JT2FIS: Java Type-2 Fuzzy Inference System
An Object-oriented Class Library for Building Java Intelligent Applications

Manuel Castañón-Puga, Juan-Ramón Castro and Miguel Flores-Parra
Facultad de Ciencias Químicas e Ingeniería, Universidad Autónoma de Baja California, Tijuana, México

Keywords: Computational Intelligence, Type-2 Fuzzy Inference System, Java Fuzzy Systems Class Library.

Abstract: This paper introduces a Java® class library for fuzzy systems that can be used to build an Interval Type-2 Fuzzy Inference System. Architecture of the system is presented and object oriented design and modules are described. We use the Water Temperature and Flow Controller as a classic example to benchmark the inputs and outputs and to show how to use it on engineering. We compare the developed library with an existing Matlab® library in order to discuss the Java® object oriented approach and its applications.

1 INTRODUCTION

Fuzzy inference systems (FIS) have been broadly used for a wide engineering applications. Especially, FIS have been successfully applied in control systems and decision-making systems and the main advantage is the way to deal with imprecise information about some system variable and let us to work with.

The most of the FIS used until now are FIS based on a Type-1 model (Zadeh, 1965), but lately, a Type-2 model have been developed and others applications are been extended with it (Zadeh, 1973). The state of the art led us to Type-2 General Fuzzy Inference Model (Lucas, 2007) that have been developed as a next step on the way to have Fuzzy Inference Systems with more capability to model real-world things (Castillo and Melin, 2008)(Castillo et al., 2010).

The purpose of this paper is to introduce a Java class library for fuzzy systems that can be used to build an Interval Type-2 Fuzzy Inference System.

1.1 Type-2 Fuzzy Inference System

A fuzzy inference system (FIS) is based on logical rules that can work with numeric values or fuzzy input, rules are evaluated and the individual results together to form what is the output fuzzy, then, a numerical value must be passed through a process of defuzzification if its required. Figure 1 shows a block diagram of the classic structure of a FIS.

The concept of a type-2 fuzzy set was introduced by Zadeh (Zadeh, 1975) as an extension of the concept of fuzzy sets usually type 1. A type-2 fuzzy set is

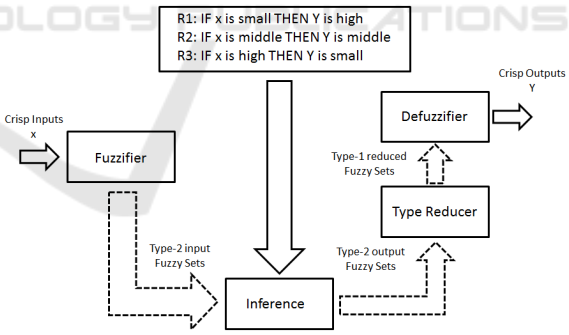

Figure 1: Type-2 Fuzzy Inference System block diagram.

characterized by a membership function whose membership value for each element of the universe is a membership function in the range [0, 1], unlike the type-1 fuzzy sets where the value of membership is a numeric value in the range [0, 1]. The creation of a fuzzy set depends on two aspects: the identification of a universe of appropriate values and specifying a membership function properly. The choice of membership function is a subjective process, meaning that different people can reach different conclusions on the same concept. This subjectivity is derived from individual differences in the perception and expression of abstract concepts and very little to do with randomness. Therefore, subjectivity and randomness of a fuzzy set is the main difference between the study of fuzzy sets and probability theory (Jang et al., 1997).

In type-1 fuzzy sets, once the membership function defined for a concept, this is based on the subjective opinion of one or more individuals and shows no more than one value for each element of the uni-

Castañón-Puga M., Castro J. and Flores-Parra M..
JT2FIS: Java Type-2 Fuzzy Inference System - An Object-oriented Class Library for Building Java Intelligent Applications.
DOI: 10.5220/0004569805240529
In *Proceedings of the 15th International Conference on Enterprise Information Systems* (ICEIS-2013), pages 524-529
ISBN: 978-989-8565-59-4

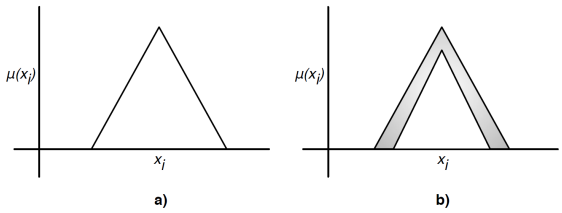

Figure 2: Type-1 fuzzy set and type-2 fuzzy set with uncertainty.

verse. In doing so, it lose some of the ambiguity that some concepts are discussed, especially where people may have a slightly different opinion and all are considered valid. The type-2 fuzzy sets allow handling linguistic and numerical uncertainties. Figure 2 depicts two graphics of fuzzy sets, a) with type-1 fuzzy logic, and b) with type-2 fuzzy logic.

In a) the values set shown is $A = \{(x, \mu_A(x)) | x \in X\}$ where $A \subseteq X$, X $A \subseteq X$, X is the universe of valid values and $\mu_A(x)$ is the membership function (MF) that contains a map of each value of X with its membership value corresponding to a value between 0 and 1. For b) the values set is $\tilde{A} = \{((x,u), \mu_{\tilde{A}}(x,u))\}$, where MF $\mu_{\tilde{A}}(x,u)$ has a membership value for each element of the universe as a function of membership in the range [0, 1], so the footprint can be seen around the curve of a).

1.2 Type-2 Fuzzy Inference System Applications

Concepts such as large/small or fast/slow can be represented as fuzzy sets, thus allowing that there may be slight variations in the definition for common concepts, an example of this can be shown in (Wagner and Hagras, 2010). When dealing with a fixed set of actions but with different criteria to decide what action to take, you can use fuzzy logic as a viable option to define the behavior profiles. There are many applications where fuzzy logic has been used, for example, a simulation of the bird age-structured population growth based on an interval type-2 fuzzy cellular structure (Leal-Ramírez et al., 2011), optimization of interval type-2 fuzzy logic controllers using evolutionary algorithms (Castillo et al., 2011), an improved method for edge detection based on interval type-2 fuzzy logic (Melin et al., 2010), a hybrid learning algorithm for a class of interval type-2 fuzzy neural networks (Castro et al., 2009), a systematic design of a stable type-2 fuzzy logic controller (Castillo and Melin, 2008), and an efficient computational method to implement type-2 fuzzy logic in control applications (Sepúlveda et al., 2007).

1.3 Object Oriented Fuzzy Inference Systems

There are available code libraries and tool-kits to build Fuzzy Inference Systems. Some of this packages are object oriented class libraries that are developed mainly for build Type-1 Fuzzy Logic with an object oriented programming language (García-Valdez et al., 2007). jFuzzyLogic (Cingolani and Alcala-Fdez, 2012) is an example of a class library written in Java The advantage of an Fuzzy Inference System in Java is that we could build intelligent systems with Type-2 Fuzzy Logic capabilities using a object oriented programming language. Java is a robust general use object oriented programming language used in a wide range of applications.

2 JT2FIS ARCHITECTURE

JT2FIS is a class library developed for Java. The main purpose is to deploy a library for build type-2 fuzzy inference systems with a object oriented programming language.

A Java library turns important due is very convenient for reuse and legibility of coding, and system portability.

2.1 JT2FIS Design

JT2FIS is integrated by a package structure that contains all classes collection. The package containment are organized and depicted in Table 1.

The library takes advantage of heritage capability of object-oriented paradigm to integrate new membership features. We can see this approach in fig. 3 where MemeberFunction are an abstract class that let us to extends as member functions as we require.

In JT2FIS version 1.0 we have seven different type-2 member functions and a core of type-1 member functions available (Gauss, Triangular, Trapezoidal). Table 2 list type-2 member function available in the library.

The Figure 4 depicts JT2FIS expressed in Unified Modeling Language (UML). The class structure JT2FIS are shown.

Fis classes are composed by a set of inputs (Input class), outputs (Output class) and rules (Rule class). Each input and output contain a set off member functions (MemberFunction class) and each one could be a specific type of member function.

The Fis class provides the main features of an Interval Type-2 Fuzzy Inference System. Fis class is an abstract class that establishes the core functionality

Table 1: JT2FIS library packages content.

Package	Content
Fis	Fis Class Mamdani Class Defuzzifier Package Fuzzifier Package MF Package
Defuzzifier	Defuzzy Class TypeReducer Package
TypeReducer	TypeReducer Class
Fuzzifier	Fuzzify Class
Mf	MemberFunction Class type1 Package type2 Package
Type1	BellMemberFunction Class GaussMemberFunction Class TrapezoidalMemberFunction Class TriangulerMemberFunction Class
Type2	GaussCutMemberFunction Class GaussUncertaintyMeanMemberFunction Class GaussUncertaintyStandardDesviationMemberFunction Class TrapaUncertaintyMemberFunction Class TrapezoidalUncertaintyMemberFunction Class TriangulerUncertaintyMemberFunction Class TriUncertaintyMemberFunction Class
Structure	Input Class Output Class Rule Class

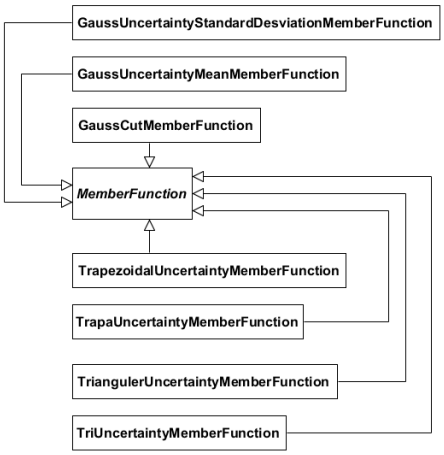

Figure 3: Member Functions Types.

and extends to Mamdani and Sugeno concrete classes that implements different approaches.

We can expand member function types extending the core class MemberFunction class. Figure 5 show actual member function available.

Table 2: JIT2FIS Type-2 Member Functions.

PackageType-2 Member Functions	Description
GaussCutMemberFunction	Params =[sigma mean alfa]
GaussUncertaintyMeanMemberFunction	Params=[sigma mean1 mean2]
GaussUncertaintyStandardDesviationMemberFunction	Params=[sigma1 sigma2 mean]
TrapaUncertaintyMemberFunction	Params=[a1 b1 c1 d1 a2 b2 c2 d2 alfa]
TrapezoidalUncertaintyMemberFunction	Params=[a d sa sd sn alfa]
TriangulerUncertaintyMemberFunction	Params=[a c sa sc]
TriUncertaintyMemberFunction	Params=[a1 b1 c1 a2 b2 c2]

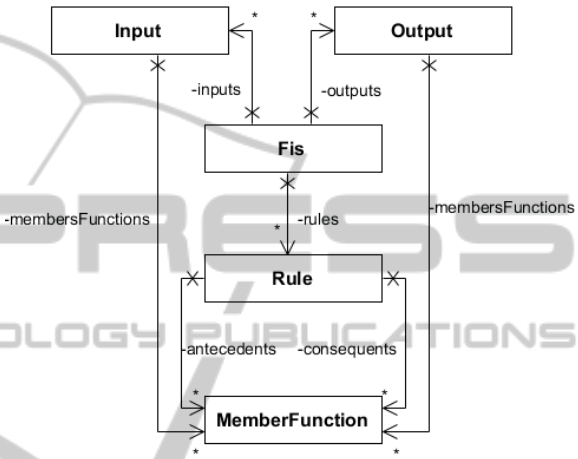

Figure 4: JT2FIS core class structure.

2.2 Building a Interval Type-2 Fuzzy Inference System

In order to create a FIS with JT2FIS in the next example, we are considering the following procedure:

1. Create a new instance of FIS class.
2. Create and configure new inputs instances of Input class to add to the FIS.
3. Create and configure new outputs instances of Output class to add to the FIS.
4. Create and configure new rules instances of Rule class to add to the FIS.
5. Evaluate inference with build FIS.

2.2.1 Creating a New FIS Instance

To build an Interval Type-2 Fuzzy Inference System, first we have to create an instance of Mamdani class. Listing 1 shows how to do that.

Listing 1: Creating a new instance of FIS.

```
Mamdani fis = new Mamdani("FIS");
```

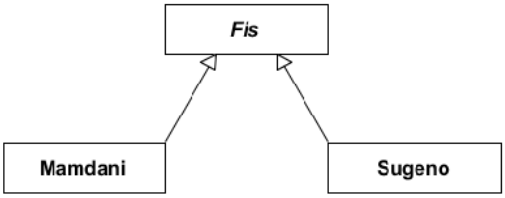

Figure 5: Fuzzy Inference Systems Types.

2.2.2 Adding inputs to FIS

Second, we can add inputs creating instances of Input class. For each input instance, we can add a member function that will represents a linguistic input variable in the system. Member functions could be from different type, depending of the application.

Listing 2 shows how to add member function to an input, and how to add an input to fis.

Listing 2: Adding a new input of FIS.

```
//Creating a new Input instance
Input input =new Input("input");
//Configuring input ranges
input.setLowerRange(-20.0);
input.setSuperiorRange(20.0);
//Creating a input member function
GaussUncertaintyMeanMemberFunction
    inputMF = new
    GaussUncertaintyMeanMemberFunction(
    "InputMF");
//Configuring input member function
inputMF.setSigma(-6.0);
inputMF.setMean1(-12.0);
inputMF.setMean2(-18.0);
//Adding member function to input
input.getMemberFunctions().add(inputMF
    );
//Adding input to fis
fis.getInputs().add(input);
```

2.2.3 Adding outputs to FIS

Third, we can add outputs creating instances of Output class. In the same way the inputs, for each output instance, we can add a member function that will represents a linguistic output variable in the system. Member functions could be from different type, depending of the application.

Listing 3 shows how to add member function to an output, and how to add an output to fis.

Listing 3: Adding a new output of FIS.

```
//Creating a new Output instance
Output output=new Output("output");
// Output ranges
output.setLowerRange(-1.0);
output.setSuperiorRange(1.0);
//Creating a output member function
GaussUncertaintyMeanMemberFunction
    outputMF=new
    GaussUncertaintyMeanMemberFunction(
    "OutputMF");
//Configuring output member function
outputMF.setSigma(-1.05);
outputMF.setMean1(-0.65);
outputMF.setMean2(-0.35);
//Adding member function to output
output.getMemberFunctions().add(
    outputMF);
//Adding output to fis
fis.getOutputs().add(output);
```

2.2.4 Adding rules to FIS

Finally, we can add rules to fis. Each rule is composed by a set of antecedents and a set of consequents. Antecedents and consequents are member functions that are part of inputs and outputs.

Listing 4 shows how to add antecedents and consequents to rule, and how to add a rule to fis.

Listing 4: Adding a new rule of FIS.

```
//Creating a new Rule instance
Rule rule =new Rule();
//Adding antecedent to rule
MemberFunction antecedent = fis.
    getInputs().get(0).
    getMemberFunctions().get(0);
rule.getAntecedents().add(
    antecedent);
//Adding consecuent to rule
MemberFunction concecuent = fis.
    getOutputs().get(0).
    getMemberFunctions().get(0);
rule1.getConsequents().add(
    concecuent);
//Adding rule to fis
fis.getRules().add(rule);
```

2.2.5 Evaluating inference with FIS

Once the system is built, we can use it to evaluate input and output values. JT2FIS have 5 different ways to evaluate fis. Each way is a method of reduction (Centroid, Center-of-Sums, Height, Modified Height, Center of Sets). Each method has parameters that must be configured depending of representation needs.

Listing 5 shows how to configure and evaluate the fis.

Listing 5: Evaluating inference with FIS.

```
//Configuring evaluation parameters
```

```
int points=101;
int andMethod=0;
int orMethod=0;
int impMethod=0;
int aggMethod=0;
//Creating a new inputStack of
    inputList
ArrayList<ArrayList<Double>>
    inputStack = new ArrayList<
    ArrayList<Double>>();
//Creating an input list
ArrayList<Double> inputList=space.
    getLinspace(-20, 20, points);
//Adding input list to inputStack
inputStack.add(inputList);
//Evaluating inputStack with
    Singleton-Centroid method on fis
fis.evaluateFisSingletonCentroid(
    inputStack, points, andMethod,
    orMethod, impMethod, aggMethod);
```

3 WATER TEMPERATURE AND FLOW CONTROLLER TEST CASE

To test the JT2FIS we going to use Water Temperature and Flow Controller case. Water Temperature and Flow Controller is a proposed problem that is provided by Matlab as a example to show how to use Fuzzy Logic Toolbox. Castro et. al. (Castro et al., 2007) extends Matlab Toolbox to Type-2 Fuzzy Logic, so we going to use this toolbox to compare our approach with it.

We configured in the same way the Type-2 Fuzzy Inference System using JT2FIS and Matlab Interval Type-2 Fuzzy Toolbox.

The system implements the following rules:

1. If (Temp is Cold) and (Flow is Soft) then (Cold is openSlow)(Hot is openFast)
2. If (Temp is Cold) and (Flow is Good) then (Cold is closeSlow)(Hot is openSlow)
3. If (Temp is Cold) and (Flow is Hard) then (Cold is closeFast)(Hot is closeSlow)
4. If (Temp is Good) and (Flow is Soft) then (Cold is openSlow)(Hot is openSlow)
5. If (Temp is Good) and (Flow is Good) then (Cold is Steady)(Hot is Steady)
6. If (Temp is Good) and (Flow is Hard) then (Cold is closeSlow)(Hot is closeSlow)
7. If (Temp is Hot) and (Flow is Soft) then (Cold is openFast)(Hot is openSlow)

Table 3: Inputs, Outputs example "ISHOWER".

Type	MemberFunction	Params
Input1	TrapaUncertaintyMemberFunction	a1=-31,b1=-31,c1=-16,d1=-1, a2=-29,b2=-29,c2=-14,d2=1.0,alfa=0.98
Input1	TriUncertaintyMemberFunction	a1=-11,b1=-1,c1=9,a2=-9,b2=1,c2=11
Input1	TrapaUncertaintyMemberFunction	a1=-1,b1=14,c1=29,d1=29, a2=1,b2=16,c2=31,d2=31,alfa=0.98
Input2	TrapaUncertaintyMemberFunction	a1=-3.1,b1=-3.1,c1=-0.9,d1=-0.1, a2=-2.9,b2=-2.9,c2=-0.7,d2=0.1,alfa=0.98
Input2	TriUncertaintyMemberFunction	a1=-0.45,b1=-0.05,c1=0.35,a2=-0.35,b2=0.05,c2=0.45
Input2	TrapaUncertaintyMemberFunction	a1=-0.1,b1=0.7,c1=2.9,d1=0.1, a2=0.9,b2=3.1,c2=3.1,d2=0.1,alfa=0.98
Output1	TriUncertaintyMemberFunction	a1=-1.05,b1=-0.65,c1=-0.35,a2=-0.95,b2=-0.55,c2=-0.25
Output1	TriUncertaintyMemberFunction	a1=-0.65,b1=-0.35,c1=-0.05,a2=-0.55,b2=-0.25,c2=0.05
Output1	TriUncertaintyMemberFunction	a1=-0.35,b1=-0.05,c1=0.25,a2=-0.25,b2=0.05,c2=0.35
Output1	TriUncertaintyMemberFunction	a1=-0.05,b1=0.25,c1=0.55,a2=0.05,b2=0.35,c2=0.65
Output1	TriUncertaintyMemberFunction	a1=0.25,b1=0.55,c1=0.95,a2=0.35,b2=0.65,c2=1.05
Output2	TriUncertaintyMemberFunction	a1=-1.05,b1=-0.65,c1=-0.35,a2=-0.95,b2=-0.55,c2=-0.25
Output2	TriUncertaintyMemberFunction	a1=-0.65,b1=-0.35,c1=-0.05,a2=-0.55,b2=-0.25,c2=0.05
Output2	TriUncertaintyMemberFunction	a1=-0.35,b1=-0.05,c1=0.25,a2=-0.25,b2=0.05,c2=0.35
Output2	TriUncertaintyMemberFunction	a1=-0.05,b1=0.25,c1=0.55,a2=0.05,b2=0.35,c2=0.65
Output2	TriUncertaintyMemberFunction	a1=0.25,b1=0.55,c1=0.95,a2=0.35,b2=0.65,c2=1.05

8. If (Temp is Hot) and (Flow is Good) then (Cold is openSlow)(Hot is closeSlow)
9. If (Temp is Hot)and (Flow is Hard) then (Cold is closeSlow)(Hot is closeFast)

3.1 JT2FIS Shower System Test Case Results

With previous fuzzy inference system configuration, and using 101 points and Centroid reduction type, we evaluate Water Temperature and Flow Controller in JT2FIS and Matlab Interval Type-2 Fuzzy Toolbox implementations. Table 4 show no difference between tools obtaining the same response.

Table 5 show comparative performance between tools with different discretizations points for input1=20 and input2=1.

4 CONCLUSIONS

JT2FIS is an Object-Oriented Class Library for Building Java Intelligent Applications using Java Interval Type-2 Fuzzy Inference System. We present architecture and object oriented design of a JT2FIS class library. We provide an example of how to create and configure an Interval Type-2 Fuzzy Inference System in Java and we show a Water Temperature and

Table 4: Comparing JT2FIS outputs versus Matlab® Interval Type-2 Fuzzy Logic Toolbox outputs.

Inputs		JT2FIS			Interval Type-2 Fuzzy Logic Toolbox		
x1	x2	Output 1	Output 2	Time(ms)	Output 1	Output 2	Time(ms)
-16	-0.8	0.3	0.6328	1	0.3	0.6328	8.6
-12	-0.6	0.3	0.6342	1	0.3	0.6342	8.6
-8	-0.4	0.2211	0.5184	1	0.2211	0.5184	8.6
-4	-0.2	-0.0098	0.2545	1	-0.0098	0.2545	8.6
0	0	0	0	1	0	0	8.6
4	0.2	0.0098	-0.2545	1	0.0098	-0.2545	8.6
8	0.4	-0.22113	-0.5184	1	-0.22113	-0.5184	8.6
12	0.6	-0.2999	-0.6342	1	-0.2999	-0.6342	8.6
16	0.8	-0.3	-0.6328	1	-0.3	-0.6328	8.6
20	1	-0.3	-0.6328	1	-0.3	-0.6328	8.6

Table 5: Comparing JT2FIS times(ms) versus Matlab® Interval Type-2 Fuzzy Logic Toolbox times(ms) with different discretizations points for input1=20 and input2=1.

Number Points	JT2FIS Time(ms)	Interval Type-2 Fuzzy Logic Toolbox Time(ms)
101	1	8.6
1001	8	13.7
10001	426	71.2

Flow Controller case of study to compare outputs response between JT2FIS and Interval Type-2 Fuzzy Logic Toolbox in Matlab.

ACKNOWLEDGEMENTS

The authors would like to thank to Universidad Autónoma de Baja California (UABC) for grant this project.

REFERENCES

Castillo, O., Melin, P., A.-G. A., Montiel, O., and Sepúlveda, R. (2011). Optimization of interval type-2 fuzzy logic controllers using evolutionary algorithms. *Soft Computing*, 15(6):1145–1160.

Castillo, O. and Melin, P. (2008). Type-2 fuzzy logic. In *Type-2 Fuzzy Logic: Theory and Applications*, volume 223 of *Studies in Fuzziness and Soft Computing*, pages 29–43. Springer Berlin Heidelberg.

Castillo, O., Melin, P., and Castro, J. R. (2010). Computational intelligence software for interval type–2 fuzzy logic. *Journal Computer Applications in Engineering Education*.

Castro, J., Castillo, O., and Martinez, L. (2007). Interval type-2 fuzzy logic toolbox. *Engineering Letters*, 15(1).

Castro, J., Castillo, O., Melin, P., and Rodríguez-Díaz (2009). A hybrid learning algorithm for a class of interval type-2 fuzzy neural networks. *Inf. Sci.*, 179(13):2175–2193.

Cingolani, P. and Alcala-Fdez, J. (2012). jFuzzyLogic: a robust and flexible Fuzzy-Logic inference system language implementation. *2012 IEEE International Conference on Fuzzy Systems*, pages 1–8.

García-Valdez, Licea-Sandoval, G., Alaníz-Garza, A., and Castillo, O. (2007). Object oriented design and implementation of an inference engine for fuzzy systems. *Engineering Notes*, 15(1).

Jang, J., Sun, C., and Mizutani, E. (1997). *Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence*. MATLAB Curriculum Series. Prentice Hall, Upper Saddle River, NJ.

Leal-Ramírez, C., Castillo, O., Melin, P., and Rodríguez-Díaz (2011). Simulation of the bird age-structured population growth based on an interval type-2 fuzzy cellular structure. *Inf. Sci.*, 181(3):519–535.

Lucas, L. (2007). General Type-2 Fuzzy Inference Systems: Analysis, Design and Computational Aspects. *Fuzzy Systems . . .* , (5).

Melin, P., Mendoza, O., and Castillo, O. (2010). An improved method for edge detection based on interval type-2 fuzzy logic. *Expert Systems Applications*, 37(12):8527–8535.

Sepúlveda, R., Castillo, O., Melin, P., and Montiel, O. (2007). *Analysis and Design of Intelligent Systems using Soft Computing Techniques*, chapter An Efficient Computational Method to Implement Type-2 Fuzzy Logic in Control Applications, pages 45–52.

Wagner, C. and Hagras, H. (2010). Fuzzy composite concepts based on human reasoning. In *IEEE International Conference on Information Reuse and Integration (IRI 2010)*, Las Vegas, Nevada, USA.

Zadeh, L. (1975). The concept of a linguistic variable and its application to approximate reasoning. *Information Science*, 8(199249):301–357.

Zadeh, L. A. (1965). Fuzzy Sets. *Information and Control*, 8:338–353.

Zadeh, L. a. (1973). Outline of a New Approach to the Analysis of Complex Systems and Decision Processes. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-3(1):28–44.