} \tag{12}$$

where U and V contains the l data points of each input and output variable. Here, $a_1, a_2, \ldots, a_n$ are the data points belonging to the input fuzzy subsets $U_1, U_2, \ldots, U_n$, respectively, and b_1 is included in the output fuzzy subset V. Each column of the U matrix corresponds to a unique feature (input variables) of the process. Therefore, the U matrix forms an n-dimensional input feature space. Each column of the training matrix U is normalized by transforming it to the $[0, 1]$ interval. As a result, the feature values are comparable on the same scale.

The fuzzy rule consists of an antecedent and a consequent part. Here, the antecedent part contains a row of U, and the consequent part is an element of V. The antecedent part of the ith fuzzy rule is given by the following relation:

$$\mathcal{L}\left(\delta_1^2(x_1)^i, \delta_2^2(x_2)^i, \ldots, \delta_n^2(x_n)^i\right) \tag{13}$$

where $\mathcal{L}$ is the fuzzy logical expression, and it is evaluated using a very general class of the fuzzy operators [51]

$$D_\gamma(x) = \frac{1}{1 + \left(\frac{1}{\gamma} \left(\prod_{i=1}^{n} \left(1 + \gamma \left(\frac{1 - \delta^2(x_i)}{\delta^2(x_i)} \right)^{\alpha} \right) - 1 \right) \right)^{\frac{1}{\alpha}}}. \tag{14}$$

The consequent part of the ith rule (i.e., y is V^i) depends on the value in the ith row of V. As the ith row of V has a single value (i.e., b_i), b_i is directly taken as the consequent value of the ith rule. The b_i value is not optimized during the training procedure. This is only picked from the V matrix. It is worth mentioning that the row indices of U and V are same for the ith rule. In our approach, the fuzzy rules will be based on sample values in the U and V matrices. Therefore, a few rows from the database matrix U are selected. These can be selected randomly, but from a practical point of view, it is beneficial to choose those rows that contain the extremum (around 0 and 1) and average (around 0.5) values of the input variables. These rows and the corresponding elements in the V matrix are used to construct the rule base. It is called the boundary value rule base (R_b) because it mostly contains those values of the inputs that lie on the boundary of the input space. Each variable in our approach is normalized in $[0, 1]$ interval. To cover the boundary and middle values for each variable, there should be rules covering 0, 0.5, and 1 values. Hence, for a single variable, the maximum initial number of rules in R_b is 3. In the same way, in the case of two variables, nine rules are required to cover the boundary and middle values of the surface (formed with two input variables). Therefore, in a general case, if there are n input variables, then the maximum initial number of rule in R_b is 3^n. Usually, in the real-world databases, rows with boundary and middle values of all the variables are not available. Therefore, the initial number of rules in R_b is usually less than 3^n.

In our procedure, two different surfaces are constructed. These are called the estimated and the fuzzy surfaces. The estimated surface is constructed directly from the database [see (12)]. For missing data values, linear interpolation is used. The estimated surface is denoted by G. The fuzzy surface is generated from R_b, and it is denoted by G^*. These two surfaces are then used to create a third surface called the error surface E. Next, we will describe the procedure used to construct the surfaces G^* and E using R_b.

A. Construction of Fuzzy Surface G^* and Error Surface E

Each selected row from the database matrix U corresponds to a single rule. It is a row vector, and it consists of unique values of all the input variables (features). A unique interval T2DF is constructed for each input variable in the row. The interval T2DF has four parameters ($\nu, \varepsilon, \lambda$, and c). Apart from these four parameters, an additional parameter Δ is required for the construction of interval T2DF [see (7)]. Δ is the upper bound on the uncertainty in the peak value coordinate c of the interval T2DF. The optimum value of each of these parameters can be determined.

The parameter c (peak value coordinate) of the interval T2DF is given, and it is equal to the value of the corresponding input variable. The value of λ can be chosen between 1 and ∞. Based on the experimental observations, the value of $\lambda = 2$ is good for most of the practical applications. The value of ε depends upon the number of rules in R_b

$$\varepsilon = \frac{1}{\text{number of rules in } R_b}. \tag{15}$$

This ensures that the whole input space is covered. The input variables are usually measured using the feedback sensors. The Δ value of each sensor depends on the tolerance intervals of the corresponding sensor. All the Δ values are transformed into the $[0, 1]$ interval to make these compatible with the values of the input variables. Interval T2DFs have a long tail. Consequently, each interval T2DF influences the other existing interval T2DFs. The ν value of each interval T2DFs will be calculated based on the principle of minimum influence on all the other interval T2DFs. This influence can never be zero, but it can be decreased by a factor k. For less influence, a large value of k should be chosen. However, from a practical point of view, a value of 10 is sufficient. It means that the influence will decrease tenfold. The influence of the ith interval T2DF at the peak value of the jth interval T2DF is given by

$$\frac{1}{1 + \frac{1-\nu}{\nu}\left(\left|\frac{x_{i1}-x_{j1}}{\epsilon}\right|^{\lambda} + \cdots + \left|\frac{x_{in}-x_{jn}}{\epsilon}\right|^{\lambda}\right)} = \frac{1}{k} \tag{16}$$

where $x_{i1}, \ldots, x_{in}$ are the n coordinates of the peak value of the ith interval T2DF and $x_{j1}, \ldots, x_{jn}$ are the coordinates of the peak value of the jth interval T2DF. This formula can be used for one-dimensional interval T2DFs as well as n-dimensional interval T2DFs. Then, the required value of ν can be calculated using

$$\nu = \frac{1}{1 + \frac{k-1}{d}} \tag{17}$$

$$\text{where} \quad d = \left(\left|\frac{x_{i1} - x_{j1}}{\epsilon}\right|^{\lambda} + \cdots + \left|\frac{x_{in} - x_{jn}}{\epsilon}\right|^{\lambda}\right). \tag{18}$$

If $\varepsilon = 1$ and $\lambda = 2$, then d is a distance measure

In the antecedent part of the ith rule, there are n interval T2DFs corresponding to n input variables. Each rule is evaluated using the Dombi conjunctive/disjunctive operator. By applying the Dombi conjunctive/disjunctive operator over the n input interval T2DFs, we get a single interval T2DF. This is called the output interval T2DF. The UMF and LMF of this output interval T2DF are given by (6) and (7), respectively. Here, l output interval T2DFs will be generated from the l rules. All these output interval T2DFs are superimposed in the input space to generate a fuzzy surface G^*.

An error surface E is defined as the difference between the estimated surface G and the fuzzy surface G^*. That is

$$E(x_1, \ldots, x_n) = G(x_1, \ldots, x_n) - G^*(x_1, \ldots, x_n). \tag{19}$$

B. Extending the Rule Base

We shall decrease the magnitude of E, and this is achieved by an iterative procedure of adding new rules to R_b. To add a new fuzzy rule, the coordinates of the maximum value on E are located. The corresponding row in the database containing these coordinates is selected. This row is then added to R_b as a new rule. This rule is evaluated to generate an output interval T2DF. The ν value of this output interval T2DF is then calculated using (17). This interval T2DF is superimposed in G^*. This will modify the surface G^* in such a way that the magnitude of the maximum error on the surface E at these coordinates will decrease. This process is repeated in an iterative manner until the error surface E is within an error threshold τ_E. It is the upper bound on the error surface E. For a very small value of τ_E, a large number of rules have to be extracted from the training data and vice versa. Therefore, a compromise has to be made between the value of the error threshold τ_E and the number of rules in R_b. The optimum value of the error threshold is called $\tau_{E_{\text{op}}}$, and it is application dependent. Based on our experimental observations, the $\tau_{E_{\text{op}}}$ value can be calculated using the heuristic

$$\tau_{E_{\text{op}}} = \frac{\text{Peak value of the output}}{15}. \tag{20}$$

It should be added that extracting the type-2 fuzzy model from the data is based on the DF. Therefore, we call this type-2 model the DFIS. The whole procedure for extracting the DFIS from the data is summarized in Algorithm 1.

If the number of rules in R_b is large, then some of the rules can be merged to reduce the computational complexity. This is achieved using a reduction procedure.

C. Reducing the Rule Base

Here, we describe a heuristic approach used to decrease the number of rules in R_b. Rule reduction will lead to a lower

computational cost and better interpretability. Various output interval T2DFs that are close to each other in the input space can be combined to get a single interval T2DF (as shown in Fig. 4). The procedure is explained as follows: One of the input variables is selected and we call it a principal feature. In a control system, the error measure of the control variable is usually chosen as the principle feature. To ensure that the rule reduction procedure does not change the fuzzy surface significantly, the input space is divided into two half-spaces. The space where the principle feature value is less than 0.5 lies in the first half, and the rest of the space lies in the second half. Within each half, the output interval T2DFs are segregated into different groups. If the Euclidean distance between the peak value coordinates of various output interval T2DFs is less than a predefined distance D, then these interval T2DFs are placed in the same group, where for each half

$$D = \frac{\text{Sum of Euclidean distances b/w peak value T2DFs}}{\text{Total number of T2DFs in the same half}}. \tag{21}$$

Each output interval T2DF is obtained by applying a unique rule in R_b. The output interval T2DFs in the same group are combined together to produce a single interval T2DF. Consequently, the rules associated with all these output interval T2DFs are eliminated and replaced by a single new rule. Therefore, the number of rules in R_b decreases. Now, it is called a reduced rule base R_r. Using R_r, a new fuzzy surface is constructed, and it is denoted by G_r^*. Then, a reduced error surface (E_r) is obtained using

$$E_r(x_1, \ldots, x_n) = G(x_1, \ldots, x_n) - G_r^*(x_1, \ldots, x_n). \tag{22}$$

This procedure is performed in an iterative way as long as $E_r(x_1, \ldots, x_n)$ is within a reduction threshold τ_R. It is the upper bound on the reduced error surface E_r. Its value is also application dependent. If the reduction threshold value τ_R is high, then a large number of rules can be eliminated to get a much simpler and interpretable model. However, the accuracy of such an identified fuzzy model decreases. Therefore, the τ_R value should be chosen based on a compromise between model accuracy and interpretability. A high τ_R value leads to better interpretability but less accuracy and vice versa.

For an accurate DFIS model, the magnitude of the error surface should be minimum. However, an error surface with magnitude very close to zero is not desirable as this will lead to poor generalization (overfitting) of the DFIS model. The value of the reduction threshold τ_R is application dependent, and its optimum value may be different for different applications. However, based on our experimental observations, the optimum value of the threshold ($\tau_{R_{\text{op}}}$) can be determined using the heuristic

$$\tau_{R_{\text{op}}} = \frac{\text{Peak value of the output}}{10}. \tag{23}$$

High value of τ_R results in less number of rules, more interpretable, but least accurate DFIS model. Low value of τ_R produces a large number of rules, and the resulting DFIS model is less interpretable and also has worst generalization capability.

The whole procedure for reducing the number of rules in the DFIS model is summarized in Algorithm 2.

D. Arithmetic-Based Fuzzy Type-2 Controller Design Using the DFIS Model

The extracted DFIS model (R_r plus interval T2DFs) can be used to design an arithmetic-based interval type-2 fuzzy controller. We briefly present the procedure here [50].

Each rule in R_r has two parts called the antecedent and the consequent. We evaluate these two parts separately to generate a crisp control signal. For a specific values of input variables, the antecedent part is evaluated [see (13)], and it results in an interval $[\underline{\hat{v}}_i(\underline{x}^*)\ \hat{\overline{v}}_i(\underline{x}^*)]$

$$\mathcal{L}\left(\bar{\delta}_1^2(x_1^*)^i, \bar{\delta}_2^2(x_2^*)^i, \ldots, \bar{\delta}_n^2(x_n^*)^i\right) = \hat{\overline{v}}_i(\underline{x}^*) \tag{24}$$

$$\mathcal{L}\left(\underline{\delta}_1^2(x_1^*)^i, \underline{\delta}_2^2(x_2^*)^i, \ldots, \underline{\delta}_n^2(x_n^*)^i\right) = \underline{\hat{v}}_i(\underline{x}^*). \tag{25}$$

Here, $\underline{\hat{v}}_i(\underline{x}^*)$ is the lower strength and $\hat{\overline{v}}_i$ is the upper strength of the ith rule. $\underline{\delta}_n^2(x_n)^i$ is the LMF and $\overline{\delta}_n^2(x_n)^i$ is the UMF of the nth interval T2DF. The rule strengths are normalized to get the lower and upper firing strengths ($\underline{v}_i(\underline{x}^*),\ \overline{v}_i(\underline{x}^*)$):

$$\underline{v}_i(\underline{x}^*) = \frac{\underline{\hat{v}}_i(\underline{x}^*)}{\sum_{i=1}^k \underline{\hat{v}}_i(\underline{x}^*)}, \qquad \overline{v}_i(\underline{x}^*) = \frac{\hat{\overline{v}}_i(\underline{x}^*)}{\sum_{i=1}^k \hat{\overline{v}}_i(\underline{x}^*)} \tag{26}$$

$$\text{where} \qquad \sum_{i=1}^k \underline{v}_i(\underline{x}^*) = 1, \qquad \sum_{i=1}^k \overline{v}_i(\underline{x}^*) = 1 \tag{27}$$

and k is the total number of rules in R_r.

The consequent part of each rule in R_r is a single numeric value in the data matrix V. Let $b^1, \ldots, b^k$ be the consequent values, $\overline{v}_1^*, \ldots, \overline{v}_k^*$ be the upper firing strengths, and $\underline{v}_1^*, \ldots, \underline{v}_k^*$ be the lower firing strengths of the k fuzzy rules in R_r. And $b^1, \ldots, b^k$ values are picked from the V matrix [see (12)]. Then, the crisp output control U_{crisp} is generated by

$$U_{\text{crisp}} = \frac{\underline{c}_a + \overline{c}_a}{2} \tag{28}$$

$$\text{where } \overline{c}_a = \sum_{i=1}^k \overline{v}_i(\overline{x}^*)b^i, \text{ and } \underline{c}_a = \sum_{i=1}^k \underline{v}_i(\underline{x}^*)b^i. \tag{29}$$

The whole procedure is summarized in Algorithm 3.

IV. Benchmark System, Simulations Results, Hardware Implementation, and Discussion

The effectiveness of the proposed technique is demonstrated by modeling the altitude control system of a quadcopter (Parrot mini-drone). In our simulations, the batch offline data are first generated by regulating the altitude of the quadcopter using a proportional–derivative (PD) controller. Using these training data, the proposed DFIS model is identified. This model consists of fuzzy rules and interval T2DFs. In addition, various other fuzzy (type-1 and type-2) models are also identified from the same data. The performance of the DFIS model is compared with other fuzzy models. Afterward, three fuzzy controllers are designed using these three fuzzy models. The performance of these controllers is compared in a situation where the altitude of the quadcopter is regulated in the presence of noisy sensor measurements. Later on, the designed type-2 controller is also

Algorithm 1: Algorithm for Extracting Type-2 Fuzzy Model (DFIS) From the Data.

1: **Step 1:** Obtain the input and output databases of the process [see (12)].
2: **Step 2:** Transform each column of U into the $[0, 1]$ interval.
3: **Step 3:** Construct the estimated surface G.
4: **Step 4:** Generate a boundary value rule base R_b from the input and output databases.
5: **Step 5:** Assign an interval T2DF to each input point in the antecedent part of the rule base R_b. Choose $k = 10$ and calculate ν using (16) and (17).
6: **Step 6:** Calculate the output interval T2DF for each rule using (6) and (7).
7: **Step 7:** Generate fuzzy surfaces G^* from all the output interval T2DFs.
8: **Step 8:** Construct the error surface E using (19). If E is within the error threshold τ_E, then stop.
9: **Step 9:** Find the maximum value coordinates on E. Add a new rule in R_b corresponding to these coordinates in the databases.
10: **Step 9:** If the total number of rules in R_b are less than 7^n, then go to Step 6; else stop.

Algorithm 2: Algorithm for Reducing the Number of Rules in DFIS.

1: **Step 1:** Divide the input space into two parts based on the values of the principle feature.
2: **Step 2:** Segregate the output interval T2DFs in various groups based on the Euclidean distance D [see (21)].
3: **Step 3:** Combine the interval T2DFs in each group to generate a single interval T2DF.
4: **Step 4:** Replace the rules associated with the combined interval T2DFs with a single rule.
5: **Step 5:** Construct the reduced error surface E_r using (22).
6: **Step 6:** If E_r is within the reduction threshold τ_R, then go to step 1 else stop.

Algorithm 3: Algorithm for Designing the Arithmetic-Based Controller Using the DFIS Model.

1: **Step 1:** Calculate the upper and lower firing strengths of each rule using (26).
2: **Step 2:** Calculate $\overline{c}_a$ and $\underline{c}_a$ using the firing strengths and consequent values.
3: **Step 3:** Generate the crisp control signal using (28).

deployed on the mini-drone hardware to check the real-time performance.

A. Quadcopter Model

A Parrot mini-drone Mambo was used in this study (shown in Fig. 6). The MATLAB Simulink Aerospace block set provides the simulation model of this quadcopter [52]. The simulation

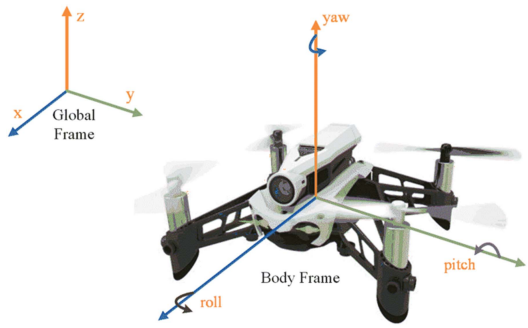

Fig. 6. Parrot mini-drone Mambo [53].

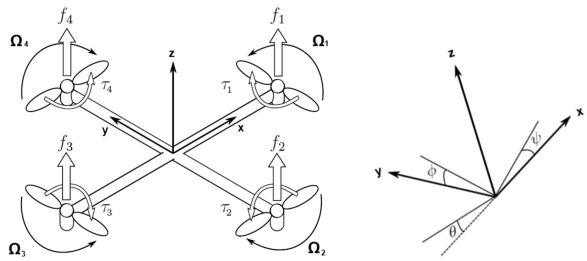

Fig. 7. Airframe model of the quadcopter structure [54].

consists of the airframe model, sensor model, environment model, and flight controller. The airframe model is schematically shown in Fig. 7. It consists of axis parameters (rotational (ϕ, θ, ψ) and translational (x, y, z)), mass, torques, and rotors. The environment model describes the effects of external factors on the quadcopter. It consists of atmosphere and gravity models. The sensor model includes three sensors: 1) sonar for altitude measurement; 2) a camera for optical flow estimation; and 3) inertial measurement units to measure the linear and rotational motions. The flight control system (FCS) contains the roll ϕ, pitch θ, yaw ψ, and altitude z controllers. The mathematical model of the system is given by

$$\dot{X} = F(x, u) + N$$

where

$$\begin{aligned} X &= [x\, y\, z\, \phi\, \theta\, \psi]^T \\ u &= [u_1\, u_2\, u_3\, u_4]^T \\ N &= [n_1\, n_2\, n_3\, n_4\, n_5\, n_6]^T. \end{aligned} \tag{30}$$

Here, X is the state vector consisting of translational and rotational components, N contains the external disturbances affecting the system states, and u represents the model inputs. Let Ω_1, Ω_2, Ω_3, and Ω_4 be the angular speeds of the four rotors of the quadcopter. Then

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ \Omega_r \end{bmatrix} = \begin{bmatrix} b\left(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2\right) \\ b\left(-\Omega_2^2 + \Omega_4^2\right) \\ b\left(\Omega_1^2 - \Omega_3^2\right) \\ d\left(-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2\right) \\ -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4 \end{bmatrix}. \tag{31}$$

TABLE I
VARIOUS PARAMETERS OF THE PARROT MAMBO QUADCOPTER MODEL

S. No.	Parameters	Values	Symbol
1	Lateral moment arm	6.24×10^2 m	L_a
2	Quadcopter mass	6.3×10^{-2} kg	m
3	Drag coefficient	7.84×10^{-4} N·m·s^2	d
4	Thrust coefficient	1.7×10^{-2} N·s^2	b
5	Rotor moment of inertia	1.02×10^{-6} Kg·m^2	J_r
5	Roll moment of inertia	5.8×10^{-5} Kg·m^2	I_{xx}
5	Yaw moment of inertia	10^{-4} Kg·m^2	I_{zz}
5	Pitch moment of inertia	7.1×10^{-5} Kg·m^2	I_{yy}

Here, u_2, u_3, and u_4 control the roll, pitch, and yaw angles. u_1 is the total thrust input, and it controls the altitude z of the quadcopter. b is the thrust coefficient, d is the drag coefficient, and Ω_r is the residual angular speed.

The state equations of the system are

$$\dot{X} = \begin{bmatrix} \dot{\phi} \\ \ddot{\phi} \\ \dot{\theta} \\ \ddot{\theta} \\ \dot{\psi} \\ \ddot{\psi} \\ \dot{x} \\ \ddot{x} \\ \dot{y} \\ \ddot{y} \\ \dot{z} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} \phi \\ \dot{\theta}\dot{\psi}\frac{I_{yy}-I_{zz}}{I_{xx}} + \dot{\theta}\frac{J_r}{I_{xx}}\Omega_r + \frac{L_a}{I_{xx}}u_2 \\ \dot{\theta} \\ \dot{\phi}\dot{\psi}\frac{I_{zz}-I_{xx}}{I_{yy}} - \dot{\phi}\frac{J_r}{I_{yy}}\Omega_r + \frac{L_a}{I_{yy}}u_3 \\ \dot{\psi} \\ \dot{\theta}\dot{\phi}\frac{I_{xx}-I_{yy}}{I_{zz}} + \frac{1}{I_{zz}}u_4 \\ \dot{x} \\ (\cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi)\frac{u_1}{m} \\ \dot{y} \\ (\cos\phi\sin\theta\sin\psi + \sin\phi\cos\psi)\frac{u_1}{m} \\ \dot{z} \\ g - (\cos\phi\cos\theta)\frac{u_1}{m} \end{bmatrix} + \begin{bmatrix} 0 \\ n_1 \\ 0 \\ n_2 \\ 0 \\ n_3 \\ 0 \\ n_4 \\ 0 \\ n_5 \\ 0 \\ n_6 \end{bmatrix} \tag{32}$$

where I_{xx}, I_{yy}, and I_{zz} are the moments of inertia along the three axes. Various parameters of the quadcopter model are given in Table I. It should be noted that this quadcopter model is used only to generate the data. These data are then used to extract the DFIS model, as explained in the next section.

B. Extracting the Data-Driven Type-2 Fuzzy Model

Here, in these simulations, we seek to model the altitude controller of the quadcopter. A training dataset that contains the samples of input and output of the altitude controller is a requirement for Algorithm 1. The requirement was satisfied by controlling the altitude of quadcopter using the PD controller in MATLAB. The dataset containing the inputs and output of the PD controller was created. Later, Algorithm 1 was used to generate the DFIS model of the altitude controller using this input and output dataset. The optimum error threshold $\tau_{E_{\text{op}}}$ was calculated to be 0.1. Algorithm 1 extracted 26 rules from dataset, and these formed the rule base R_b.

The number of rules in R_b was reduced using Algorithm 2. The optimum reduction threshold $\tau_{R_{\text{op}}}$ was calculated to be 0.15. Algorithm 2 reduced the number of rules to 17 by merging a few rules. This is called the reduced rule base R_r. A few interval T2DFs (3 out of 17) of each input are shown in Fig. 8. R_r and all the interval T2DFs in it collectively form a reduced (simplified) DFIS model. The reduced error surface plot for the DFIS model is shown in Fig. 9. It can be seen that the reduced error surface (E_r) is less than $\tau_{R_{\text{op}}}$ (i.e., 0.15). The surface of the DFIS model is shown in Fig. 10.

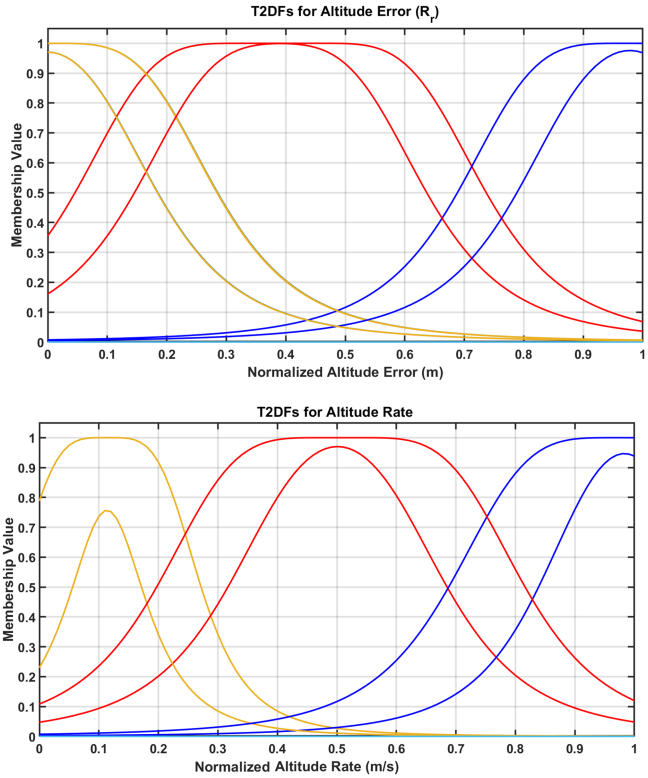

Fig. 8. Three interval T2DFs of each normalized input (DFIS model).

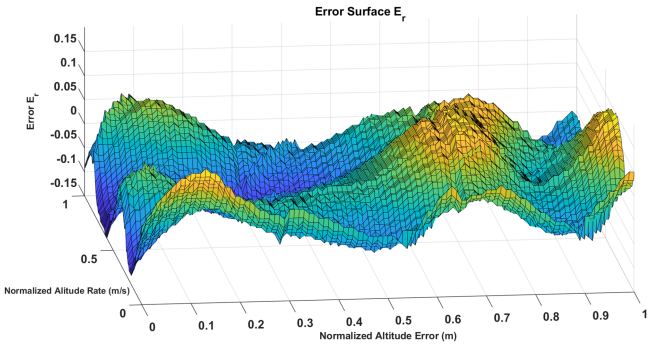

Fig. 9. Reduced error surface (E_r) plot for the DFIS model.

For comparison purpose, various fuzzy models were tested. The ANFIS type-1 model was extracted from the same input and output dataset using the MATLAB fuzzy logic toolbox. The dataset was divided into train and test sets (2:1). The model was trained for 100 epochs and 36 rules were generated. It is a Sugeno-based fuzzy model. Fig. 11 shows the surface plot of the ANFIS type-1 Sugeno model. The fuzzy logic toolbox can be used to convert the type-1 fuzzy model to a type-2 fuzzy model. Therefore, an ANFIS type-2 model was also generated. This

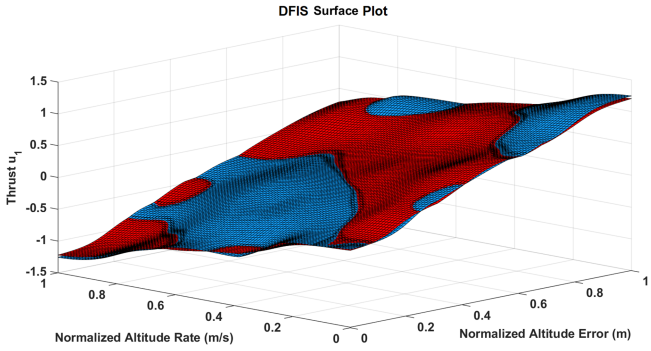

Fig. 10. Surface plot of the DFIS model. The upper surface is in blue and the lower surface is in red.

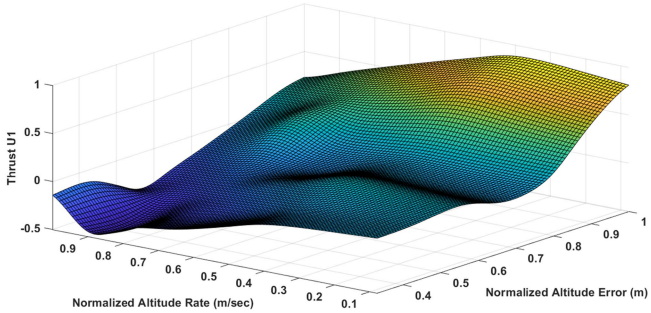

Fig. 11. Surface plot of the ANFIS type-1 model.

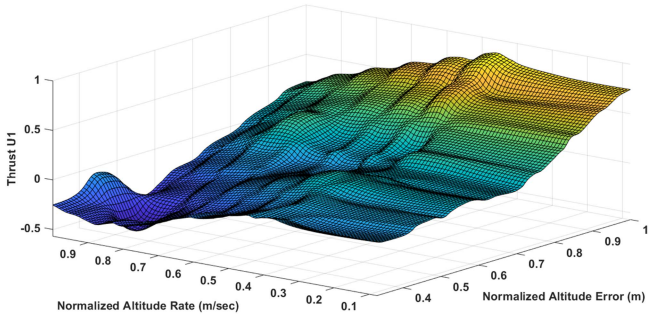

Fig. 12. Surface plot of the ANFIS type-2 model.

TABLE II
PERFORMANCE COMPARISON OF THE PROPOSED DFIS MODEL WITH OTHER MODELS

S. No.	Model	Number of rules	Number of tunable parameters	Membership functions used
1	ANFIS Type-1	36	144	Bell shape
2	ANFIS Type-2	36	168	Type-2 Bell shape
3	IT2FNN [43]	72	336	T2GMF
4	MIT2FC [43]	36	168	T2GMF
5	Proposed DFIS model	17	34	T2DF

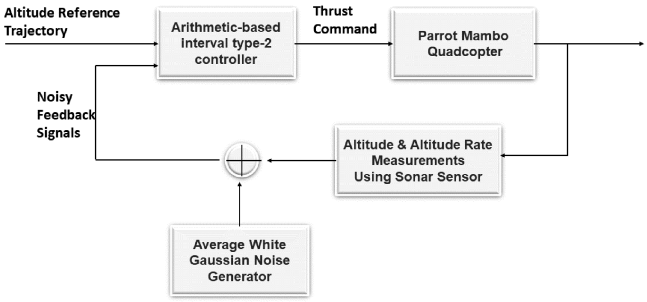

Fig. 13. Structure of the designed control system.

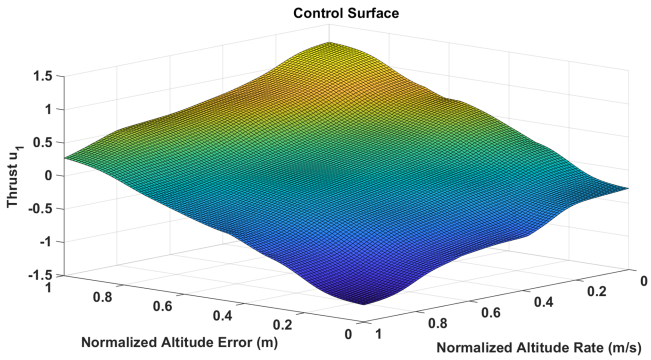

Fig. 14. Control surface of the arithmetic-based controller.

type-2 model uses the type-2 bell-shaped membership functions (see Fig. 12). Table II summarizes the key differences among the DFIS and various other fuzzy models.

C. Designing the Arithmetic-Based Type-2 Altitude Controller

The objective is to control the altitude z by generating an appropriate total thrust u_1. The thrust u_1 depends on the height (sonar) measurements and the rate of change of the height of the quadcopter. Using Algorithm 3, an arithmetic-based controller was designed using R_r. The structure of the designed closed-loop control system is presented as a block diagram in Fig. 13. Fig. 14 shows the surface plot of the arithmetic-based controller. The ANFIS fuzzy models (type-1 and type-2) were converted to a fuzzy controller using the Simulink fuzzy logic toolbox. These controllers were based on Sugeno inference logic. Then, four controllers (arithmetic-based, ANFIS type-1, ANFIS type-2, and PD controllers) were used to regulate the altitude of the quadcopter in MATLAB Simulink. Average white Gaussian noise (AWGN) with a signal-to-noise ratio (SNR) of 10 dB was added to the sensor measurements. Altitude response was simulated with noisy feedback signals. The quadcopter was programmed to take off and reach an altitude of 0.7 m, then rise to an altitude of 1 m, and finally descend to 0.7 m. Fig. 15 shows the altitude response of the controlled quadcopter during this simulation study. It is evident from Fig. 15 that the proposed controller produced oscillations with less amplitude compared with other three controllers. It should be noted that oscillations are magnified for the purpose of comparison in Fig. 15. To supplement the comparison results with quantitative analysis, three separate variation measuring metrics [variance, range, and root-mean-square error (RMSE)] were computed from the responses of all the four controllers. These metrics

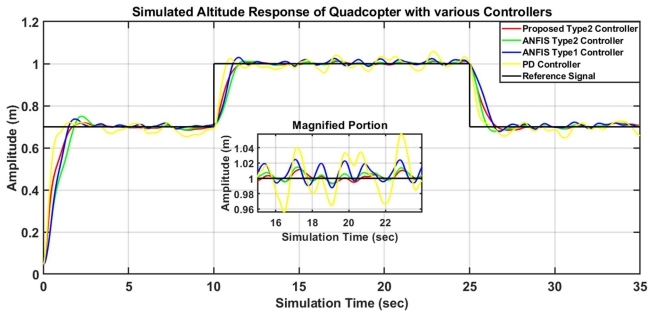

Fig. 15. Comparison of altitude response of various controllers (in MATLAB Simulink). The measurements got from the altitude sensor (sonar) were corrupted with AWGN.

TABLE III
PERFORMANCE COMPARISON OF THE PROPOSED TYPE-2 CONTROLLER WITH OTHER CONTROLLERS

S. No.	Controller	Range	Variance	RMSE
1	ANFIS Type-1 Controller	0.0371	9.88×10^{-5}	1.2×10^{-2}
2	ANFIS Type-2 controller	0.0216	6.24×10^{-5}	5.9×10^{-3}
3	PD Controller	0.1023	6.08×10^{-4}	2.48×10^{-2}
4	Proposed Type-2 Controller	0.0182	1.85×10^{-5}	4.6×10^{-3}

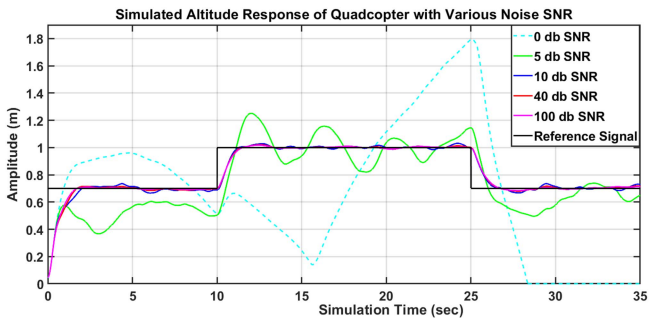

Fig. 16. Simulated altitude response of the quadcopter with various noise levels.

were computed in the time interval ($14 \leq$ simulation time ≤ 25 in Fig. 15). The results obtained are shown in Table III. These results clearly indicate that the performance of the proposed type-2 controller was least affected by the noise in the feedback altitude signal compared with other controllers.

To further investigate the robustness property of the proposed controller, several simulations were performed to regulate the altitude of the quadcopter with various noise levels. AWGN was added to the data of the altitude sensor. This noisy altitude signal is the feedback signal and is the input to the proposed controller. Fig. 16 shows the altitude response of the quadcopter receiving noisy altitude sensor data with various SNRs. A higher value of SNR indicates that the signal quality is better and vice versa.

It is evident from Fig. 16 that when the noise level was negligible (SNR 100 dB), the proposed controller regulated the altitude without any oscillations (magenta line) and perfectly followed the reference signal (black line). However, when the SNR was increased, the oscillations became more visible in the response. For an SNR of 5 dB, the controller was still able to regulate the altitude but with high oscillations. The borderline case was reached when the signal and noise had equal power (SNR 0 dB). In this case, the proposed controller could not differentiate between the signal and noise. In this case, it was not able to regulate the altitude (cyan line) and the quadcopter crashed.

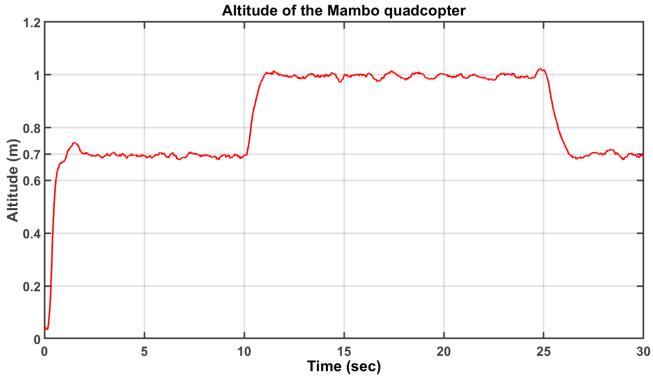

Fig. 17. Actual trajectory followed by the Mambo quadcopter. (Flight data.[1])

D. Hardware Implementation

MATLAB provides a support package for parrot quadcopter drones (Mambo FPV and Bebop2). It connects with the quadcopter hardware via Bluetooth/Wi-Fi, and it can send control commands. MATLAB Simulink includes the simulation model of the quadcopter. The model contains the algorithm for the FCS. This algorithm implements roll, pitch, yaw, and altitude controllers. The support package generates the C code of the FCS and deploys it in the quadcopter. This algorithm can access the onboard sensors such as the accelerometer, gyroscopes, camera, and sonar. The flight data values (altitude, images, etc.) are saved in onboard storage, and they can be retrieved from the quadcopter at the end of the flight. The altitude controller in the FCS was replaced with the proposed arithmetic-based type-2 controller. The C code of the FCS was generated and deployed in the quadcopter via Bluetooth. A fixed flight trajectory corresponding to the simulation scenario was also hard coded. The uncertainty in the controller input (altitude) data was created by adding white noise to the sonar measurements. The noisy data were the input to the controller in the FCS. The drone flight was tested in a protected (indoor) environment. The data recorded during the flight were retrieved at the end. Fig. 17 shows the altitude measurements recorded during this test flight. This tells us that the proposed controller successfully followed the desired trajectory, even in the presence of noisy sensor data.

E. Results and Discussion

From the results obtained, it is evident that the DFIS model can be extracted directly from the process data. The DFIS model

[1][Online]. Available: https://github.com/Abrarlaghari/Mambo-Quadcopter-Simulink.git

is composed of interval T2DFs and rules based on the Dombi operator. As shown in Fig. 10, the proposed model has a smooth surface, and it covers the whole input space. In these simulations, the λ and ϵ parameters were kept fixed, the c parameter was determined directly from the data, and ν was calculated using (17). This reduces the number of parameter calculations per interval T2DF. In Fig. 11, it can be seen that the ANFIS type-1 model also has a smooth surface except for the boundary areas. On the boundaries, there are a few bumps. The ANFIS type-2 model (see Fig. 12) has a less smooth surface compared to those for the DFIS and ANFIS type-1 models. Boundary bumps in ANFIS models are due to the fact that only a few data points are available to accurately model these areas. However, these bumps are nonexistent in the DFIS model because of the long tails of interval T2DFs. These tails tend to approximate a smooth transition in the areas where there are only a few data points or even no data points.

The performance of the DFIS model is evaluated by comparing it with the output of the ANFIS type-1 and ANFIS type-2 fuzzy models. These are two well-established and widely accepted fuzzy modeling approaches. As shown in Table II, the DFIS model used a smaller number of rules and tunable parameters than the ANFIS type-1 and ANFIS type-2 models did. Figs. 10, 11, and 12 show the surface generated by these three models. It is evident that the DFIS model has much smoother surface transitions. Furthermore, the qualitative aspects of these three models are compared by designing the fuzzy controller. The controllers are based on the rules and membership functions in these models. These controllers are then used to control the altitude of the quadcopter in the presence of noisy sensor data. The altitude control responses of these three controllers are shown in Fig. 15 for comparison purposes. The performance of the ANFIS type-2 controller is worse than the ANFIS type-1 and proposed arithmetic-based controllers. The proposed controller is better than the ANFIS type-1 controller as it is more robust to the variations in the noisy sensor measurements. In addition to the simulations, the hardware implementations of the proposed controller showed very promising results for the real-time control applications. The controller performed the altitude control function in the presence of uncertain (noisy) measurement data (see Fig. 17). This demonstrates the effectiveness of the proposed arithmetic-based type-2 controller.

In ANFIS type-1 and type-2 controllers, the choice of membership functions and number of rules is hyperparameters. One has to decide in advance on the type of membership functions (triangular, Gaussian, trapezoidal, etc.) and the number of categories and rules. This hyperparameter selection is usually based on experience or data insights. However, in our proposed DFIS model, the number of rules and interval T2DF are determined automatically by the algorithm based on the reduction threshold τ_R.

The reduction threshold τ_R plays a key role in determining the generalization performance of the proposed methodology. A value of reduction threshold τ_R less than $\tau_{R_{\text{op}}}$ [see (23)] will lead to a large number of rules. This is similar to overfitting in the learning paradigms. The DFIS model will accurately fit the training data, but it will have poor generalization. This type of DFIS model will be less interpretable, and it performs badly on unseen data. A value of τ_R greater than $\tau_{R_{\text{op}}}$ will produce a DFIS model with a smaller number of rules. This type of DFIS model will have a good generalization capability and will be more interpretable with low computational complexity. However, in this case, the accuracy of the DFIS model will be poor. Thus, an optimum value of the threshold $\tau_{R_{\text{op}}}$ will lead to an optimum number of rules, and it will produce a DFIS model with higher accuracy, better interpretability, and a good generalization capability. The proposed DFIS methodology is data driven. It means that the technique can be applied in cases where the expert knowledge is not available, but the data of the system is available. The methodology presented in this article is very general, and the only assumption is the availability of the data. We want to emphasize that the proposed methodology is very general and can be used to model and control any real-world physical system. The altitude control of quadcopter is studied here only to validate the proposed methodology and to compare it with various other fuzzy modeling and control techniques.

As mentioned in the introduction section, most of the developed methods extract the fuzzy rule and tune the parameters using the metaheuristics. However, these algorithms can get stuck in a local optimum, and there is no guarantee that these will find the global optimum solution. In addition, the local optimum solution provided by these algorithms depends on the initial values of the tunable parameter. These tunable parameters are usually randomly initialized, and then, a heuristic is applied to get the optimum value. The process is repeated with several random initial values, and the best values are chosen at the end via a comparison of the results achieved with each random initialization. The algorithm proposed in this article uses a different approach to tune the parameters of the DFIS model. Values of a few parameters are fixed, and they are never changed during the training phase. For the remaining tunable parameters, there are closed-form expressions [see (16) and (17)], which can precisely determine the values of these parameters. Rules are added iteratively, and the parameters in these rules are determined using the expressions. The rule addition process continues until the error is within the bounds of reduction threshold τ_R. As no heuristic is employed here, the values determined for the tunable parameters are near optimum.

F. Interpretability of the DFIS Model and Results

The proposed DFIS model is based on rules that have the following form:

$$\begin{aligned}&\text{if } x_1 \text{ is around } c_1 \text{ and } \ldots\ldots \text{ and } x_n \text{ is around } c_n\\&\text{then } y \text{ is } v_1\end{aligned} \tag{33}$$

where $x_1, \ldots, x_n$ are the input variables and $c_1, \ldots, c_n$ are the peak coordinates of the corresponding interval T2DFs. y is the output variable and v_1 is the value of the output. The rule is self-explanatory, and it is very easy to interpret a single rule. The identified DFIS model consists of a number of these types of rules. Therefore, the interpretability of the whole DFIS model

depends on the number of these rules. The small number of rules leads to a more interpretable DFIS model and vice versa. As shown in Table II, the proposed DFIS model has the least number of rules compared with various other models. Therefore, the DFIS model is comparatively more interpretable.

In the simulations, 17 rules were identified by the proposed approach. Three of these rules are as follows.

1) *If the normalized altitude error is around 0 and the normalized altitude rate is around 0, then the total thrust is −0.469.*
2) *If the normalized altitude error is around 1 and the normalized altitude rate is around 1, then the total thrust is 0.417.*
3) *If the normalized altitude error is around 0 and the normalized altitude rate is around 1, then the total thrust is −1.036.*

For a given set of values of the input variables x_1 and x_2, then the upper and lower firing strengths ($\overline{v}_i(\underline{x}^*)$ and $\underline{v}_i(\underline{x}^*)$) of each rule are calculated using (26). These firing strengths are then multiplied with the corresponding $thrust$ values (in matrix V) to get the final crisp output value using (28). In this systematic way, the DFIS model can easily be interpreted.

G. Comparison With Closely Related Work

Le et al. [43] designed an interval type-2 fuzzy controller (MIT2FC) by proposing a multilayer structure. The network parameters were tuned using gradient-based methods. The performance of the proposed model was demonstrated using the trajectory control of a quadcopter UAV. There are some key differences between MIT2FC and our proposed methodology.

1) The MIT2FC model employed type-2 Gaussian membership functions (T2GMFs). A separate T2GMF is required to cover a specific region in the input space. Therefore, to cover the whole input space completely, a large number of rules are required. However, in the proposed design, interval T2DFs are used. The interval T2DF has a long tail, and the whole input space can be covered by identifying a few important rules. This is evident from the results in Table II. Therefore, our proposed strategy can reduce the complexity of the fuzzy model.
2) A fixed number of rules are defined in the MIT2FC. The parameters of these T2GMFs are learnt using the gradient-based optimization method. The layer structure is fixed and so is the number of rules. This results in a less flexible model. However, in the DFIS design, the number of rules is variable, and rules are learnt dynamically depending on the error thresholds. Furthermore, the number of rules is further decreased by combining the interval T2DFs, which are acting in the close proximity.
3) The parameters of T2GMFs are learnt using the gradient-based method in the MIT2FC. If the surface has sharp boundaries (discontinuities), then it is difficult to optimize the parameters using this method. However, in the DFIS model, the parameters of the interval T2DF are calculated using a closed-form equation [see (17)], and it is guaranteed to work well even in the presence of discontinuities.
4) The efficiency of the MIT2FC model was demonstrated using numerical examples (simulation studies) only. However, the performance of the proposed DFIS model is checked by real-time implementation on the quadcopter hardware (Parrot mini-drone Mambo FPV). The flight data are available.[2] Owing to the reduced number of rules and arithmetic-based inference, the DFIS-based controller was able to produce the desired flight trajectory (see Fig. 17).

V. Conclusion

In this article, we presented the solutions to some of the limitations associated with the existing fuzzy type-2 modeling and control techniques. A procedure was proposed to identify the type-2 model directly from the data, which we called the DFIS model. This model consists of rules and interval T2DFs. The whole input space is covered using a few rules. Interval T2DFs can model various types of uncertainties using their parameters. A rule reduction procedure is also proposed. It combines the interval T2DFs in the close vicinity, and it significantly reduces the number of rules. The DFIS model was compared with various type-1 and type-2 fuzzy models. The DFIS model produced a smooth surface with a comparatively small number of rules and tunable parameters. Furthermore, a procedure is proposed to design an arithmetic-based interval type-2 fuzzy controller using the rules. As controller design does not include the implication and type-reduction steps, this greatly reduces the computational complexity and paves the way for the real-time implementation of the proposed design. The effectiveness of the whole procedure was demonstrated by designing an altitude controller for the Parrot Mambo quadcopter. Simulations were carried out in MATLAB Simulink to compare the proposed controller with various fuzzy controllers. The proposed controller performed better and regulated the altitude of the quadcopter even in the presence of noisy (uncertain) sensor measurements. This robustness to noisy data is due to the use of interval T2DFs. The designed controller was then deployed and tested in the FCS on the quadcopter hardware. Real-time hardware implementations produced the same results as those obtained in the simulations. Because of its low computational complexity and design simplicity, the controller is suitable for real-time control applications.

The proposed DFIS model has a few limitations. It can produce an interpretable and computationally less expensive model only if the number of features (inputs) is small. For a large number of features, a fuzzy surface has to be constructed in higher dimension, and also, a large number of complex rules are required to correctly capture the interfeature correlations. In this case, the DFIS model will still be accurate, but it will be less interpretable. Therefore, the proposed methodology is only applicable to application domains where the number of features is small, such as control system design and financial modeling. For other domains, where the number of features is large (computer vision and natural language processing), the DFIS will produce a very complex model with low interpretability. Future work includes the designing of an ensemble of the

[2][Online]. Available: https://github.com/Abrarlaghari/Mambo-Quadcopter-Simulink.git

DFIS models to scale the proposed methodology for a complex system with a large feature set. Another possible future direction is to extend the proposed DFIS model to include the general T2MFs. An arithmetic-based control design procedure could also be extended to design the general type-2 controllers with reduced computational complexity. This would help to model the uncertainties associated with the real-world physical systems with more accuracy and less complexity.

REFERENCES

[1] R. R. Yager and L. A. Zadeh, *An Introduction to Fuzzy Logic Applications in Intelligent Systems*, vol. 165. New York, NY, USA: Springer, 2012.

[2] L. Yu and Y.-Q. Zhang, "Evolutionary fuzzy neural networks for hybrid financial prediction," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 35, no. 2, pp. 244–249, May 2005.

[3] M. Mahfouf, M. F. Abbod, and D. A. Linkens, "A survey of fuzzy logic monitoring and control utilisation in medicine," *Artif. Intell. Med.*, vol. 21, no. 1–3, pp. 27–42, 2001.

[4] C. Li, J. Zhou, L. Chang, Z. Huang, and Y. Zhang, "T–S fuzzy model identification based on a novel hyperplane-shaped membership function," *IEEE Trans. Fuzzy Syst.*, vol. 25, no. 5, pp. 1364-1370, Oct. 2017.

[5] P. P. Angelov and D. P. Filev, "An approach to online identification of Takagi-Sugeno fuzzy models," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 34, no. 1, pp. 484–498, Feb. 2004.

[6] S.-H. Tsai and Y.-W. Chen, "A novel identification method for Takagi-Sugeno fuzzy model," *Fuzzy Sets Syst.*, vol. 338, pp. 117-135, 2018.

[7] E. Mamdani and S. Assilian, "An experiment in linguistic synthesis with a fuzzy logic controller," *Int. J. Man-Mach. Stud.*, vol. 7, no. 1, pp. 1–13, 1975.

[8] M. Sugeno, *Industrial Applications of Fuzzy Control*. Amsterdam, The Netherlands: Elsevier, 1985.

[9] L.-C. Duţu, G. Mauris, and P. Bolon, "A fast and accurate rule-base generation method for Mamdani fuzzy systems," *IEEE Trans. Fuzzy Syst.*, vol. 26, no. 2, pp. 715–733, Apr. 2018.

[10] H. Ying and F. Lin, "Online self-learning fuzzy discrete event systems," *IEEE Trans. Fuzzy Syst.*, vol. 28, no. 9, pp. 2185–2194, Sep. 2020.

[11] J. M. Mendel and P. P. Bonissone, "Critical thinking about explainable AI (XAI) for rule-based fuzzy systems," *IEEE Trans. Fuzzy Syst.*, vol. 29, no. 12, pp. 3579–3593, Dec. 2021.

[12] L. dos Santos Coelho and B. M. Herrera, "Fuzzy identification based on a chaotic particle swarm optimization approach applied to a nonlinear yo-yo motion system," *IEEE Trans. Ind. Electron.*, vol. 54, no. 6, pp. 3234–3245, Dec. 2007.

[13] R. Zhang and J. Tao, "A nonlinear fuzzy neural network modeling approach using an improved genetic algorithm," *IEEE Trans. Ind. Electron.*, vol. 65, no. 7, pp. 5882–5892, Jul. 2018.

[14] H. Hellendoorn and D. Driankov, *Fuzzy Model Identification: Selected Approaches*. New York, NY, USA: Springer, 2012.

[15] R.-E. Precup and H. Hellendoorn, "A survey on industrial applications of fuzzy control," *Comput. Ind.*, vol. 62, no. 3, pp. 213–226, 2011.

[16] A. Piegat, *Fuzzy Modeling and Control*, vol. 69. Heidelberg, Germany: Physica-Verlag GmbH, 2013.

[17] G. Osório, J. Matias, and J. Catalão, "Short-term wind power forecasting using adaptive neuro-fuzzy inference system combined with evolutionary particle swarm optimization, wavelet transform and mutual information," *Renewable Energy*, vol. 75, pp. 301–307, 2015.

[18] W. Chen, M. Panahi, and H. R. Pourghasemi, "Performance evaluation of GIS-based new ensemble data mining techniques of adaptive neuro-fuzzy inference system (ANFIS) with genetic algorithm (GA), differential evolution (DE), and particle swarm optimization (PSO) for landslide spatial modelling," *Catena*, vol. 157, pp. 310–324, 2017.

[19] D. Karaboga and E. Kaya, "Adaptive network based fuzzy inference system (ANFIS) training approaches: A comprehensive survey," *Artif. Intell. Rev.*, vol. 52, no. 4, pp. 2263–2293, 2019.

[20] S. Harifi, M. Khalilian, J. Mohammadzadeh, and S. Ebrahimnejad, "Optimizing a neuro-fuzzy system based on nature-inspired emperor penguins colony optimization algorithm," *IEEE Trans. Fuzzy Syst.*, vol. 28, no. 6, pp. 1110–1124, Jun. 2020.

[21] L. A. Zadeh, "The concept of a linguistic variable and its application to approximate reasoning-I," *Inf. Sci.*, vol. 8, no. 3, pp. 199–249, 1975.

[22] N. N. Karnik, J. M. Mendel, and Q. Liang, "Type-2 fuzzy logic systems," *IEEE Trans. Fuzzy Syst.*, vol. 7, no. 6, pp. 643–658, Dec. 1999.

[23] 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.

[24] J. M. Mendel, "Computing with words: Zadeh, Turing, Popper and Occam," *IEEE Comput. Intell. Mag.*, vol. 2, no. 4, pp. 10–17, Nov. 2007.

[25] D. Wu and W. W. Tan, "Genetic learning and performance evaluation of interval type-2 fuzzy logic controllers," *Eng. Appl. Artif. Intell.*, vol. 19, no. 8, pp. 829–841, 2006.

[26] H. Hagras, "Type-2 FLCs: A new generation of fuzzy controllers," *IEEE Comput. Intell. Mag.*, vol. 2, no. 1, pp. 30–43, Feb. 2007.

[27] R. H. Abiyev and O. Kaynak, "Type 2 fuzzy neural structure for identification and control of time-varying plants," *IEEE Trans. Ind. Electron.*, vol. 57, no. 12, pp. 4147–4159, Dec. 2010.

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

[29] J. Mendel, H. Hagras, W.-W. Tan, W. W. Melek, and H. Ying, *Introduction to Type-2 Fuzzy Logic Control: Theory and Applications*. Hoboken, NJ, USA: Wiley, 2014.

[30] A. Niewiadomski, "A type-2 fuzzy approach to linguistic summarization of data," *IEEE Trans. Fuzzy Syst.*, vol. 16, no. 1, pp. 198–212, Feb. 2008.

[31] F. Gaxiola, P. Melin, F. Valdez, and O. Castillo, "Interval type-2 fuzzy weight adjustment for backpropagation neural networks with application in time series prediction," *Inf. Sci.*, vol. 260, pp. 1–14, 2014.

[32] M. Naderipour, M. H. F. Zarandi, and S. Bastani, "A type-2 fuzzy community detection model in large-scale social networks considering two-layer graphs," *Eng. Appl. Artif. Intell.*, vol. 90, 2020, Art. no. 103206.

[33] F. Cuevas, O. Castillo, and P. Cortés-Antonio, "Design of a control strategy based on type-2 fuzzy logic for omnidirectional mobile robots," *J. Multiple-Valued Log. Soft Comput.*, vol. 37, pp. 107–136, 2021.

[34] K. Mittal, A. Jain, K. S. Vaisla, O. Castillo, and J. Kacprzyk, "A comprehensive review on type 2 fuzzy logic applications: Past, present and future," *Eng. Appl. Artif. Intell.*, vol. 95, 2020, Art. no. 103916.

[35] J. Mendel, *Uncertain Rule-based Fuzzy Logic Systems: Introduction and New Directions*. Englewood Cliffs, NJ, USA: Prentice-Hall, 2000, pp. 25–200.

[36] H. Hassani and J. Zarei, "Interval type-2 fuzzy logic controller design for the speed control of DC motors," *Syst. Sci. Control Eng.*, vol. 3, no. 1, pp. 266–273, 2015.

[37] J. Zheng, W. Du, I. Nascu, Y. Zhu, and W. Zhong, "An interval type-2 fuzzy controller based on data-driven parameters extraction for cement calciner process," *IEEE Access*, vol. 8, pp. 61775–61789, 2020.

[38] S. Bhattacharyya, D. Basu, A. Konar, and D. Tibarewala, "Interval type-2 fuzzy logic based multiclass ANFIS algorithm for real-time EEG based movement control of a robot arm," *Robot. Auton. Syst.*, vol. 68, pp. 104–115, 2015.

[39] T.-L. Le, "Intelligent fuzzy controller design for antilock braking systems," *J. Intell. Fuzzy Syst.*, vol. 36, no. 4, pp. 3303–3315, 2019.

[40] A. Mohammadzadeh, M. H. Sabzalian, and W. Zhang, "An interval type-3 fuzzy system and a new online fractional-order learning algorithm: Theory and practice," *IEEE Trans. Fuzzy Syst.*, vol. 28, no. 9, pp. 1940–1950, Sep. 2020.

[41] Z.-X. Wei, F. Doctor, Y.-X. Liu, S.-Z. Fan, and J.-S. Shieh, "An optimized type-2 self-organizing fuzzy logic controller applied in anesthesia for propofol dosing to regulate BIS," *IEEE Trans. Fuzzy Syst.*, vol. 28, no. 6, pp. 1062–1072, Jun. 2020.

[42] M. Pratama, J. Lu, and G. Zhang, "Evolving type-2 fuzzy classifier," *IEEE Trans. Fuzzy Syst.*, vol. 24, no. 3, pp. 574–589, Jun. 2016.

[43] T.-L. Le, N. V. Quynh, N. K. Long, and S. K. Hong, "Multilayer interval type-2 fuzzy controller design for quadcopter unmanned aerial vehicles using Jaya algorithm," *IEEE Access*, vol. 8, pp. 181246–181257, 2020.

[44] K. W. Wong, D. Tikk, T. D. Gedeon, and L. T. Kóczy, "Fuzzy rule interpolation for multidimensional input spaces with applications: A case study," *IEEE Trans. Fuzzy Syst.*, vol. 13, no. 6, pp. 809–819, Dec. 2005.

[45] K. Tai, A.-R. El-Sayed, M. Biglarbegian, C. I. Gonzalez, O. Castillo, and S. Mahmud, "Review of recent type-2 fuzzy controller applications," *Algorithms*, vol. 9, no. 2, 2016, Art. no. 39.

[46] E. Ontiveros-Robles, P. Melin, and O. Castillo, "New methodology to approximate type-reduction based on a continuous root-finding Karnik Mendel algorithm," *Algorithms*, vol. 10, no. 3, 2017, Art. no. 77.

[47] J. Dombi and A. Hussain, "Interval type-2 fuzzy control using distending function," in *Fuzzy Systems and Data Mining V*. Amsterdam, The Netherlands: IOS Press, 2019, pp. 705–714.

[48] L. A. Zadeh, "Outline of a new approach to the analysis of complex systems and decision processes," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-3, no. 1, pp. 28–44, Jan. 1973.

[49] J. Dombi and T. Jónás, "Kappa regression: An alternative to logistic regression," *Int. J. Uncertainty, Fuzziness Knowl.-Based Syst.*, vol. 28, no. 2, pp. 237–267, 2020.
[50] J. Dombi and A. Hussain, "A new approach to fuzzy control using the distending function," *J. Process Control*, vol. 86, pp. 16–29, 2020.
[51] J. Dombi, "A general class of fuzzy operators, the DeMorgan class of fuzzy operators and fuzziness measures induced by fuzzy operators," *Fuzzy Sets Syst.*, vol. 8, no. 2, pp. 149–163, 1982.
[52] MathWorks MATLAB Hardware Team, *"Parrot drone support from MATLAB*." Accessed: Mar. 11, 2020. [Online]. Available: https://www.mathworks.com/hardware-support/parrot-drone-matlab.html
[53] T. Tuna, A. Beke, and T. Kumbasar, "Deep learning frameworks to learn prediction and simulation focused control system models," *Appl. Intell.*, vol. 52, no. 1, pp. 662–679, 2022.
[54] T. Luukkonen, "Modelling and control of quadcopter," in *Independent Research Project in Applied Mathematics*, vol. 22. Espoo, Finland: Aalto Univ., 2011.

József Dombi received the Ph.D. (*summa cum laude*), CSc (fuzzy sets' structure in the aspect of multicriteria decision aid), and D.Sc. (one class of continuous valued logical operators and its application: pliant concept) degrees from the University of Szeged, Szeged, Hungary, in 1977, 1994, and 2012, respectively.

He is a Professor Emeritus with the University of Szeged, Szeged, Hungary. He is also the Founder and Head of Dopti Ltd., Szeged. He teaches intelligent visualization, theory of artificial intelligence, fuzzy systems, multicriteria decision making, and intelligent systems. His research interests include computational intelligence, theory of fuzzy sets, multicriteria decision making, genetic and evolutional algorithms, and operation research.

Abrar Hussain received the M.Sc. degree in systems engineering from the Pakistan Institute of Engineering and Applied Sciences, Islamabad, Pakistan, in 2012. He is currently working toward the Ph.D. degree in applicability of fuzzy theory in machine learning with the University of Szeged, Szeged, Hungary.

He conducts research to develop new types of fuzzy type-2 inference systems and merging fuzzy operators with machine learning models. His research interests include artificial intelligence and machine learning.