An Open Source Matlab/Simulink Toolbox for Interval Type-2 Fuzzy Logic Systems

Ahmet Taskin
Department of Software and Electronics
AVL Research and Engineering
Istanbul, Turkey
ahmet.taskin@avl.com

Tufan Kumbasar
Department of Control and Automation Engineering
Istanbul Technical University
Istanbul, Turkey
kumbasart@itu.edu.tr

Abstract—In the last two decades, we have witnessed that Interval Type-2 Fuzzy Logic Systems (IT2-FLSs) have been successfully implemented in various engineering areas. In this paper, we will introduce a free open source Matlab/Simulink toolbox for the development of IT2-FLSs for a wider accessibility to users beyond the type-2 fuzzy logic community. The presented IT2-FLS toolbox allows intuitive implementation of Takagi-Sugeno-Kang (TSK) type IT2-FLSs where it is capable to cover all the phases of its design. In order to allow users to easily construct IT2-FISs, a GUI is developed which is similar to that of Matlab® Fuzzy Logic Toolbox. We have embedded various Type Reduction (TR) methods in the toolbox since the TR method is the most important operation that must be taken into consideration. This gives the opportunity to the user to examine the performance of his/her IT2-FLS design with respect to the TR methods. We have also developed an IT2-FLS Simulink library so that the designer can perform various simulation analyses and can also investigate the effect of TR method on the performance of the IT2-FLS. Moreover, the developed TSK IT2-FLS Matlab/Simulink toolbox contains an automatic connection between Matlab and Simulink environments. Once the user has finished the design of the IT2-FLS via the GUI, it is possible to export his/her design directly to Simulink via an automatic Simulink file generation. We believe that the availability of the developed free and open-source TSK IT2-FLS Matlab/Simulink toolbox will be an important step for a wider deployment and development of IT2-FLSs.

I. INTRODUCTION

In the last two decades, we have witnessed that Interval Type-2 (IT2) Fuzzy Logic Systems (FLSs) have been successfully implemented in various engineering areas [1-12]. It has been demonstrated that, in comparison with its Type-1 (T1) and conventional counterparts, IT2-FLSs resulted with performance improvements due to the additional degree of freedom provided by the Footprint of Uncertainty (FOU) present in their IT2 Fuzzy Sets (FSs) [10-16]. The internal structure of the IT2-FLS is similar to its T1 counterpart [15-17]. However, the major difference is that there is an extra Type Reduction (TR) procedure since IT2-FLSs employ and process IT2-FSs [17]. The widely used TR method is the Karnik-Mendel (KM) algorithm that calculates the type reduced set in an iterative manner [18-20]. However, due to its iterative nature, the computational cost of the calculation of the IT2-FLS output is relatively big [19-21]. In this context, several TR methods have been proposed for reducing the computational cost of the IT2 fuzzy inference mechanism. Wu [14], [20] categorized the TR methods as Enhancements to the KMs, which improved the computational cost of the KM, and Alternative TR methods, which are closed-form approximations to the KM algorithm. Nevertheless, the alternative TR methods cannot capture the features of the KM which are namely novelty and adaptiveness [14].

Toolboxes for the implementation of T1-FLSs are widespread and have led to the application of T1-FLSs in various engineering applications [22]. In [23], it has been stated that Fuzzy Logic Toolbox provided for Matlab® (The Mathworks, Inc.) is the highly popular one in engineering and computer science since it allows users to easily construct T1 Fuzzy Inference Systems (FISs) using a Graphical User Interface (GUI) [24]. On the other hand for IT2-FLSs, there does exist only few software tools in comparison. Mendel [16] and Wu [20] published various open source codes but the use of these source codes still requires knowledge about IT2-FLSs and thus they cannot be seen as toolkits in terms of using the IT2-FLSs relatively easy for researcher especially not with non-Computer Science background. Besides, researchers developed toolboxes for Matlab® that provides tools for the development of IT2-FLSs, including a GUI similar to that of Matlab® Fuzzy Logic Toolbox [26-28]. However, these researchers did not make their source code and toolboxes publicly available. Beyond MATLAB® toolboxes, Java based toolkits have been also presented for the development of type-2 FLSs [14], [30].

In this paper, we will introduce a free open source Matlab/Simulink toolbox for the development of Takagi-Sugeno-Kang (TSK) type IT2-FLSs for a wider accessibility to users beyond the type-2 fuzzy logic community. The developed IT2-FLS toolbox allows intuitive implementation of IT2-FLSs where it is capable to cover all the phases of its design. In order to allow users to easily construct TSK IT2-FISs, a GUI is developed which is similar to that of Matlab® Fuzzy Logic Toolbox. As it has been asserted above, the TR method is the most important operation that must be taken into consideration. Therefore, we embedded the most commonly or practically used TR and defuzzification methods (8 methods) in the toolbox so that the user can examine the performance of his/her IT2-FLS design for various TR methods. We have also developed an IT2-FLS Matlab/ Simulink library so that the

978-1-4799-7560-0/15 $31.00 © 2015 IEEE
DOI 10.1109/SSCI.2015.220

designer can perform various simulation analyses and investigate the effect of TR method on the performance of the IT2-FLS. Moreover, the developed IT2-FLS Matlab/ Simulink toolbox contains an automatic connection between Matlab and Simulink environments. Once the user has finished the design of the IT2-FLS via the GUI, it is possible to export his/her design directly to Simulink where an automatic Simulink file will be generated. We believe that the availability of the developed free and open-source TSK IT2-FLS software will be an important step for a wider deployment and development of IT2-FLSs.

The paper organized as follows: Section II briefly presents the IT2-FLSs. Section III presents the developed IT2-FLS toolbox. Section IV presents an illustrative example done with the IT2-FLSs toolbox. Section V presents conclusions and future work.

II. Interval Type-2 Fuzzy Logic Systems

In this section, we will present briefly the internal structure of IT2-FLSs. The internal structure of the IT2-FLS is similar to its T1 counterpart. However, the major differences are that IT2-FLSs employ and use IT2-FSs (rather than T1-FSs) and thus the IT2-FLC has the extra type-reduction process [15-17]. An IT2-FS $(\tilde{X})$ is defined with a type-2 membership function $\mu_{\tilde{X}}(x,u)$ as follows:

$$\tilde{X} = \int_{x\in D_{\tilde{X}}} \int_{u\in J_x} \frac{\mu_{\tilde{X}}(x,u)}{(x,u)} \tag{1}$$

where $\iint$ denotes the union over all admissible x and u, J_x is referred to as the primary membership of x, while $\mu_{\tilde{A}}(x,u)$ is a T1-FS known as the secondary set [15-17]. The uncertainty in the primary membership of a type-2 fuzzy set $\tilde{X}$ is defined by a region named Footprint Of Uncertainty (FOU). The FOU can be described in terms of an Upper Membership Function (UM) $\overline{\mu}_{\tilde{X}}$ and a Lower Membership Function (LMF) $\underline{\mu}_{\tilde{X}}$. If the primary membership J_x is an interval set an IT2-FS is constructed, i.e. $\mu_{\tilde{A}}(x,u) = 1$ for $\forall u \in J_x \subseteq [0,1]$ [15-17].

The block diagram of the IT2-FLS structure is given in Fig. 1. Similar to a T1-FLS, an IT2-FLS includes fuzzifier, rule-base, inference engine, and substitutes the defuzzifier by the output processor comprising a type-reducer and a defuzzifier [15-17]. The generic rule structure of TSK type IT2-FLS constructed from N rules is as follows:

$$R^n\text{: If } x_1 \text{is } \tilde{X}_1^n \text{ and } \ldots \text{ and } x_I \text{is } \tilde{X}_I^n \text{ , Then } y \text{ is } Y^n \tag{2}$$

where $\tilde{X}_i^n$ $(i = 1,\ldots,I)$ are the antecedent IT2-FSs and $Y^n = [\underline{y}^n, \overline{y}^n]$ are the consequent MFs. Here $\underline{y}^n$ and $\overline{y}^n$ can be crisp consequents or linear functions as follows:

$$\begin{aligned} \underline{y}^n &= \underline{a}_1^n x_1 + \ \ldots \ + \ \underline{a}_I^n x_I + \underline{b}^n \\ \overline{y}^n &= \overline{a}_1^n x_1 + \ \ldots \ + \ \overline{a}_I^n x_I + \overline{b}^n \end{aligned} \tag{3}$$

For a crisp input vector $\boldsymbol{x}' = (x_1', x_2', \ldots, x_I')$, the output calculation steps of the IT2-FLS are as below [15-17]:

i. Compute the membership interval of each x_i' on each $\tilde{X}_i^n$, i.e. $\left[\underline{\mu}_{\tilde{X}_i^n}(x_i'), \overline{\mu}_{\tilde{X}_i^n}(x_i')\right], i = 1,2,\ldots I, n = 1,2,\ldots N.$

ii. Calculate the rule firing interval of the n^{th}, $F^n(\boldsymbol{x}')$:

$$F^n(\boldsymbol{x}') \equiv \left[\underline{f}^n, \overline{f}^n\right], n = 1,\ldots,N \tag{4}$$

where

$$\begin{aligned} \underline{f}^n &= [\underline{\mu}_{\tilde{X}_1^n}(x_1') * \ldots \times \underline{\mu}_{\tilde{X}_I^n}(x_I')] \\ \underline{f}^n &= [\overline{\mu}_{\tilde{X}_1^n}(x_1') \times \ldots \times \overline{\mu}_{\tilde{X}_I^n}(x_I')] \end{aligned} \tag{5}$$

iii. Perform TR to combine $F^n(\boldsymbol{x}')$ and the corresponding rule consequents. The most commonly used TR is the center-of-sets type reducer defined as:

$$Y_{cos}(\boldsymbol{x}') = \bigcup_{\substack{f^n\in F^n(\boldsymbol{x}') \\ y^n\in Y^n}} \frac{\sum_{n=1}^N y^n f^n}{\sum_{n=1}^N f^n} = [y_l, y_r] \tag{6}$$

where y_l and y_r are defined as:

$$y_l = \frac{\sum_{n=1}^L \underline{y}^n \overline{f}^n + \sum_{n=L+1}^N \underline{y}^n \underline{f}^n}{\sum_{n=1}^L \overline{f}^n + \sum_{n=L+1}^N \underline{f}^n} \tag{7}$$

$$y_r = \frac{\sum_{n=1}^R \overline{y}^n \underline{f}^n + \sum_{n=R+1}^N \overline{y}^n \overline{f}^n}{\sum_{n=1}^R \underline{f}^n + \sum_{n=R+1}^N \overline{f}^n} \tag{8}$$

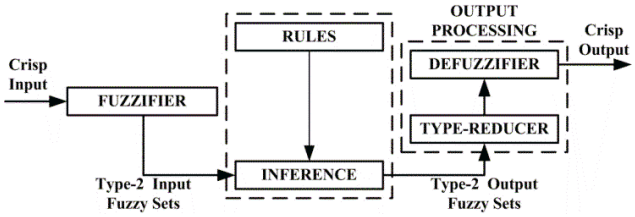
Here, R and L are the switching points that can be found via the iterative KM algorithms or its enhancements [15-17].

iv. Compute the defuzzified (crisp) output as follows:

$$y = \frac{y_l + y_r}{2} \tag{9}$$

Note that, since the iterative KM algorithms have an iterative nature that makes difficult to analyze and design IT2-FLSs, various alternative TR and defuzzification methods have been proposed that have closed-form representation [19], [20].

Fig. 1. Block diagram of an IT2-FLS



III. The IT2-FLS Toolbox

In this section, we will present the fundamental components of the developed IT2-FLS toolbox for Matlab/Simulink. The IT2-FLS toolbox is constructed by reusing the functions of the Matlab® commercial Fuzzy Logic Toolbox, adding new functions for TR operations, developing a user interface, creating a Simulink library and connecting the toolbox to Simulink. The IT2-FLS Matlab/Simulink Toolbox is licensed with GNU general public license version 3 and available for download together with the presented simulation study from "http://web.itu.edu.tr/kumbasart/type2fuzzy.htm".

The heart of the IT2-FLS toolbox is the structure named as the "**.it2fis*" that is created in the Matlab workspace. As

shown in Fig.2, the IT2-FIS includes all the information of the user's design such as MF parameters and types, rules and TR method. Thus, as the user is designing or updating his/her IT2-FLS, the structure is updated automatically.

The developed IT2-FLSs toolbox consists of four main user interfaces which are the Main Editor, MF Editor, Rule Editor and Surface Viewer as shown in Fig.3. It can be seen that the developed user interface is similar to the Matlab® commercial (T1) Fuzzy Logic Toolbox [24]. Thus, a user familiar with Matlab toolbox can easily operate the developed IT2-FLS toolbox.

A. Main Editor

The main editor tab is the entry point of the developed IT2-FLS Toolbox. The other user interfaces of the IT2-FLSs toolbox can be accessed directly from the main editor screen. An overview of the main editor is given in the Fig.3a. In this screen, the basic operations such as saving and loading, exporting to workspace, defining the number of input and output variables, the implication and aggregation operators. The main editor screen provides also two new useful features for the designers which are the "Export to Simulink" and "TR method selector".

The "Export to Simulink" option in the main screen gives the opportunity to export automatically his/her IT2-FLS design to the Matlab/Simulink environment automatically. Thus, the designer can easily perform and analyze various simulation studies directly. In this context, an IT2-FLS Simulink library has been developed. The IT2-FLSs Simulink library is explained in detail in Section IV. It is worth to mention that the Matlab® commercial Fuzzy Logic Toolbox does not have this property.

The IT2-FLS toolbox gives also the designer to select the TR and defuzzification method. In literature, various TR and defuzzification methods have been proposed. It has been shown in [19] that the TR method can directly affect the performance of the IT2-FLSs although it is a structural design parameter like the aggregation and implication operators.

Fig. 2. The schematic diagram of the IT2-FIS of the toolbox

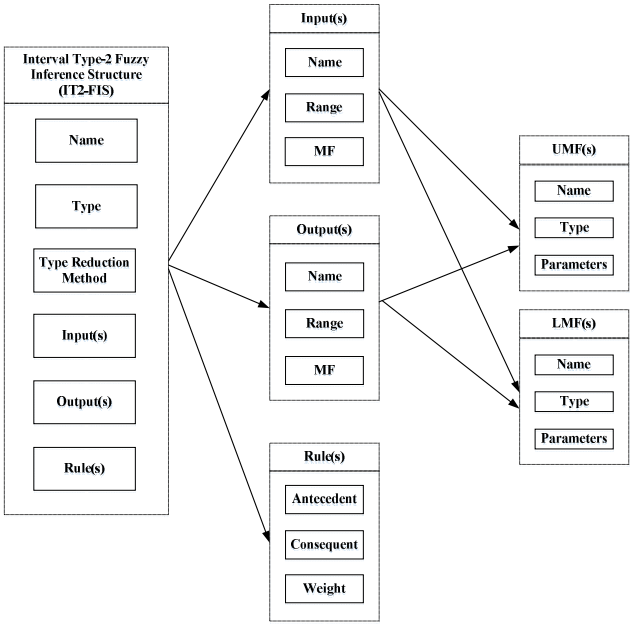
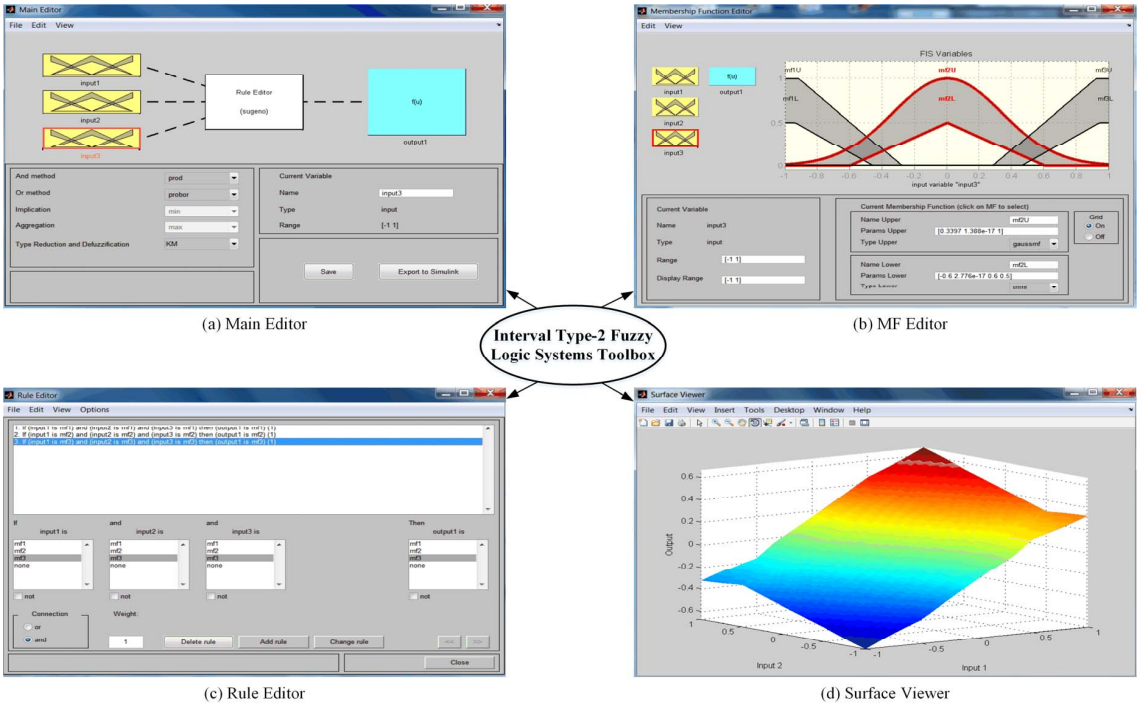


Fig. 3. Overview of the GUI of the IT2-FLS toolbox



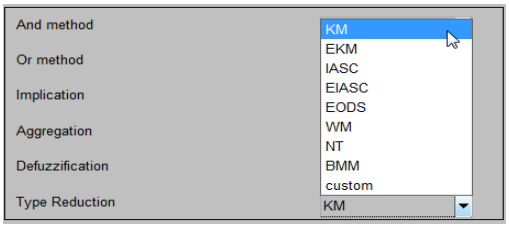
(a) Main Editor (b) MF Editor

(c) Rule Editor (d) Surface Viewer

In this context, we have developed and embedded the most commonly used ones so that the designer can examine the performance of his/her design for various TR and defuzzification methods. In the current version of the IT2-FLS toolbox, the designer can choose various methods simply throughout a pop-up screen as illustrated in Fig. 4. In the current version of the IT2-FLS toolbox, the following TR and defuzzification methods are supported:

1) Karnik-Mendel Algorithm (KM) [16]
2) Enhanced KM Algorithm (EKM) [21]
3) Iterative Algorithm with Stop Condition (IASC) [30], [31]
4) Enhanced IASC (EIASC) [19]
5) Enhanced Opposite Direction Searching Algorithm (EODS) [32], [33]
6) Wu-Mendel Uncertainty Bound Method (WM) [34]
7) Nie-Tan Method (NT) [35]
8) Begian-Melek-Mendel Method (BMM) [36]

Fig. 4. The user interface for the embedded TR Methods



Besides, the IT2-FLS toolbox gives also the opportunity to the designer to use his/her developed TR method by providing a "custom" function option.

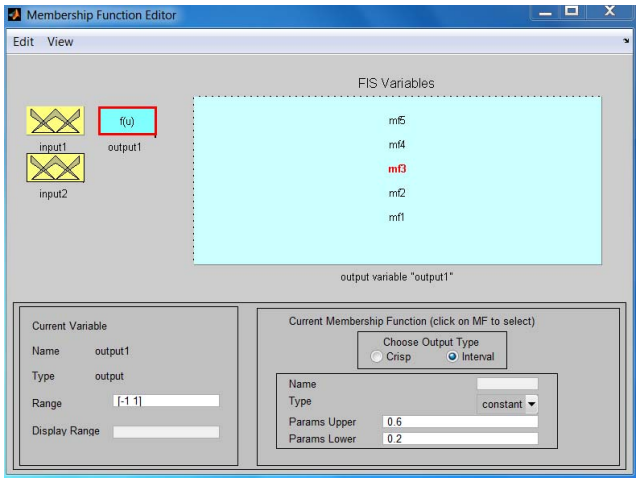
B. *Membership Function Editor*

Membership Function Editor screen can be accessed from the Main Editor Screen by double clicking on of the input or output variables. An overview of the Membership Function Editor page is given in the Fig.3b. Similar to the Matlab Fuzzy Logic Toolbox, the user can define the range, type and value of the LMFs and UMFs of the input and output variables.

In the current version of the IT2-FLS toolbox, the designer can define the antecedent MFs with the MF types that already exist in the Matlab Fuzzy Logic Toolbox (11 different types of MFs). Thus, the user can use and employ the Matlab functions of LMF and UMF in a similar manner. However, the only difference, there is an extra parameter for each type of MFs that defines the height of the corresponding MF. For instance, a triangle MF is defined with the parameters l, c, r, h that define the left point, the center point, right point and the height of the MF, respectively. The parameter h is commonly used to create FOU in the IT2-FSs, especially in IT2 fuzzy controller design [7], [10], [13]. Moreover, designing output MFs via the Membership Function Editor page has some features. As shown in Fig. 5, it is possible to choose the type of the output MFs either as crisp/interval constants or crisp/interval linear functions (as given in Equation (3)).

One of the developed features of the IT2-FLS toolbox is possibility of designing the antecedent UMFs and LMFs in either the same type or different type. The only prerequisite is that the membership degree values of the LMFs have to be smaller than or equal to the UMFs ones, i.e. $\overline{\mu}_{\tilde{X}} \geq \underline{\mu}_{\tilde{X}}$. In Fig. 6, various types of IT2-FSs are illustrated.

Fig. 5. Output Membership Function Editor Screen of the IT2-FLS Toolbox



C. *Rule Editor*

Rule Editor Page is the screen where the users can define or modify the rules of the IT2-FLS. As it can be seen from Fig.3c, we preserved general structure of the Rule editor of the Matlab® commercial Fuzzy Logic Toolbox for an easy implementation. Here, to add a new rule, the desired input and output variables are chosen by clicking and then the new rule is added by clicking the add rule button. It is also possible to modify preexisting rules. All added or modified rule base information are written or updated to the IT2-FIS that is present in the Matlab workspace automatically, and after the rule editor page closed, this information are hold within this structure.

D. *Surface Viewer*

After completing the IT2-FLS design, it is possible to see the surface of the designed IT2-FLS as shown in Fig.3d. It is possible to edit, save the surface and also employ the plot options of Matlab of the designed IT2-FLS.

E. *The Simulink Library of the IT2-FLS Toolbox*

We have also developed an IT2-FLS Simulink library that is also directly connected to the IT2-FLS Matlab toolbox. The IT2-FLSs Simulink library consists of two blocks as shown in Fig.7. Both blocks require an "*.it2fis" structure in Matlab workspace that can be easily created and then exported via the IT2-FLS Matlab toolbox.

The first block is the default library block named "Interval Type-2 Fuzzy Logic Controller" of the IT2-FLSs toolbox. The second library block named as the "Interval Type-2 Fuzzy Logic Controller with TR selection" has a feature to choose a TR method. As shown in Fig. 8, the method can be selected in straight forward manner. If the designer enters "0", the IT2-FLS will run with the TR method that has been chosen in its "*.it2fis" file. This block gives the opportunity to the designer to perform simulation studies of the constructed IT2-FLSs where he/she can examine the performance of IT2-FLSs for various TR and defuzzification methods.

Fig. 6. Input Membership Function Editor Screen of the IT2-FLS Toolbox

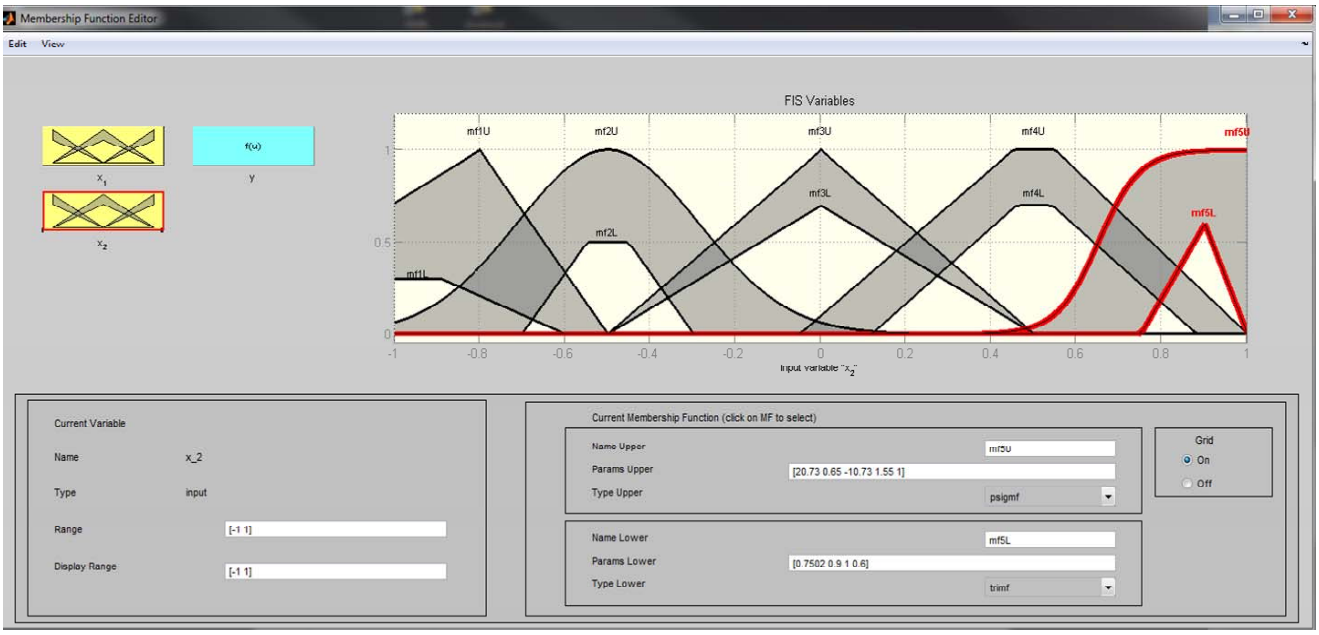


Fig. 7. Simulink Library of the IT2-FLS Toolbox

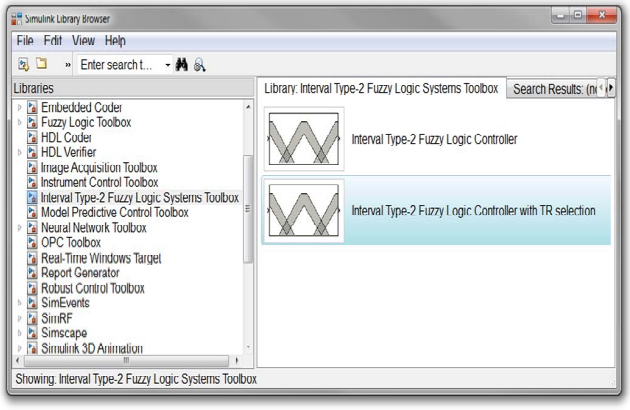
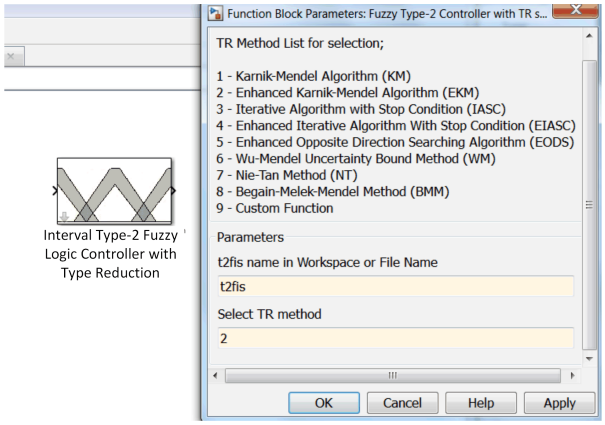
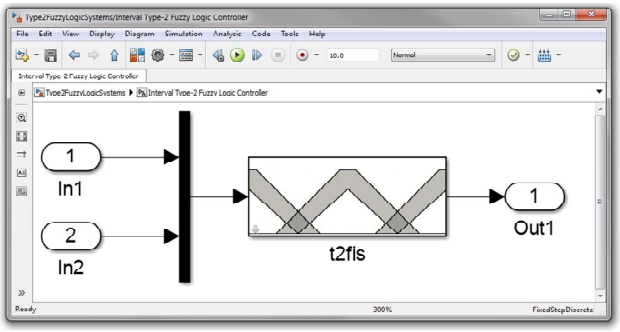


Fig. 8. IT2-FLS Toolbox Library Block with TR Selection



Moreover, as it has been asserted subsection 3.A, the developed Simulink library for the IT2-FLS toolbox also provides a bridge to the IT2-FLS Matlab toolbox. It gives the opportunity to designer for an automatic deployment of his/her design into Matlab/Simulink environment. Once the user has finished the design of the IT2-FLS, it is possible to use this feature by just clicking the button 'Export to Simulink' in the Main Menu. Then, the current design will be exported to Simulink automatically. An example Simulink model created via the IT2-FLS Matlab toolbox is given in the Fig. 9.

Fig. 9. An example Simulink Model that is created via the IT2-FLS Toolbox



IV. Simulation Studies

In this section, we will present several simulations that have been done via the developed IT2-FLS Matlab/ Simulink Toolbox. The simulations were performed on a personal computer with an Intel Core i5-4300U CPU 2.50 GHz processor, 8.00 GB RAM, and software package MATLAB/Simulink 8.0.0.

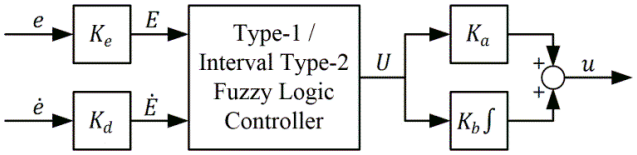
We will compare the performances of the IT2-FPID that employ the KM, EKM, IASC, EIASC, EODS, WM, NT and BMM TR+defuzzification methods. As shown in Fig.10, the

IT2-FPID is constructed by choosing the inputs as the error signal (e) and the change of error signal ($\dot{e}$) while the output is the control signal (u) [10]. Here, K_e, K_d and K_a, K_b are the input and output scaling factors, respectively. The IT2-FLC is constructed with $N = 9$ rules which are given in Table 1. The antecedent MFs are defined with the IT2-FSs shown in Fig 11. The consequent MFs are defined with crisp singletons $NB = -1,\ NM = -0.8,\ Z = 0, PM = 0.8$ and $P = 1$.

TABLE 1. THE RULE BASE OF THE IT2-FPID CONTROLLERS

$\dot{E}/E$	N	Z	P
N	NB	NM	Z
Z	NM	Z	PM
P	Z	NM	PB

Fig. 10. Illustration of the IT2-FPID controller structure

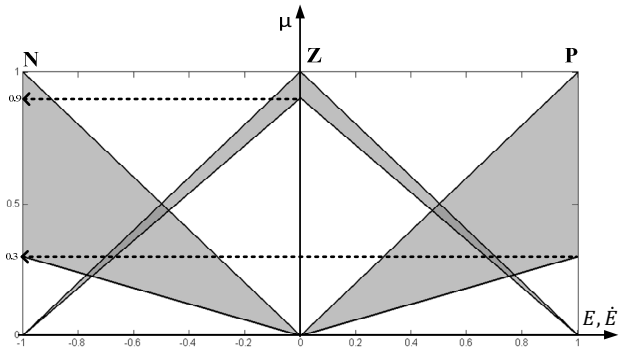


We will examine and compare the transient state response of the IT2-FPID controllers on the following process:

$$G(s) = \frac{K}{Ts+1}e^{-Ls} \qquad (10)$$

where K is the process gain, T is the time constant and L is the time delay. The nominal process parameters are set as $K = 1$, $T = 1$ and $L = 0.2$ (Nominal Process). The robustness against parameter uncertainty of the IT2-FPID controllers is also investigated. In this context, two more parameter settings are considered; $K = 1.3$, $T = 1.9$, $L = 0.4$ (Perturbed Process-1) and $K = 1.1$, $T = 1.3$, $L = 0.45$ (Perturbed Process-2). We will compare the performances of the control systems with respect to their Settling Time (*Ts*), Overshoot (*OS*%) and Integral Time Absolute Error (*IAE*) values. We will also compare their Average (Avg), Maximum (Max) and Minimum (Min) Computational Times (CTs). For a fair comparison and to show the effect of the TR method on the controller performance, we will set and fix the scaling factors of IT2-FPID as $K_e = 1, K_d = 0.5141, K_a = 0.077$ and $K_b = 7.336$. The constructed block diagram via the developed IT2-FLS Matlab/Simulink toolbox is given in Fig.12..

Fig. 11. The Inputs MFs used for the IT2-FPID controller



The performance results of the IT2-FPID control systems for the Nominal and Perturbed Process is given in Fig.13 and Fig.14, respectively. The corresponding performance measures are tabulated in Table 1. It can be clearly seen that the TR method can directly affect the control performance. For instance, from the Nominal Process results, it can be observed that all four IT2-FPID controllers resulted with identical *Ts*. On the other hand, the IT2-FPID-WN was able to reduce the *OS* to zero but resulted with largest *ITAE* value. Similar comments can be also made for the results of the Perturbed Processes. It can be concluded that, although the TR+ defuzzification method is a structural parameter, it has to be determined with respect to the design criteria. For this specific simulation study, we can see that if the reducing the OS value is relatively more important that the WM method should be preferred. Yet, to be able to generalize these observations, extensive comparative simulation and real-time studies must be performed.

Remark: We have not presented the results of the IT2-FPID structures that employ EKM, IASC, EIASC and EODS since they only improve the CT of the original KM method [20]. Thus, their results will be identical to IT2-FPID-KM ones.

Fig. 12. The Simulink block diagram of the IT2-FPID control system

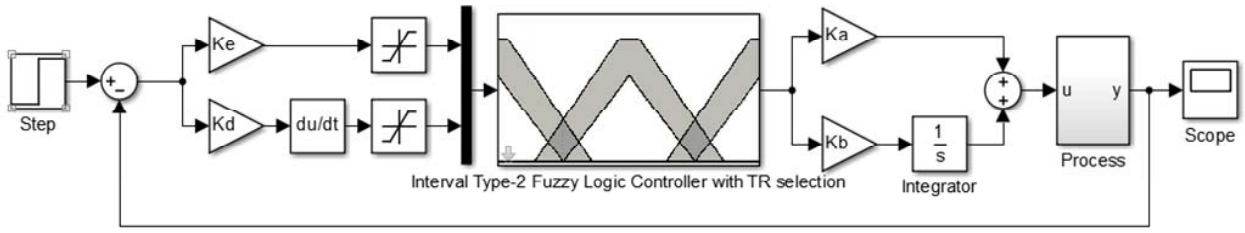


Fig. 13. The closed loop step responses for the Nominal Process.

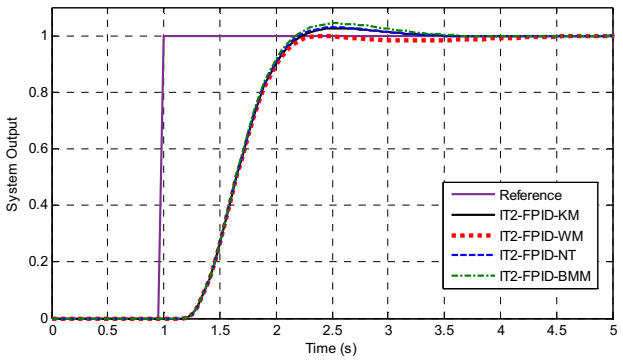


TABLE 2. PERFORMANCE MEASURES OF THE IT2-FPID CONTROLLERS

Process	TR	*Ts*	*OS* (%)	*ITAE*
Nominal Process	KM	1.15	2.6	13.938
	WM	1.15	0.0	14.059
	NT	1.15	2.9	13.840
	BMM	1.15	4.3	14.035
Perturbed Process-1	KM	8.05	61.3	44.737
	WM	10.5	57.0	51.875
	NT	7.95	61.4	45.578
	BMM	8.20	61.7	44.379
Perturbed Process-2	KM	13.30	76.0	63.358
	WM	12.10	71.4	65.594
	NT	14.45	76.3	74.995
	BMM	11.15	79.2	61.389

Moreover, to establish if the TR methods will cause drastic impact on the controllers CT, we have investigated the computation time needed by the IT2-FPID structures to map an input to an output for all possible combinations of the input values in their corresponding universe of discourses, i.e., $e \in [-1,+1]$ and $\dot{e} \in [-1,+1]$ with a discretization size of 0.01. This experiment has been run 500 times and the computational costs of the TR methods are measured by their CTs, obtained from Matlab *tic* and *toc* functions. The obtained Avg, Max and Min CTs are tabulated in Table 3. It can be clearly observed that the WM method resulted with lowest CT values. However, although there is some increase in the CT time of the other methods in comparison to the WM method, the obtained Max CTs are still feasible for various real-time applications.

TABLE 3. COMPARISON OF CTS OF THE TR METHODS

TR	Avg CT (s)	Min CT (s)	Max CT (s)
KM	0.0064	0.0054	0.0215
EKM	0.0060	0.0054	0.0224
IASC	0.0059	0.0053	0.0215
EIASC	0.0058	0.0052	0.0205
EODS	0.0058	0.0053	0.0216
WM	0.0058	0.0051	0.0210
NT	0.0059	0.0051	0.0216
BMM	0.0060	0.0052	0.0226

V. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented an overview of an open source IT2-FLS Matlab/ Simulink toolbox. The toolbox is free of charge for non-commercial purposes and we only ask authors/developers to cite this paper when using the toolbox.

The main goal of this paper is to introduce the research community with a free open source Matlab/Simulink toolbox for the development of IT2-FLSs for a wider accessibility to users beyond the fuzzy logic community. We have reviewed the main features of the current toolkit, including the support for a GUI for an easy design, various IT2-FS constructions, 8 TR methods, a Simulink library, an automatic Simulink file generation via the GUI and visualization features. We have also presented a simulation study which has been conducted via the developed toolbox and analyzed the control system performance of IT2-FPIDs with respect to their employed TR methods. However, it is worth to mention that it is not the purpose of this paper to compare the TR methods but only to demonstrate the features of the Matlab/Simulink toolbox.

We would like to encourage the research community to contribute to the development of the IT2-FLS toolbox through suggestions, comments or contributions. For our future work, we aim to improve the IT2-FLS toolbox by providing support for Mamdani type IT2-FLSs and General Type-2 FLSs; and also integrating a learning technique to develop a toolbox with learning and reasoning capabilities.

ACKNOWLEDGMENT

We would like to state that the IT2-FLS toolbox makes use of the Matlab® commercial Fuzzy Logic Toolbox.

REFERENCES

[1] H. Hagras, "A Hierarchical Type-2 Fuzzy Logic Control Architecture for Autonomous Mobile Robots," IEEE Trans. Fuzzy Syst., vol. 12, no. 4, pp. 524-539, 2004.

[2] Q. Liang and J. Mendel, "Equalization of nonlinear time-varying channels using type-2 fuzzy adaptive filters," IEEE Trans. Fuzzy Syst., vol. 8, no. 5, pp. 551–563, 2000

[3] A. Bilgin, H. Hagras, A. Malibari, M. J. Alhaddad, D. Alghazzawi, "Towards a linear general type-2 fuzzy logic based approach for computing with words," Soft Computing, vol. 17, pp. 2203-2222, 2013.

[4] P. Melin and O. Castillo, "A review on the applications of type-2 fuzzy logic in classification and pattern recognition," Expert Systems with Applications, vol. 40, no. 13, pp 5413–5423, 2013.

[5] Q. Liang, N. Karnik, and J. Mendel, "Connection admission control in ATM networks using survey-based type-2 fuzzy logic systems," IEEE Trans. Syst., Man, Cybern. C, Appl. Rev., vol. 30, pp. 329–339, 2000.

[6] H. Hagras and C. Wagner, "Towards the wide spread use of type-2 fuzzy logic systems in real world applications," IEEE Computational Intell. Mag., vol. 7, no. 3, pp. 14–24, 2012.

[7] B. Yao, H. Hagras, M. Alhaddad, and D. Alghazzawi, "A type-2 fuzzy logic based system for linguistic summarization of video monitoring in indoor intelligent environments," in Proc. IEEE Int. Conf. Fuzzy Systems, Beijing, China, pp. 1–8, , 2014.

[8] H. B. Mitchell, "Pattern recognition using type-II fuzzy sets," Inform. Sci., vol. 170, no. 2–4, pp. 409–418, 2005.

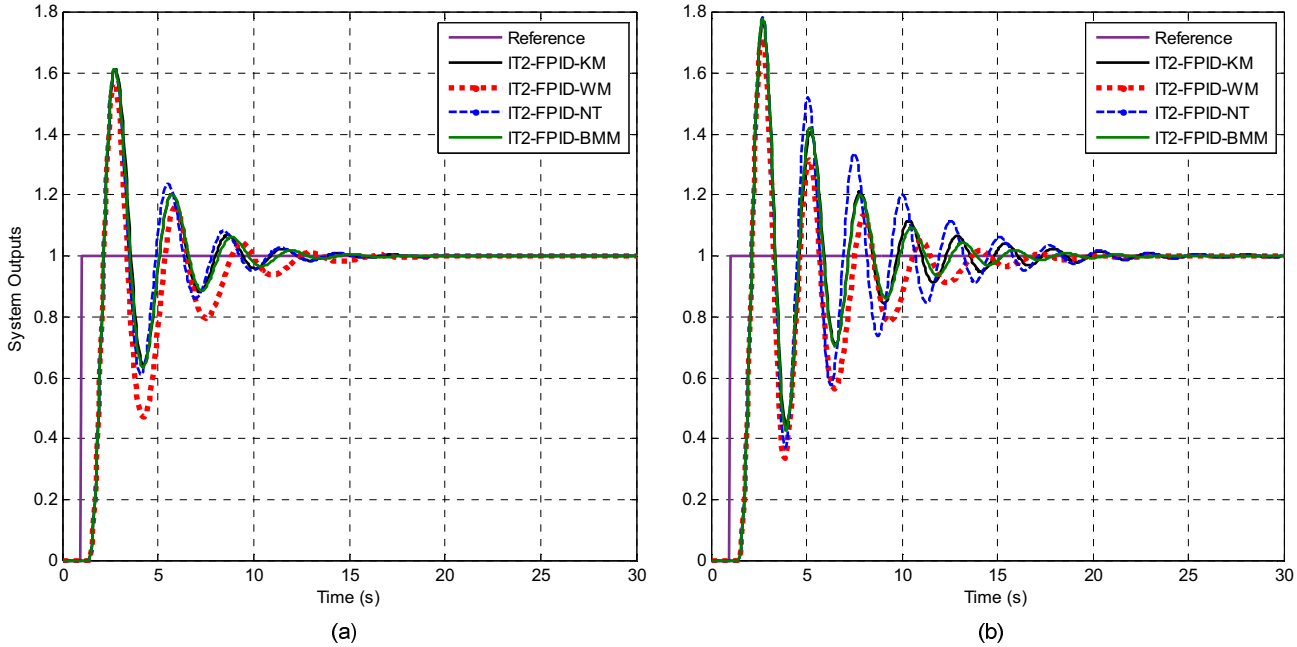
[9] J. Zeng and Z.-Q. Liu, "Type-2 fuzzy Markov random fields and their application to handwritten chinese character recognition," IEEE Trans. Fuzzy Syst., vol. 16, no. 3, pp. 747–760, 2008.

[10] T. Kumbasar and H. Hagras, "An Overview on Interval Type-2 Fuzzy PID Controllers" in the book entitled Springer Handbook of Computational Intelligence, Eds. J. Kacprzyk W. Pedrycz, Springer Verlag, 2015.

[11] L. A. Lucas, T. M. Centeno, and M. R. Delgado, "Land cover classification based on general type-2 fuzzy classifiers," Int. J. Fuzzy Syst., vol. 10, no. 3, pp. 207–216, 2008. R. I. John, P. R. Innocent, and

[12] M. R. Barnes, "Neuro-fuzzy clustering of radiographic tibia image data using type-2 fuzzy sets," Inform. Sci., vol. 125, no. 1–4, pp. 65–82, 2000.

Fig. 14. The closed loop step responses for the (a) Perturbed Process-1 and (b) Perturbed Process-2



[13] T. Kumbasar, "A Simple Design Method for Interval Type-2 Fuzzy PID Controllers," Soft Computing, vol. 18, no. 7, pp. 1293-1304, 2014.

[14] D. Wu, "On the Fundamental Differences between Type-1 and Interval Type-2 Fuzzy Logic Controllers," IEEE Trans. Fuzzy Syst., vol., 10, no.5, pp. 832- 848, 2012.

[15] 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, 2000.

[16] J. M. Mendel, Uncertain rule-based fuzzy logic systems: Introduction and new directions. Prentice Hall PTR, Upper Saddle River, NJ, 2001.

[17] J. M. Mendel and X. Liu, "Simplified Interval Type-2 Fuzzy Logic Systems," IEEE Trans. Fuzzy Syst., vol. 21, no. 6, pp.1056-1069, 2013.

[18] J.M. Mendel and F. Liu, "Super-exponential convergence of the Karnik–Mendel algorithms for computing the centroid of an interval type-2 fuzzy set," IEEE Trans. Fuzzy Syst., vol. 15, no. 2, pp. 309–320, 2007.

[19] D. Wu and M. Nie, "Comparison and practical implementation of type reduction algorithms for type-2 fuzzy sets and systems," in Proc. IEEE Int. Conf. Fuzzy Systems, Taipei, Taiwan, 2011.

[20] D. Wu, "Approaches for reducing the computational cost of interval type-2 fuzzy logic systems: Overview and comparisons," IEEE Trans. Fuzzy Syst., vol. 21, no. 1, pp. 80–99, 2013.

[21] D. Wu and J. M. Mendel, "Enhanced Karnik–Mendel algorithms," IEEE Trans. Fuzzy Syst., vol. 17, no. 4, pp. 923–934, 2009.

[22] J. Alcala-Fdez and J. M. Alonso, "A Survey of Fuzzy Systems Software: Taxonomy, Current Research Trends and Prospects," IEEE Trans. Fuzzy Syst., 2015.

[23] C. Wagner, S. Miller, J.M. Garibaldi, "A fuzzy toolbox for the R programming language," in Proc. IEEE Int. Conf. Fuzzy Systems, pp.1185-1192, 2011, Taipei, Taiwan.

[24] MATLAB® Fuzzy Logic Tool™ 2 Users Guide, The MathWorks, Inc., Natick, USA, 2010.

[25] C. Wagner, "Juzzy-A Java based toolkit for Type-2 Fuzzy Logic Proc. Int. Conference on Fuzzy Systems, pp. 45–52,

[26] O. Castillo, P. Melin, and J. R. Castro, "Computational intelligence software for interval type-2 fuzzy logic," Computer Applications in Engineering Education, vol. 21, no. 4, pp. 737–747, 2013.

[27] J. R. Castro, O. Castillo, P. Melin, A. R. Diaz, "Building Fuzzy Inference Systems with a New Interval Type-2 Fuzzy Logic Toolbox", Transactions on Computational Science, pp. 104-114, 2008.

[28] M. B. Ozek and Z. H. Akpolat, "A software tool: Type-2 fuzzy logic toolbox," Computer Applications in Engineering Education, vol. 16, no. 2, pp. 137–146, 2008.

[29] M. Castanon-Puga, J. Castro, M. Flores-Parra, C. Gaxiola- Pacheco, L. Martiınez-Mendez and L. Palafox-Maestre, "JT2FIS ´ A Java Type-2 Fuzzy Inference Systems Class Library for Building Object-Oriented Intelligent Applications," in Advances in Soft Computing and Its Applications, ser. Lecture Notes in Computer Science, vol. 8266. Berlin Heidelberg: Springer, pp. 204–215, 2013

[30] K. Duran, H. Bernal, and M. Melgarejo, "Improved iterative algorithm for computing the generalized centroid of an interval type-2 fuzzy set," in Proc. Annual Meeting of the North American Fuzzy Information Processing Society, New York, pp. 1–5, 2008.

[31] M. Melgarejo, "A fast recursive method to compute the generalized centroid of an interval type-2 fuzzy set," in Proc. Annual Meeting of the North American Fuzzy Information Processing Society, San Diego, CA, pp. 190–194, 2007.

[32] H. Z. Hu, G. Zhao, and H. N. Yang, "Fast algorithm to calculate generalized centroid of interval type-2 fuzzy set," Control Decis., vol. 25, no. 4, pp. 637–640, 2010.

[33] H. Hu, Y.Wang, and Y. Cai, "Advantages of the enhanced opposite direction searching algorithm for computing the centroid of an interval type-2 fuzzy set," Asian J. Control, vol. 14, no. 6, pp. 1–9, 2012.

[34] H.Wu and J. M. Mendel, "Uncertainty bounds and their use in the design of interval type-2 fuzzy logic systems," IEEE Trans. Fuzzy Syst., vol. 10, no. 5, pp. 622–639, 2002.

[35] M. Nie and W. W. Tan, "Towards an efficient type-reduction method for interval type-2 fuzzy logic systems," in Proc. IEEE Int. Conf. Fuzzy Systems, 2008, pp. 1425–1432.

[36] M. Begian, W. Melek, J. Mendel, Stability analysis of type-2 fuzzy systems, in Proc. IEEE Int. Conf. Fuzzy Systems, Hong Kong, pp. 947–953, 2008.