Interval Type-2 Fuzzy Logic Applications

Oscar Castillo and Patricia Melin

Tijuana Institute of Technology,
Division of Graduate Studies,
Tijuana, Mexico
{ocastillo,epmelin}@hafsamx.org

Abstract. In this chapter three applications of interval type-2 fuzzy logic are considered. First, we consider the use of interval type-2 fuzzy systems in conjunction with modular neural networks for image recognition. A type-2 fuzzy system is used for feature extraction in the training data, and another type-2 fuzzy system is used to find the optimal parameters for the integration method of the modular neural network. Type-2 Fuzzy Logic is shown to be a tool to help improve the results of a neural system by facilitating the representation of the human perception. The second application involves edge detection in digital images, which is a problem that has been solved by means of the application of different techniques from digital signal processing, and also the combination of some of these techniques with type-1 fuzzy systems have been proposed. In this chapter a new interval type-2 fuzzy method is implemented for the detection of edges and the results of three different techniques for the same goal are compared. The third application, concerns the problem of stability, which is one of the more important aspects in the traditional knowledge of Automatic Control. Interval type-2 fuzzy logic is an emerging and promising area for achieving intelligent control (in this case, Fuzzy Control). In this chapter we use the Fuzzy Lyapunov Synthesis, as proposed by Margaliot, to build a Lyapunov stable type-1 fuzzy logic control system, and then we make an extension from a type-1 to a type-2 fuzzy controller, ensuring the stability on the control system and proving the robustness of the corresponding fuzzy controller.

1 Interval Type-2 Fuzzy Logic for Image Recognition

At the moment, many methods for image recognition are available. But most of them include a phase of feature extraction or another type of preprocessing closely related to the type of image to recognize (Melin and Castillo, 2005) (Chuang et al., 2000). The method proposed in this paper can be applied to any type of images, because the preprocessing phase does not need specific data about the type of image (Melin et al., 2007) (Mendoza and Melin, 2007).

Even if the method was not designed only for face recognition, we have made the tests with the ORL face database (AT&T Laboratories Cambridge) composed of 400 images of size 112x92. There are 40 persons, with 10 images of each person. The images are taken at different times, lighting and facial expressions. The faces are in upright position of frontal view, with slight left-right rotation. Figure 1 shows the 10 samples of one person in ORL database. To explain the proposed steps of the method, we need to separate it them in two phases: the training phase in figure 3 and the recognition phase in figure 4.

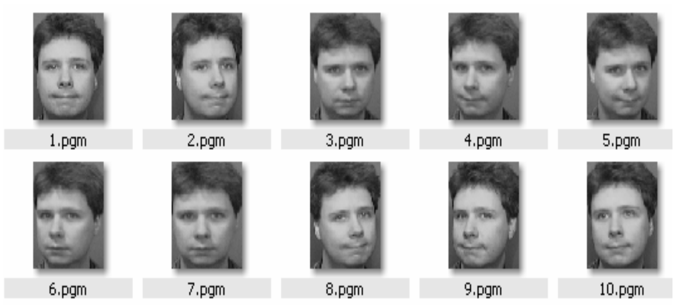


Fig. 1. Set of 10 samples of a person in ORL

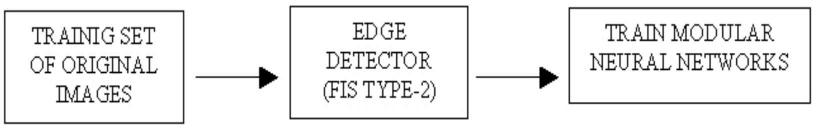

TRAINIG SET OF ORIGINAL IMAGES
EDGE DETECTOR (FIS TYPE-2)
TRAIN MODULAR NEURAL NETWORKS

Fig. 2. Steps in Training Phase

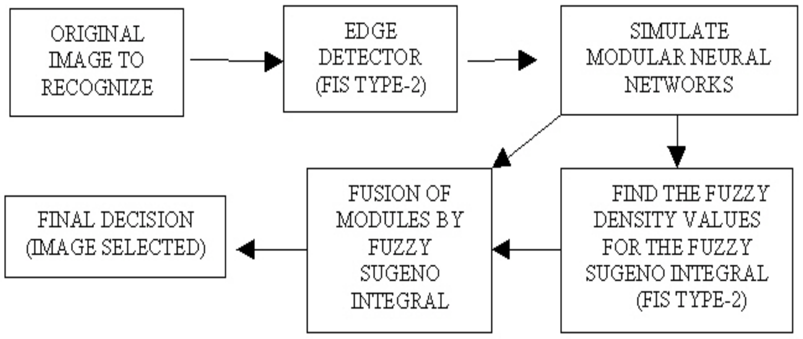

ORIGINAL IMAGE TO RECOGNIZE
EDGE DETECTOR (FIS TYPE-2)
SIMULATE MODULAR NEURAL NETWORKS
FIND THE FUZZY DENSITY VALUES FOR THE FUZZY SUGENO INTEGRAL (FIS TYPE-2)
FUSION OF MODULES BY FUZZY SUGENO INTEGRAL
FINAL DECISION (IMAGE SELECTED)

Fig. 3. Steps in Recognition Phase

2 Type-2 Fuzzy Inference System as an Edge Detector

In previous work we presented an efficient Fuzzy Inference System for edges detection, in order to use the output image like input data for modular neural networks (Mendoza and Melin, 2006). In the proposed technique, it is necessary to apply Sobel operators to the original images, then use a Fuzzy Inference System Type-2 to generate the vector of edges that would serve like input data in a neural network. Type-2 Fuzzy Logic enables us to handle uncertainties in decision making and recognition in a more convenient way and for this reason was proposed (Castillo et al., 2007).

For the Type-2 Fuzzy Inference System, 3 inputs are required, 2 of them are the gradients with respect to x-axis and y-axis, calculated with (1), to which we will call DH and DV respectively.

The Sobel edges detector uses a pair of 3x3 convolution masks, one estimating the gradient in the x-direction (columns) and the other estimating the gradient in the y-direction (rows).

$$Sobel_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad Sobel_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \tag{1}$$

Where Sobely y Sobelx are the Sobel Operators throughout x-axis and y-axis.

If we define I as the source image, g_x and g_y are two images which at each point contain the horizontal and vertical derivative approximations, the latter are computed as (2) and (3).

$$g_x = \sum_{i=1}^{i=3}\sum_{j=1}^{j=3} Sobel_{x,i,j} * I_{r+i-2,c+j-2} \tag{2}$$

$$g_y = \sum_{i=1}^{i=3}\sum_{j=1}^{j=3} Sobel_{y,i,j} * I_{r+i-2,c+j-2} \tag{3}$$

Where gx and gy are the gradients along axis-x and axis-y, and * represents the convolution operator.

The other input is a filter that calculates when applying a mask by convolution to the original image. The low-pass filter hMF (4) allow us to detect image pixels belonging to regions of the input were the mean gray level is lower. These regions are proportionally more affected by noise, supposed it is uniformly distributed over the whole image.

The goal here is to design a system which makes it easier to include edges in low contrast regions, but which does not favor false edges by effect of noise.

$$hMF = \frac{1}{25} * \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \tag{4}$$

Then the inputs for FIS type 2 are: DH=g_x, DV=g_y, M= $hMF*I$, where * is the convolution operator, and de output is a column vector contains the values of the image edges, and we can represent that in graphics shown in figure 4. The Edge Image is smaller than the original because the result of convolution operation is a central matrix where the convolution has a value. Then in our example, each image with dimension 112x92 is reduced to 108x88.

The inference rules and membership function parameters allow to calculate a gray value between -4.5 and 1.5 for each pixel, where the most negative values corresponds to the dark tone in the edges of the image. Then if we see the rules, only when the increment value of the inputs DH and DV are low the output is HIGH or clear (the background), in the rest of rules the output is LOW or dark (the edges). The complete set of fuzzy rules is given as follows (Castro et al., 2006):

1. If (DH is LOW) and (DV is LOW) then (EDGES is HIGH) (1)
2. If (DH is MEDIUM) and (DV is MEDIUM) then (EDGES is LOW) (1)
3. If (DH is HIGH) and (DV is HIGH) then (EDGES is LOW) (1)
4. If (M is LOW) and (DV is MEDIUM) then (EDGES is LOW) (1)
5. If (M is LOW) and (DH is MEDIUM) then (EDGES is LOW) (1)

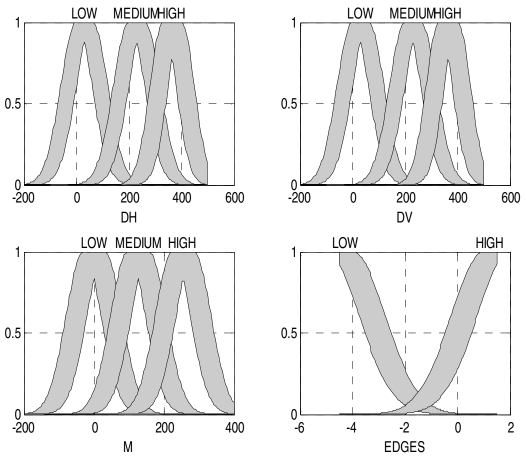


Fig. 4. Membership Function for the Type-2 FIS Edge Detector

The edge detector allows us to ignore the background color. We can see in this database of faces, different tones present for the same or another person. Then we eliminate a possible influence of a bad classification by the neural network, without losing detail in the image. Another advantage of edge detector is that the values can be normalized to a homogenous value range, independently the light, contrast or background tone in each image. At the examples in figure 5, all the edges in the images have a

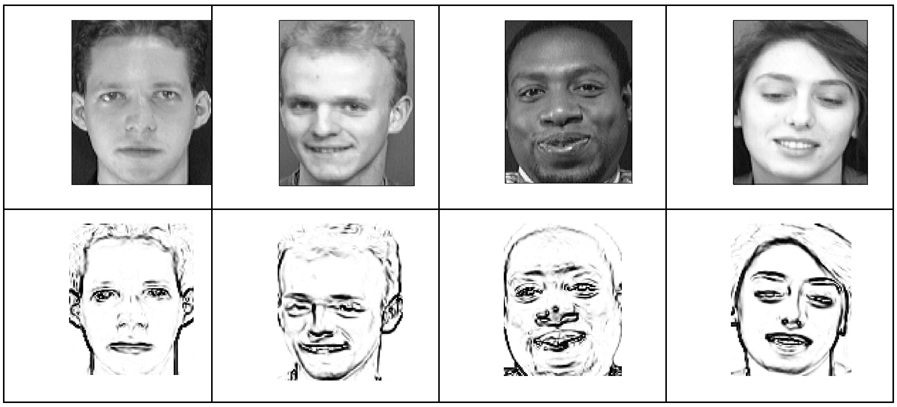


Fig. 5. Examples of edge detection with the Type-2 FIS method

minimum value of -3.8 and a maximum value of 0.84. In particular for neural network training, we find these values to make the training faster: the mean of the values is near 0 and the standard deviation is near 1 for all the images.

3 The Modular Structure

The design of the Modular Neural Network consists of 3 monolithic feedforward neural networks (Sharkey, 1999), each one trained with a supervised method with the first 7 samples of the 40 images. Then the edges vector column is accumulated until the number of samples to form the input matrix for the neural networks as it is in the scheme of figure 7. Once the complete matrix of images is divided in 3 parts, each module is training with a correspondent part, with some rows of overlap.

The target to the supervised training method consist of one identity matrix for each sample, building one matrix with dimensions 40x(40*number_of_samples).

Each Monolithic Neural Network has the same structure and is trained under the same conditions, like we can see in the next code segment:

```
layer1=200; layer2=200; layer3=number_of_subjects;
net=newff(minmax(p),[layer1,layer2,layer3],{'tansig','tansig','logsig'},'traingdx');
net.trainParam.goal=1e-5;
net.trainParam.epochs=1000;
```

The average number of epochs to meet the goal in each module is of 240, and the required time of 160 seconds.

4 Simulation Results

A program was developed in Matlab that simulates each module with the 400 images of the ORL database, building a matrix with the results of the simulation of each module. These matrices are stored in the file “mod.mat” to be analyzed later for the combination of results. We can observe that in the columns corresponding to the training data, the position with a value near one is the image selected correctly. However in the columns that correspond to the test data this doesn’t always happens, reason why it is very important to have a good combination method to recognize more images.

According to exhaustive tests made in the simulation matrices, we know that recognition of the images that were used for the training of the neural networks is of the 100%. Therefore the interest is focused on the recognition of the samples that do not belong to the training set, is to say samples 8,9 and 10. The parameters for the Sugeno Fuzzy Integral that will be inferred will be the Fuzzy Densities, a value between 0 and 1 for each module, which determines the rate for each module. The parameter lambda, according to the theory of fuzzy measures depends on the values of the fuzzy densities, and is calculated by searching for the roots of a polynomial. After the simulation of an image in the Neural Network, the simulation value is the only known parameter to make a decision, then to determine the fuzzy density for each module is the unique available information. For this reason we analyze the values in many simulations

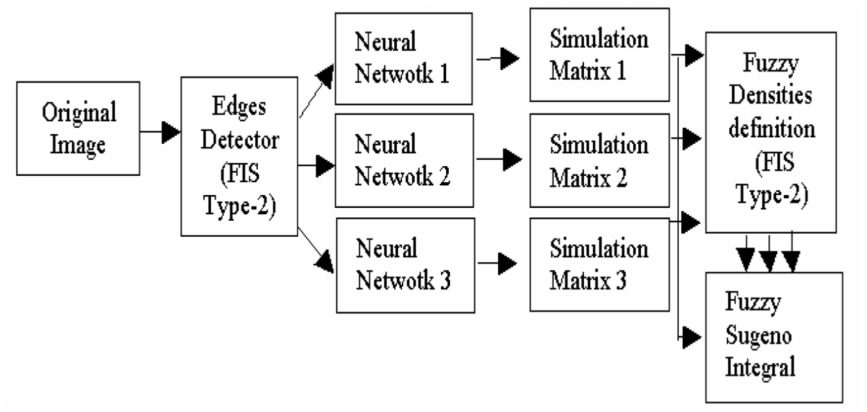


Fig. 6. Process of recognition using the type-2 fuzzy modular approach

matrix and decide that each input to the FIS Type-2 corresponds to the maximum value of each column corresponding to the simulation of each module of each one of the 400 images. The process to recognize each one of the images is shown in figure 6.

Then each output corresponds to one fuzzy density, to be applied for each module to perform the fusion of results later with the Fuzzy Sugeno Integral. The inference rules found fuzzy densities near 1 when de maximum value in the simulation is between 0.5 and 1, and near 0 when the maximum value in the simulation is near 0. The fuzzy rules are shown below and membership functions in Figure 7.

1. If (max1 is LOW) then (d1 is LOW) (1)
2. If (max2 is LOW) then (d2 is LOW) (1)
3. If (max3 is LOW) then (d3 is LOW) (1)
4. If (max1 is MEDIUM) then (d1 is HIGH) (1)
5. If (max2 is MEDIUM) then (d2 is HIGH) (1)
6. If (max3 is MEDIUM) then (d3 is HIGH) (1)
7. If (max1 is HIGH) then (d1 is HIGH) (1)
8. If (max2 is HIGH) then (d2 is HIGH) (1)
9. If (max3 is HIGH) then (d3 is HIGH) (1)

Although the rules are very simple, allows to model the fuzziness to rate de modules when the simulation result don't reach the maximum value 1.

However some of the images don't reach the sufficient value in the simulation of the three modules, in these cases, do not exists enough information to select an image at the modules combination, and the image is wrongly selected.

In order to measure of objective form the final results, we developed a method of random permutation, which rearranges the samples of each person before the training. Once a permutation is made, the modular neural networks are trained and combined four times to obtain the sufficient information to validate the results. The average recognition rate is of 96.5%.

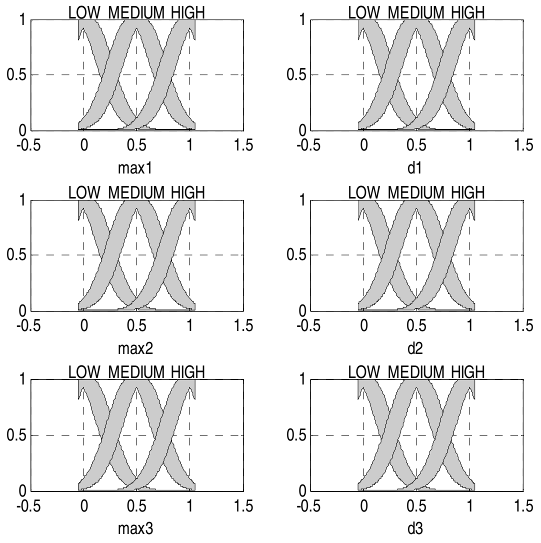


Fig. 7. Membership functions for the FIS to find fuzzy densities

We show in Table 1 the summary of simulation results for each of the modules and the average and maximum results of the modular network (after fusion or combination of the results).

Table 1. Summary of the Simulation Results with the Hybrid Approach

Permu-tation	Image Recognition (%)					
	Train 1	Train 2	Train 3	Train 4	Average	Maximum
1	92.75	95	92.2	93.25	93.3	95
2	96.5	95.25	94.25	95.5	95.375	96.5
3	91.5	92	93.75	95.25	93.125	95.25
4	94.5	94.5	93.25	94	94.0625	94.5
5	93.75	93.5	94	96	94.3125	96
					94.035	**96.5**

5 Interval Type-2 Fuzzy Logic for Digital Image Edge Detection

In the area of digital signal processing, methods have been proven to solve the problem of image recognition. Some of them include techniques like binarization, bidimensional filtrate, detection of edges and compression using banks of filters and trees, among others.

Specifically in methods for the detection of edges we can find comparative studies of methods like: Canny, Narwa, Iverson, Bergholm y Rothwell. Others methods can be grouped into two categories: Gradient and Laplacian.

The gradient methods like Roberts, Prewitt and Sobel detect edges, looking for maximum and minimum in first derived from the image. The Laplacian methods like

Marrs-Hildreth do it finding the zeros of second derived from the image (Mendoza and Melin, 2005).

This work is the beginning of an effort for the design of new pre-processing images techniques, using Fuzzy Inference Systems (FIS), which allows feature extraction and construction of input vectors for neural networks with aims of image recognition.

Artificial neural networks are one of the most used techniques in the automatic recognition of patterns, here are some reasons:

- Theoretically any function can be determined.
- Except the input patterns, it is not necessary to provide additional information.
- They are possible to be applied to any type of patterns and to any data type.

The idea to apply artificial neural networks for images recognition, tries to obtain results without providing another data that the original images, of this form the process is more similar to the form in which the biological brain learns to recognize patterns, only knowing experiences of past.

Models with modular neural networks have been designed, that allow recognizing images divided in four or six parts. This is necessary due to the great amount of input data, since an image without processing is of 100x100 pixels, needs a vector 10000 elements, where each one corresponds to pixel with variations of gray tones between 0 and 255 (Mendoza and Melin, 2005).

This chapter shows an efficient Fuzzy Inference System for edges detection, in order to use the output image like input data for modular neural networks. In the proposed technique, it is necessary to apply Sobel operators to the original images, and then use a Fuzzy System to generate the vector of edges that would serve as input data to a neural network.

6 Edge Detection by Gradient Magnitude

Although the idea presented in this chapter, is to verify the efficiency of a FIS for edges detection in digital images, from the approaches given by Sobel operator, is necessary to display first results using only the gradient magnitude.

The first image of subject number one of the ORL database will be used as an example (Figure 8). The gray tone of each pixel of this image is a value of between 0 and 255.

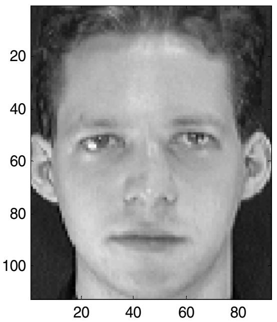


Fig. 8. Original Image 1.pgm

In figure 9 the image generated by *gx* is shown, and Figure 10 presents the image generated by *gy*.

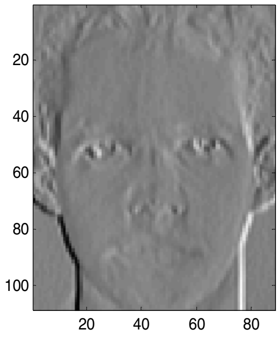


Fig. 9. Image given by *gx*

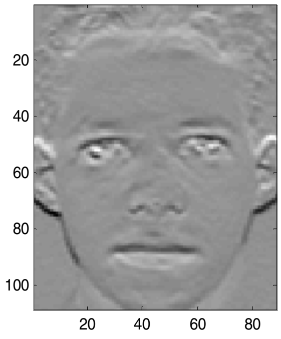


Fig. 10. Image given by *gy*

An example of maximum and minimum values of the matrix given by *gx*, *gy* and *g* from the image 1.pgm is shown in Table 2.

Table 2. Maximum and Minimum values from 1.pgm, gx, gy and g

Tone	1.pgm	gx	gy	g
Minimum	11	-725	-778	0
Maximum	234	738	494	792

After applying equation (4), g is obtained as it is in Figure 11.

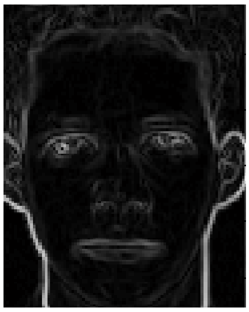


Fig. 11. Edges image given by g

7 Edge Detection Using Type-1 Fuzzy Logic

A Mamdani FIS was implemented using Type-1 Fuzzy Logic, with four inputs, one output and 7 rules, using the Matlab Fuzzy Logic Toolbox, which is shown in Figure 12.

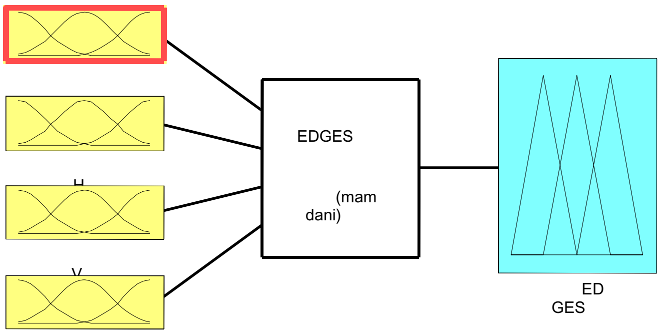


Fig. 12. FIS in Matlab Fuzzy Logic Tool Box

For the Type-1Fuzzy Inference System, 4 inputs are required, 2 of them are the gradients with respect to x-axis and y-axis, calculated with equation (2) and equation (3), to which we will call DH and DV respectively.

The other two inputs are filters: A high-pass filter, given by the mask of the equation (5), and a low-pass filter given by the mask of equation (6). The high-pass filter hHP detects the contrast of the image to guarantee the border detection in relative low contrast regions. The low-pass filter hMF allow to detects image pixels belonging to regions of the input were the mean gray level is lower. These regions are proportionally more affected by noise, supposed it is uniformly distributed over the whole image.

The goal here is to design a system which makes it easier to include edges in low contrast regions, but which does not favor false edges by effect of noise (Miosso and Bauchspiess, 2001).

$$hHP = \begin{bmatrix} -\frac{1}{16} & -\frac{1}{8} & -\frac{1}{16} \\ -\frac{1}{8} & \frac{3}{4} & -\frac{1}{8} \\ -\frac{1}{16} & -\frac{1}{8} & -\frac{1}{16} \end{bmatrix} \tag{5}$$

$$hMF = \frac{1}{25} * \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \tag{6}$$

Then the inputs for type-1 FIS are:

$DH=g_{x},$ $DV=g_{y}$ $HP= hHP*I$ $M= hMF*I$

where * is the convolution operator.

For all the fuzzy variables, the membership functions are of Gaussian type. According to the executed tests, the values in DH and DV, go from -800 to 800, then the ranks in x-axis adjusted as it is in figures 13, 14 and 15, in which the membership functions are:

LOW: gaussmf(43,0),
MEDIUM: gaussmf(43,127),
HIGH: gaussmf(43,255).

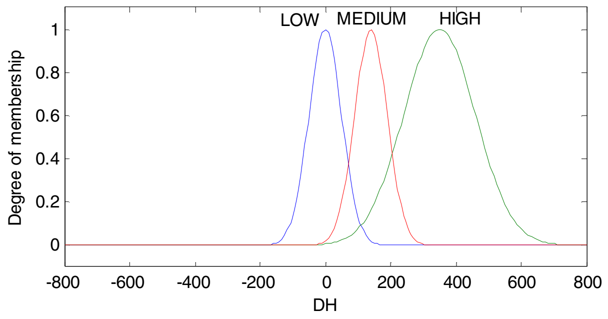


Fig. 13. Input variable DH

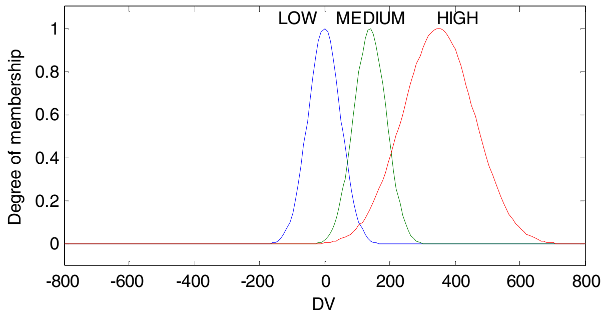


Fig. 14. Input variable DV

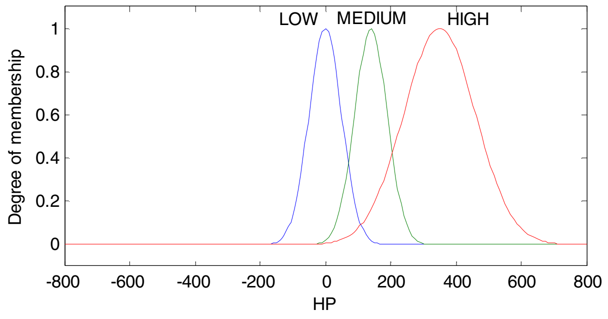


Fig. 15. Input variable HP

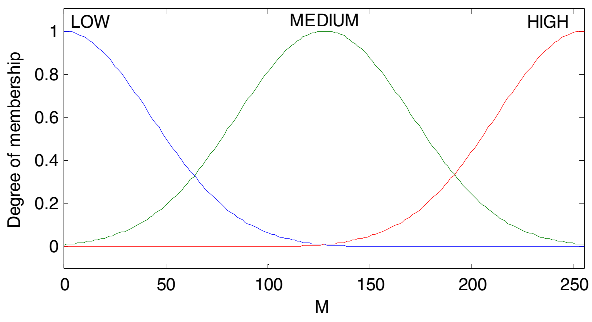


Fig. 16. Input variable M

In the case of variable M, the tests threw values in the rank from 0 to 255, and thus the rank in x-axis adjusted, as it is shown in figure 16.

In figure 17 the output variable EDGES is shown, that also adjusted the ranks between 0 and 255, since it is the range of values required to display the edges of an image.

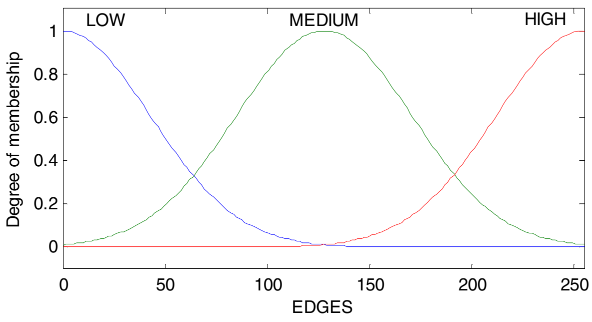


Fig. 17. Output variable EDGES

The seven fuzzy rules that allow to evaluate the input variables, so that the exit image displays the edges of the image in color near white (HIGH tone), whereas the background was in tones near black (tone LOW).

1. If (DH is LOW) and (DV is LOW) then (EDGES is LOW)
2. If (DH is MEDIUM) and (DV is MEDIUM) then (EDGES is HIGH)
3. If (DH is HIGH) and (DV is HIGH) then (EDGES is HIGH)
4. If (DH is MEDIUM) and (HP is LOW) then (EDGES is HIGH)
5. If (DV is MEDIUM) and (HP is LOW) then (EDGES is HIGH)
6. If (M is LOW) and (DV is MEDIUM) then (EDGES is LOW)
7. If (M is LOW) and (DH is MEDIUM) then (EDGES is LOW)

The result obtained for image of figure 8 is remarkably better than the one than it was obtained with the method of gradient magnitude, as it is in Figure 18.

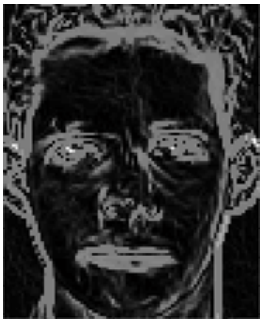


Fig. 18. EDGES Image by FIS Type 1

Reviewing the values of each pixel, we see that all fall in the rank from 0 to 255, which is not obtained with the method of gradient magnitude.

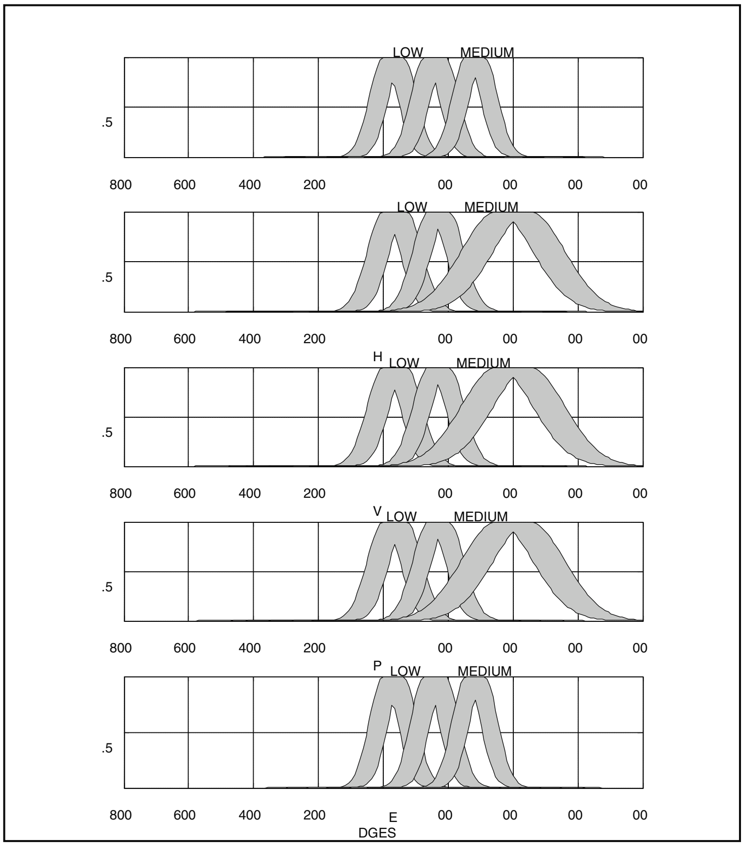


Fig. 19. Type-2 fuzzy variables

8 Edge Detection Using Type-2 Fuzzy Logic

For the Type-2 FIS, the same method was followed as in Type-1 FIS, indeed to be able to make a comparison of both results. The tests with the type-2 FIS, were executed using the computer program imagen_bordes_fis2.m, which creates a Type-2 Inference System (Mamdani, 1993) by intervals (Mendel, 2001).

The mentioned program creates the type-2 fuzzy variables as it is seen in figure 19. The wide of the FOU chosen for each membership function was the one that had better results after several experiments. The program imagen_bordes_fuzzy2.m was implemented to load the original image, and to apply the filters before mentioned. Because the great amount of data that the fuzzy rules must evaluate, the image was divided in four parts, and the FIS was applied to each one separately. The result of each evaluation gives a vector with tones of gray by each part of the image, in the end is the complete image with the edges (figure 20).

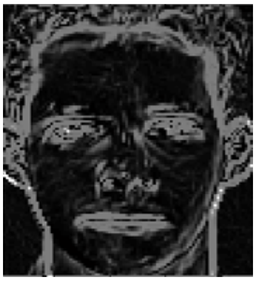


Fig. 20. EDGES Image by the Type-2 FIS

9 Comparison of Results

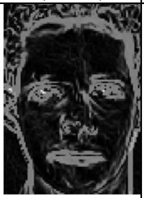
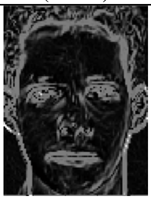
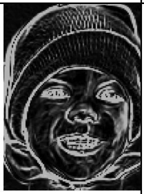
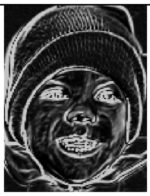
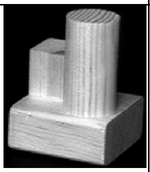
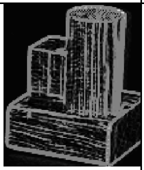
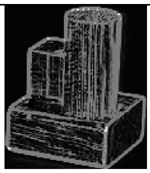
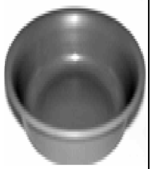
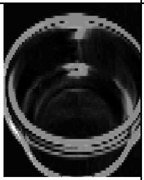
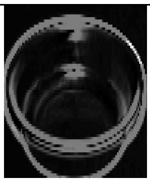
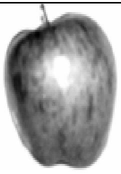
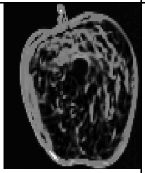
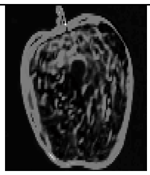
The first results of several tests conducted in different images can be appreciated in table 3.

At first, the results with the Type-1 FIS and Type-2 FIS are seen to be very similar. However thinking about that to show the images with a dark background it could confuse the contrast of tones, tests were done inverting the consequent of the rules, so that the edges take the dark tone and the bottom the clear tone, the rules changed to the following form:

1. If (DH is LOW) and (DV is LOW) then (EDGES is HIGH)
2. If (DH is MEDIUM) and (DV is MEDIUM) then (EDGES is LOW)
3. If (DH is HIGH) and (DV is HIGH) then (EDGES is LOW)
4. If (DH is MEDIUM) and (HP is LOW) then (EDGES is LOW)
5. If (DV is MEDIUM) and (HP is LOW) then (EDGES is LOW)
6. If (M is LOW) and (DV is MEDIUM) then (EDGES is HIGH)
7. If (M is LOW) and (DH is MEDIUM) then (EDGES is HIGH)

Fuzzy Systems were tested both (Type-1 and Type-2), with the new fuzzy rules and same images, obtaining the results that are in Table 4.

Table 3. Results of Edge Detection by FIS1 and FIS2 (Dark Background)

Original Image	EDGES (FIS 1)	EDGES (FIS 2)
		
		
		
		
		

In this second test can be appreciated a great difference between the results obtained with the FIS 1 and FIS 2, noticing at first a greater contrast in the images obtained with the FIS 1 and giving to the impression of a smaller range of tones of gray in the type-2 FIS.

In order to obtain an objective comparison of the images, histograms were elaborated respectively corresponding to the resulting matrices of edges of the FIS 1 and FIS 2, which are in table 5.

The histograms show in the y-axis the range of tones of gray corresponding to each image and in x-axis the frequency in which he appears pixel with each tone.

Table 4. Results of Edge Detection by FIS1 and FIS2 (Clear Background)

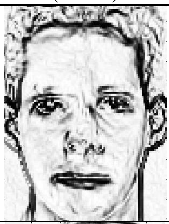
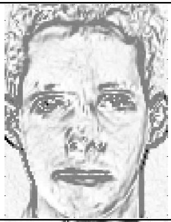
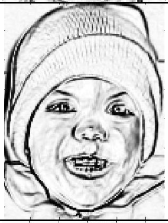
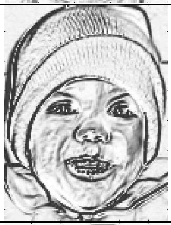
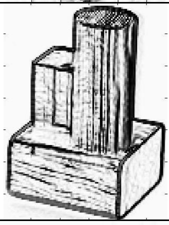
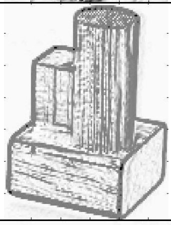
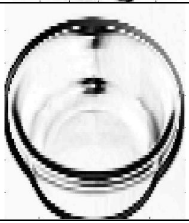
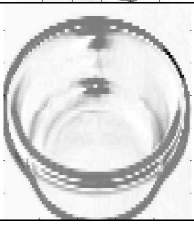
EDGES (FIS 1)	EDGES (FIS 2)
	
	
	
	

Table 5. Histograms of the Resulting Images of Edge Detection by the Gradient Magnitude, FIS1 and FIS 2 methods

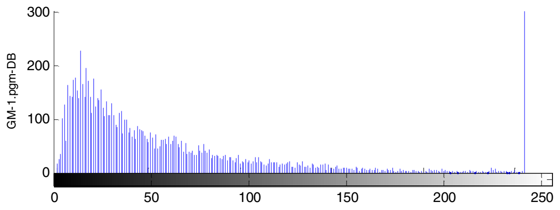
IMAGE: 1.PGM
METHOD: GRADIENT MAGNITUDE

METHOD: FIS 1 (CLEAR BACKGROUND)

Table 5. (*continued*)

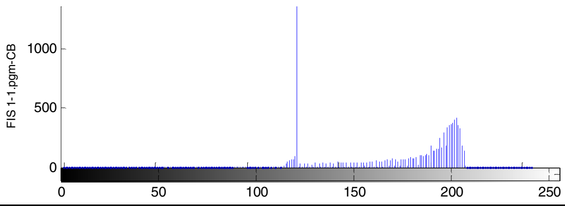
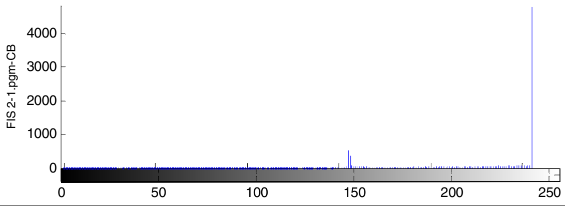
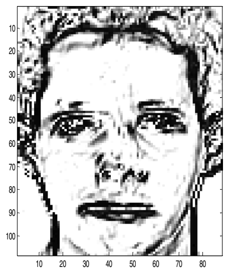
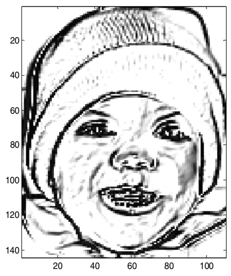

METHOD: FIS 2 (CLEAR BACKGROUND)


Table 6. Type-2 FIS Edges Images including Pixels with Tones between 150 and 255

BORDERS IMAGE	DIMENSION (pixels)	PIXELS INCLUDED
	108x88 (9504)	4661 49 %
	144x110 (15840)	7077 44.6 %

As we can observe, unlike detector FIS1, with FIS2 the edges of an image could be obtained from very complete form, only taking the tones around 150 and 255.

As a last experiment, in this occasion to the resulting images of the Type-2 FIS the every pixel out of the range between 50 and 255 was eliminated.

Table 6 shows the amount of elements that was possible to eliminate in some of the images, we see that the Type-2 Edges Detector FIS allows to using less than half of the original pixels without losing the detail of the images. This feature could be a great advantage if these images are used like input data in neural networks for detection of images instead the original images.

10 Systematic Design of a Stable Type-2 Fuzzy Logic Controller

Stability has been one of the central issues concerning fuzzy control since Mamdani's pioneer work (Mamdani and Assilian, 1975). Most of the critical comments to fuzzy control are due to the lack of a general method for its stability analysis.

But as Zadeh often points out, fuzzy control has been accepted by the fact that it is task-oriented control, while conventional control is characterized as setpoint-oriented control, and hence do not need a mathematical analysis of stability. Also, as Sugeno has mentioned, in general, in most industrial applications, the stability of control is not fully guaranteed and the reliability of a control hardware system is considered to be more important than the stability (Sugeno, 1999).

The success of fuzzy control, however, does not imply that we do not need a stability theory for it. Perhaps the main drawback of the lack of stability analysis would be that we cannot take a model-based approach to fuzzy control design. In conventional control theory, a feedback controller can be primarily designed so that a close-loop system becomes stable. This approach of course restricts us to setpoint-oriented control, but stability theory will certainly give us a wider view on the future development of fuzzy control.

Therefore, many researchers have worked to improve the performance of the FLC's and ensure their stability. Li and Gatland in 1995 proposed a more systematic design method for PD and PI-type FLC's. Choi, Kwak and Kim (Choi et al., 2000) present a single-input FLC ensuring stability. Ying in 1994 presented a practical design method for nonlinear fuzzy controllers, and many other researchers have results on the matter of the stability of FLC's, in (Castillo et al., 2005) and (Cázarez et al., 2005) presents an extension of the Margaliot work (Margaliot and G. Langholz, 2000) to built stable type-2 fuzzy logic controllers in Lyapunov sense.

This work is based on Margaliot´s work (Margaliot and Langholtz, 2000), we use the Fuzzy Lyapunov Synthesis to build an Stable Type-2 Fuzzy Logic Controller for a 1 Degree of Freedom (DOF) manipulator robot, first without gravity effect to prove stability, and then with gravity effect to prove the robustness of the controller. The same criteria can be used for any number of DOF manipulator robots, linear or nonlinear, and any kind of plants.

11 Fuzzy Logic Controllers

11.1 Type-1 Fuzzy Logic Control

Type-1 FLCs are both intuitive and numerical systems that map crisp inputs to a crisp output. Every FLC is associated with a set of rules with meaningful linguistic interpretations, such as

R^l : If x_1 is F_1^l and x_2 is F_2^l and … and x_n is F_n^l Then w is G^l

which can be obtained either from numerical data, or experts familiar with the problem at hand. Based on this kind of statement, actions are combined with rules in an antecedent/consequent format, and then aggregated according to approximate reasoning theory, to produce a nonlinear mapping from input space $U = U_1 x U_2 x..U_n$ to the output space W, where $F_k^l \subset U_k, k = 1,2,...,n$, are the antecedent type-1 membership functions, and $G^l \subset W$ is the consequent type-1 membership function. The input linguistic variables are denoted by $u_k, k = 1,2,...,n$, and the output linguistic variable is denoted by w.

A Fuzzy Logic System (FLS), as the kernel of a FLC, consist of four basic elements (Fig. 21): the type-1 fuzzyfier, the fuzzy rule-base, the inference engine, and the type-1 defuzzyfier. The fuzzy rule-base is a collection of rules in the form of R^l, which are combined in the inference engine, to produce a fuzzy output. The type-1 fuzzyfier maps the crisp input into type-1 fuzzy sets, which are subsequently used as inputs to the inference engine, whereas the type-1 defuzzyfier maps the type-1 fuzzy sets produced by the inference engine into crisp numbers.

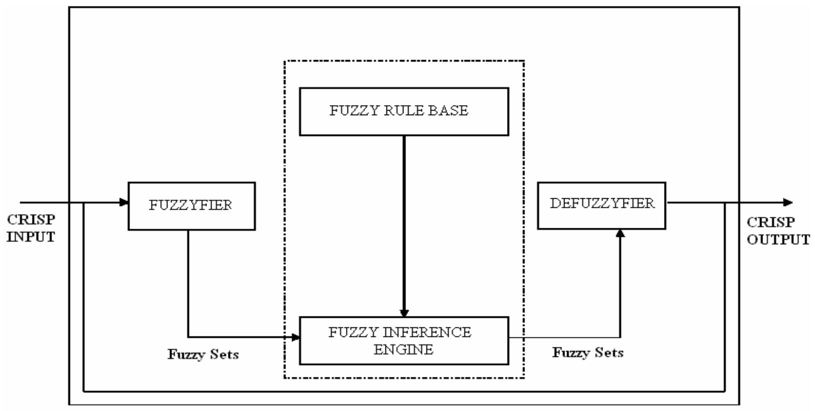


Fig. 21. Structure of type-1 fuzzy logic system

Fuzzy sets can be interpreted as membership functions u_X that associate with each element x of the universe of discourse, U, a number $u_X(x)$ in the interval [0,1]:

$$u_X : U \rightarrow [0,1] \tag{7}$$

For more detail of Type-1 FLS see (Chen and Pham, 2001).

11.2 Type-2 Fuzzy Logic Control

As with the type-1 fuzzy set, the concept of type-2 fuzzy set was introduced by Zadeh as an extension of the concept of an ordinary fuzzy set (Zadeh, 1975).

A FLS described using at least one type-2 fuzzy set is called a type-2 FLS. Type-1 FLSs are unable to directly handle rule uncertainties, because they use type-1 fuzzy sets that are certain. On the other hand, type-2 FLSs, are very useful in circumstances where it is difficult to determine an exact, and measurement uncertainties (Mendel, 2000).

It is known that type-2 fuzzy set let us to model and to minimize the effects of uncertainties in rule-based FLS. Unfortunately, type-2 fuzzy sets are more difficult to use and understand that type-1 fuzzy sets; hence, their use is not widespread yet.

Similar to a type-1 FLS, a type-2 FLS includes type-2 fuzzyfier, rule-base, inference engine and substitutes the defuzzifier by the output processor. The output processor includes a type-reducer and a type-2 defuzzyfier; it generates a type-1 fuzzy set output (from the type reducer) or a crisp number (from the defuzzyfier). A type-2 FLS is again characterized by IF-THEN rules, but its antecedent and consequent sets are now of type-2. Type-2 FLSs, can be used when the circumstances are too uncertain to determine exact membership grades. A model of a type-2 FLS is shown in Figure 22.

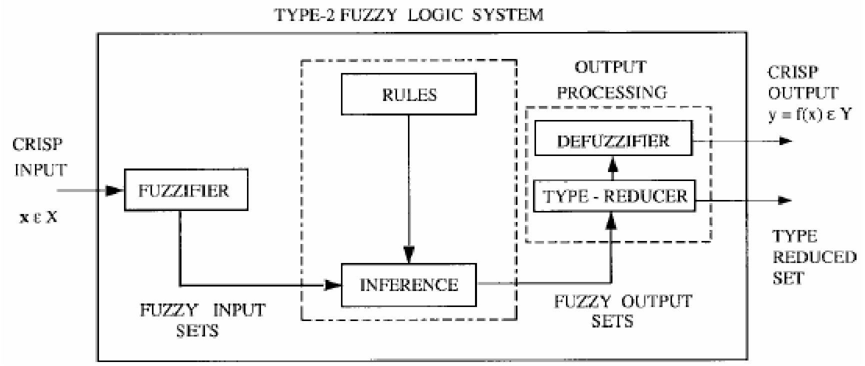


Fig. 22. Structure of type-2 fuzzy logic system

In the case of the implementation of type-2 FLCs, we have the same characteristics as in type-1 FLC, but we now use type-2 fuzzy sets as membership functions for the inputs and for the outputs. Fig. 23 shows the structure of a control loop with a FLC.

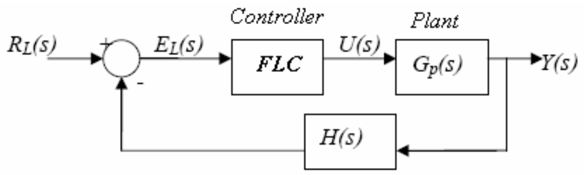


Fig. 23. Fuzzy control loop

12 Systematic and Design of Stable Fuzzy Controllers

For our description we consider the problem of designing a stabilizing controller for a 1DOF manipulator robot system depicted in Fig. 24. The state-variables are $x_1 = \theta$ - the robot arm angle, and $x_2 = \dot{\theta}$ - its angular velocity. The system's actual dynamical equation, which we will assume that are unknown, is as shown in equation (8) (Paul and Yang, 1999):

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \tau \tag{8}$$

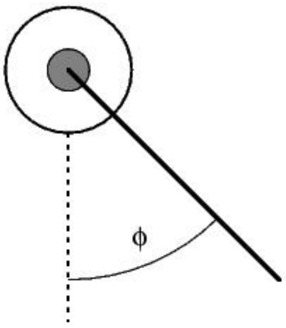


Fig. 24. 1DOF Manipulator robot

To apply the fuzzy Lyapunov synthesis method, we assume that the exact equations are unknown and that we have only the following partial knowledge about the plant (see Figure 24):

1. The system may have really two degrees of freedom θ and $\dot{\theta}$, referred to as x_1 and x_2, respectively. Hence, $\dot{x}_1 = x_2$.
2. $\dot{x}_2$ is proportional to u, that is, when u increases (decreases) $\dot{x}_2$ increases (decreases).

To facilitate our control design we are going to suppose no gravity effect in our model, see (equation 9).

$$ml^2\ddot{q} = \tau \tag{9}$$

Our objective is to design the rule-base of a fuzzy controller that will carry the robot arm to a desired position $x_1 = \theta d$. We choose (10) as our Lyapunov function candidate. Clearly, V is positive-definite.

$$V(x_1, x_2) = \frac{1}{2}(x_1^2 + x_2^2) \tag{10}$$

Differentiating V, we have equation (11),

$$\dot{V} = x_1\dot{x}_1 + x_2\dot{x}_2 = x_1x_2 + x_2\dot{x}_2 \tag{11}$$

Hence, we require:

$$x_1 x_2 + x_2 \dot{x}_2 < 0 \tag{12}$$

We can now derive sufficient conditions so that condition (12) holds: If x_1 and x_2 have opposite signs, then $x_1 x_2 < 0$ and (12) will hold if $\dot{x}_2 = 0$; if x_1 and x_2 are both positive, then (12) will hold if $\dot{x}_2 < -x_1$; and if x_1 and x_2 are both negative, then (12) will hold if $\dot{x}_2 > -x_1$.

We can translate these conditions into the following fuzzy rules:

- If x_1 is *positive* and x_2 is *positive* then $\dot{x}_2$ must be *negative big*
- If x_1 is *negative* and x_2 is *negative* then $\dot{x}_2$ must be *positive big*
- If x_1 is *positive* and x_2 is *negative* then $\dot{x}_2$ must be *zero*
- If x_1 is *negative* and x_2 is *positive* then $\dot{x}_2$ must be *zero.*

However, using our knowledge that $\dot{x}_2$ is proportional to u, we can replace each $\dot{x}_2$ with u to obtain the fuzzy rule-base for the stabilizing controller:

- If x_1 is *positive* and x_2 is *positive* Then u must be *negative big*
- If x_1 is *negative* and x_2 is *negative* Then u must be *positive big*
- If x_1 is *positive* and x_2 is *negative* Then u must be *zero*
- If x_1 is *negative* and x_2 is *positive* Then u must be *zero.*

It is interesting to note that the fuzzy partitions for x_1, x_2, and u follow elegantly from expression (11). Because $\dot{V} = x_2(x_1 + \dot{x}_2)$, and since we require that $\dot{V}$ be negative, it is natural to examine the signs of x_1 and x_2; hence, the obvious fuzzy partition is *positive*, *negative*. The partition for $\dot{x}_2$, namely *negative big*, *zero*, *positive big* is obtained similarly when we plug the linguistic values *positive*, *negative* for x_1 and x_2 in (11). To ensure that $\dot{x}_2 < -x_1$ $(\dot{x}_2 > -x_1)$ is satisfied even though we do not know x_1's exact magnitude, only that it is *positive* (*negative*), we must set $\dot{x}_2$ to *negative big* (*positive big*). Obviously, it is also possible to start with a given, predefined, partition for the variables and then plug each value in the expression for $\dot{V}$ to find the rules. Nevertheless, regardless of what comes first, we see that fuzzy Lyapunov synthesis transforms classical Lyapunov synthesis from the world of exact mathematical quantities to the world of computing with words (Zadeh, 1996).

To complete the controllers design, we must model the linguistic terms in the rule-base using fuzzy membership functions and determine an inference method. Following (Wang, 1997), we characterize the linguistic terms *positive*, *negative*, *negative big*, *zero* and *positive big* by the type-1 membership functions shown in Fig. 25 for a Type-1 Fuzzy Logic Controller, and by the type-2 membership functions shown in Figure 26 for a Type-2 Fuzzy Logic Controller. Note that the type-2 membership functions are extended type-1 membership functions.

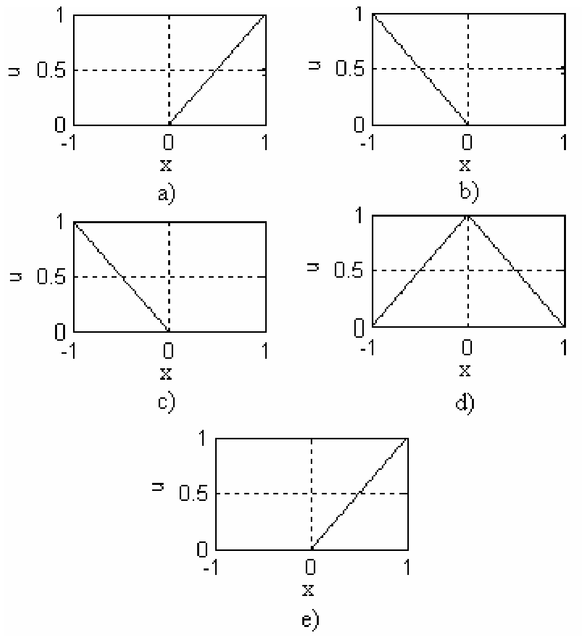


Fig. 25. Set of type-1 membership functions: a) positive, b)negative, c) negative big, d) zero and e) positive big

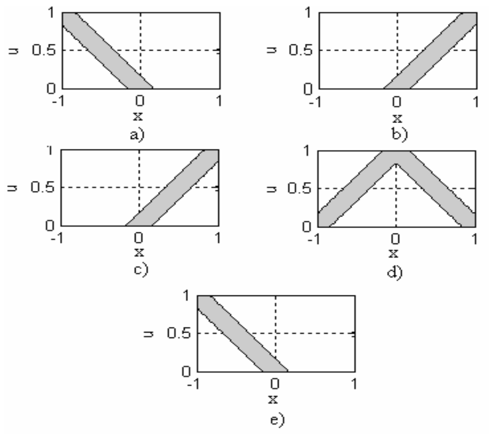


Fig. 26. Set of type-2 membership functions: a)negative, b) positive, c) positive big, d) zero and e) negative big

To this end, we had systematically developed a FLC rule-base that follows the Lyapunov Stability criterion. In Section 13 we present some experimental results using our fuzzy rule-base to build a Type-2 Fuzzy Logic Controller.

13 Experimental Results

In Section 12 we had systematically developed a stable FLC rule-base, and now we are going to show some experimental results using our stable rule-base to build a Type-2 FLC. The plant description used in the experiments is the same shown in Section 12.

Our experiments were done with Type-1 Fuzzy Sets and Interval Type-2 Fuzzy Sets. In the Type-2 Fuzzy Sets the membership grade of every domain point is a crisp set whose domain is some interval contained in [0,1] (Mendel, 2000). On Fig. 26 we show some Interval Type-2 Fuzzy Sets, and for each fuzzy set, the grey area is known as the Footprint of Uncertainty (FOU) (Mendel, 2000), and this is bounded by an upper and a lower membership function as shown in Fig. 27.

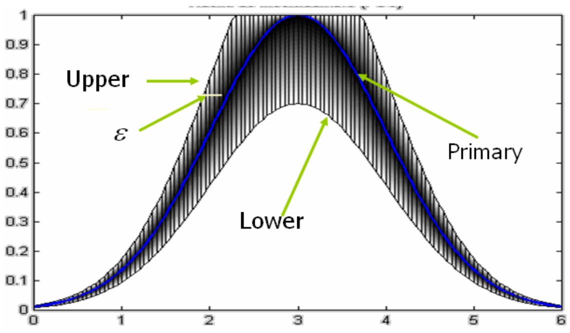


Fig. 27. Interval Type-2 Fuzzy Set

In our experiments we increase and decrease the value of ε to the left and to the right side having a εL and a εR values respectively to determine how much the FOU can be extended or perturbed without losing stability in the FLC.

We did make simulations with initial conditions of θ having values in the whole circumference [0, 2π], and the desired angle θd having values in the same range. The initial conditions considered in the experiments shown in this paper are an angle $\theta = 0rad$ and $\theta_d = 0.1rad$.

In Fig. 28 we show a simulation of the plant made with a Type-1 FLC, as can be seen, the plant has been regulated in around 8 seconds, and in Fig. 29 we show the graph of equation (11) which is always negative defined and consequently the system is stable.

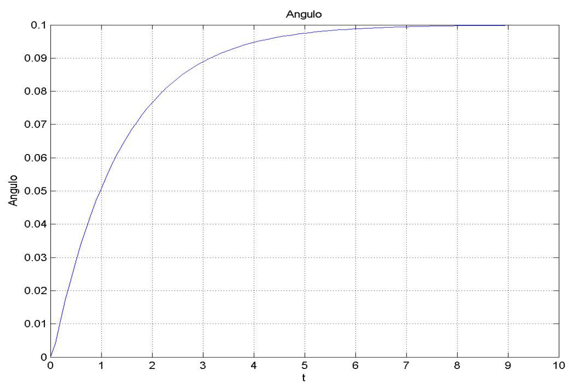


Fig. 28. Response for the Type-1 FLC

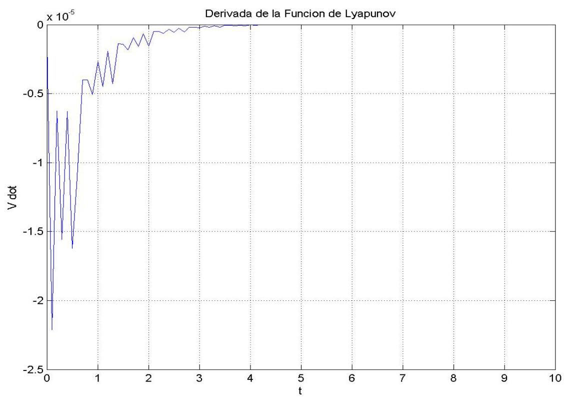


Fig. 29. $\dot{V}$ for the Type-1 FLC

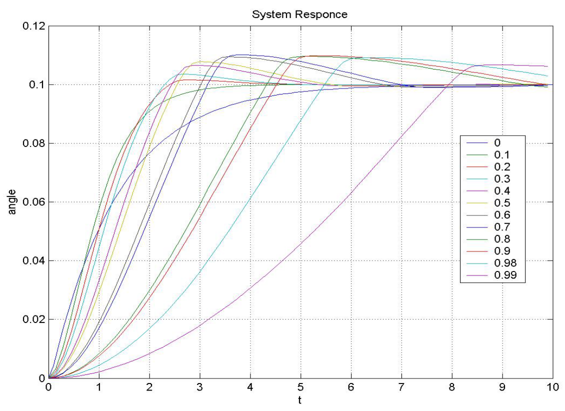


Fig. 30. Response for the Type-2 FLC ($\varepsilon \to [0,1)$)

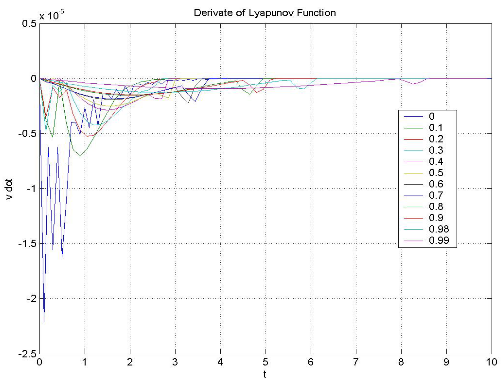


Fig. 31. $\dot{V}$ for the Type-2 FLC ($\varepsilon \rightarrow [0,1]$)

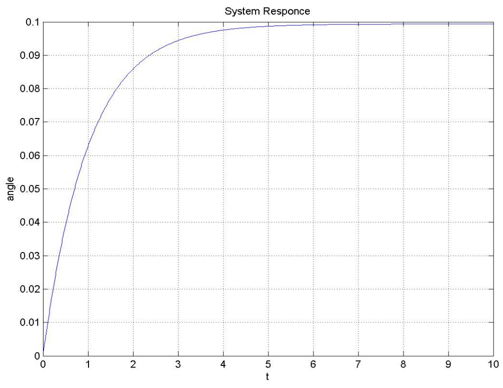


Fig. 32. Response for the Type-1 FLC

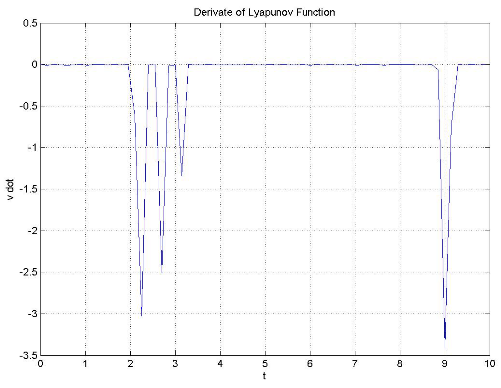


Fig. 33. $\dot{V}$ for the Type-1 FLC

Figure 30 shows the simulation results of the plant made with the Type-2 FLC increasing and decreasing ε in the range of [0,1], and as can be seen the plant has been regulated in around 10 seconds, and the graph of equation (11), which is depicted in Fig. 31 is always negative defined and consequently the system is stable. As we can seen, the time response is increasing when the value of ε is increasing.

With the variation of ε in the definition of the FOU, the control surface changes proportional to the change of ε, for this reason, the value of u for $\varepsilon \geq 1$ is practically zero, and the plant does not have physical response. To test the robustness of the built Fuzzy Controller, now we are going to use the same controller designed in Section 12, but at this time, we are going to use it to control equation (8) considering the gravity effect as shown in equation (13).

$$ml^2\ddot{q} + gml\cos q = \tau \tag{13}$$

In Figure 32 we can see a simulation of the plant obtained with a Type-1 FLC, and as can be seen, the plant has been regulated in approximately 8 seconds and Figure 33 shows the graph of equation (11) which is always negative defined and consequently the system is stable.

Figure 34 shows the simulation results of the plant obtained with the Type-2 FLC with increasing and decreasing ε values in the range of [0,1], and the graph of (11) depicted at Fig. 35 is always negative defined and consequently the system is stable. As we can seen, if we use an adaptive gain like in (Castillo et al., 2005) all the cases of ε can be regulated around 8 seconds.

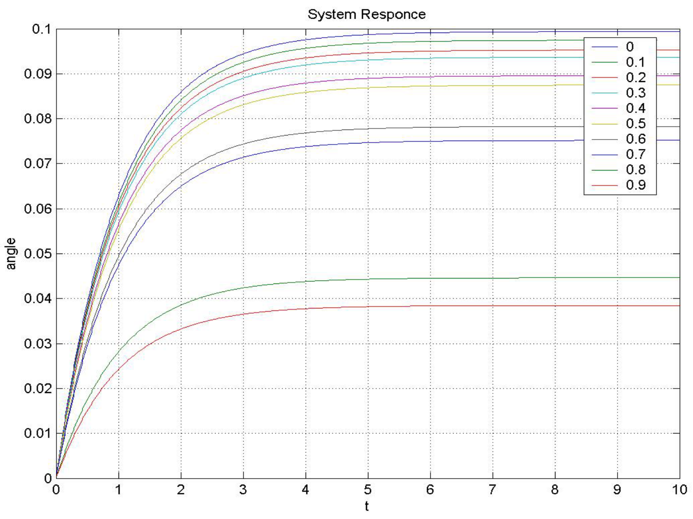


Fig. 34. Response for the Type-2 FLC ($\varepsilon \to [0,1)$)

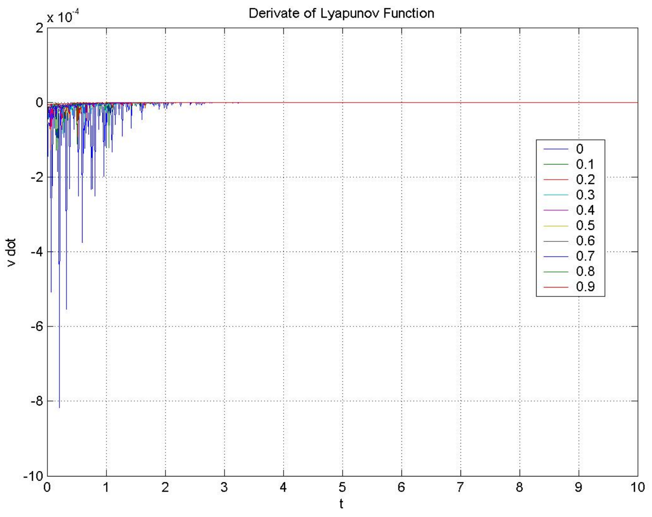


Fig. 35. $\dot{V}$ for the Type-2 FLC ($\varepsilon \rightarrow [0,1]$)

14 Conclusions

In this chapter three applications of interval type-2 fuzzy logic have been described. First, the use of interval type-2 fuzzy logic is used to improve performance on a modular neural network for face recognition. Second, interval type-2 fuzzy logic is used to improve edge detection in image processing. Finally, a method for designing stable interval type-2 fuzzy logic controllers is proposed. In all cases, the results of type-2 fuzzy logic are shown to be superior with respect to the type-1 corresponding ones.

References

1. Castillo, O., Aguilar, L., Cazarez, N., Rico, D.: Intelligent Control of Dynamical Systems with Type-2 Fuzzy Logic and stability Study. In: Proceedings of the Conference on Artificial Intelligence (ICAI 2005), pp. 400–405 (2005)
2. Castillo, O., Melin, P., Kacprzyk, J., Pedrycz, W.: Hybrid Intelligent Systems: Design and Analysis. Springer, Germany (2007)
3. Castro, J.R., Castillo, O., Martínez-Méndez, L.G.: Tool Box para Lógica Difusa Tipo-2 por Intervalos. In: International Seminar on Computational Intelligence, IEEE - CIS Mexico Chapter, Tijuana, Mexico, October 9-11 (CD-rom proceedings) (2006)
4. Cázarez, N.R., Cárdenas, S., Aguilar, L., Castillo, O.: Lyapunov Stability on Type-2 Fuzzy Logic Control. In: IEEE-CIS International Seminar on Computational Intelligence, México Distrito Federal, México, October 17-18 (Proceedings in CD-rom) (2005)
5. Chen, G., Pham, T.T.: Introduction to Fuzzy Sets, Fuzzy Logic, and Fuzzy Control Systems. CRC Press, Boca Raton (2001)
6. Choi, B.J., Kwak, S.W., Kim, B.K.: Design and Stability Analysis of Single-Input Fuzzy Logic Controller. IEEE Trans. Fuzzy Systems 30, 303–309 (2000)

7. Chuang, M.-M., Chang, R.-F., Huang, Y.-L.: Automatic, Facial Feature Extraction in Model-based Coding. Journal of Information Science and Engineering 16, 447–458 (2000)
8. Klir, G.J., Yuan, B.: Fuzzy Sets and Fuzzy Logic. Prentice-Hall, New York (1995)
9. Kolmanovsky, I., McClamroch., N.H.: Developments in Nonholonomic Nontrol Problems. IEEE Control Syst. Mag. 15, 20–36 (1995)
10. Kosko, B.: Neural Networks and Fuzzy Systems: A Dynamical Systems Approach to Machine Intelligence. Prentice-Hall, Englewood Cliffs (1992)
11. Lee, T.H., Leung, F.H.F., Tam, P.K.S.: Position Control for Wheeled Mobile Robot Using a Fuzzy Controller, pp. 525–528. IEEE, Los Alamitos (1999)
12. Li, H.-X., Gatland, H.B.: A new methodology for designing a fuzzy logic controller. IEEE Trans. Systems Man and Cybernetics 25, 505–512 (1995)
13. Mamdani, E.H., Assilian, S.: An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller. International Journal of Man-Machine Studies 7, 1–13 (1975)
14. Mamdani, E.H.: Twenty years of Fuzzy Control: Experiences Gained and Lessons Learned. In: Proc. 2nd IEEE International Conference on Fuzzy Systems, San Francisco, CA, pp. 339–344 (1993)
15. Margaliot, M., Langholz, G.: New Approaches to Fuzzy Modeling and Control. World Scientific Press, Singapore (2000)
16. Melin, P., Castillo, O.: Hybrid Intelligent Systems for Pattern Recognition Using Soft Computing. Springer, Heidelberg (2005)
17. Melin, P., Mancilla, A., Lopez, M., Mendoza, O.: A hybrid modular neural network architecture with fuzzy Sugeno integration for time series forecasting. Journal of Applied Soft Computing (accepted for publication, 2007)
18. Mendel, J.M.: Uncertainty, fuzzy logic, and signal processing. Signal Processing Journal 80, 913–933 (2000)
19. Mendel, J.M.: Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions. Prentice-Hall, New Jersey (2001)
20. Mendoza, O., Melin, P.: The Fuzzy Sugeno Integral as a Decision Operator in the Recognition of Images with Modular Neural Networks. In: International Conference on Fuzzy Systems, Neural Networks and Genetic Algorithms FNG 2005 (Proceedings in CD-rom) (2005)
21. Mendoza, O., Melin, P.: Sistemas de Inferencia Difusos Tipo-1 y Tipo-2 Aplicados a la Detección de Bordes en Imágenes Digitales. In: International Seminar on Computational Intelligence, IEEE - CIS Mexico Chapter, Tijuana, Mexico, October 9-11, 2006, Tijuana Institute of Technology (CD-rom proceedings) (2006)
22. Mendoza, O., Melin, P.: The Fuzzy Sugeno Integral as a Decision Operator in the Recognition of Images with Modular Neural Networks. In: Edited Book: Hybrid Intelligent Systems: Design and Analysis, pp. 299–310. Springer, Heidelberg (2007)
23. Sharkey, A.: Combining Artificial Neural Nets: Ensemble and Modular Multi-Net Systems. Springer, London (1999)
24. Sugeno, M.: On Stability of Fuzzy Systems Expressed by Fuzzy Rules with Singleton Consequents. IEEE Trans. Fuzzy Systems 7(2) (1999)
25. Wang, L.-X.: A Course in Fuzzy Systems and Control. Prentice-Hall, Englewood Cliffs (1997)
26. Zadeh, L.A.: The Concept of a Linguistic Variable and its Application to Approximate Reasoning–1. Journal of Information Sciences 8, 199–249 (1975)
27. Zadeh, L.A.: Fuzzy Logic = Computing with Words. IEEE Transactions on Fuzzy Systems 4(2) (1996)