Information Sciences 179 (2009) 2078–2101

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

A hybrid approach for image recognition combining type-2 fuzzy logic, modular neural networks and the Sugeno integral

Olivia Mendoza [a], Patricia Melin [b,*], Guillermo Licea [a]

[a] Division of Research and Graduate Studies, Universidad Autonoma de Baja California, Tijuana, Mexico
[b] Department of Research and Graduate Studies, Tijuana Institute of Technology, P.O. Box 4207, Chula Vista, CA 91909, United States

ARTICLE INFO

Article history:
Received 7 November 2007
Received in revised form 23 September 2008
Accepted 10 November 2008

ABSTRACT

In this paper, a hybrid approach for image recognition combining type-2 fuzzy logic, modular neural networks and the Sugeno integral is described. Interval type-2 fuzzy inference systems are used to perform edge detection and to calculate fuzzy densities for the decision process. A type-2 fuzzy system is used for edge detection, which is a pre-processing applied to the training data for better use in the neural networks. Another type-2 fuzzy system calculates the fuzzy densities necessary for the Sugeno integral, which is used to integrate results of the neural network modules. In this case, fuzzy logic is shown to be a good methodology to improve the results of a neural system facilitating the representation of the human perception. A comparative study is also made to verify that the proposed approach is better than existing approaches and improves the performance over type-1 fuzzy logic.

1. Introduction

In this paper, we describe an image recognition method that is able to adapt automatically to any type of images, and in consequence can be considered a general method of image recognition. We illustrate the method with the problem of face recognition and its performance is compared with other face recognition methods. The general method consists of two phases: the training and the recognition phases.

The training phase consists of two steps (Fig. 1):

1. The edge detection of each training image using a fuzzy inference system:
   As we can see in the training phase, the input dataset contains only the original images of the database, which means the images of the training set without modifications. Note that one disadvantages of many existing methods for image recognition is that they need to include a phase of feature extraction or another type of pre-processing technique directly dependent on the type of image to be recognized [1–7]. Since the source of the images is not analyzed, then we do not need to specify parameters corresponding to the subject, like nose or eyes position; this feature allow us to apply the method to recognize different kinds of objects instead of only human faces. We describe the development and comparison of three edge detection methods: Sobel, type-1 fuzzy inference system (FIS1) [8] and interval type-2 fuzzy inference system (FIS2) [9]. In both of the FIS approaches not only image edge detection is performed, also the values of each pixel are normalized for made the training of the neural networks faster.

* Corresponding author.
E-mail addresses: epmelin@hafsamx.org, pmelin@tectijuana.mx (P. Melin).

0020-0255/$ - see front matter © 2008 Elsevier Inc. All rights reserved.
doi:10.1016/j.ins.2008.11.018

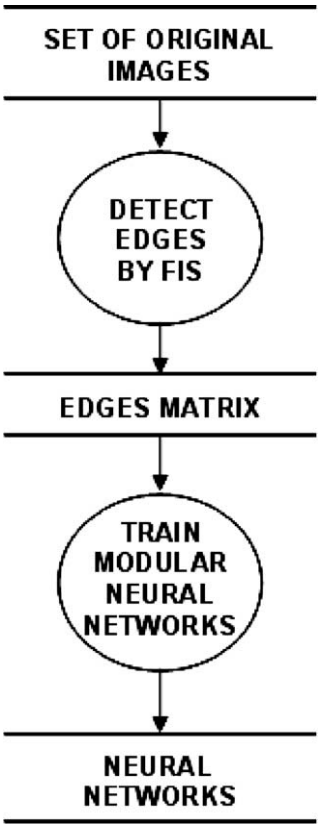


Fig. 1. Steps in the training phase.

2. The training of the modular neural networks:
   The modular structure consists of three monolithic feed-forward neural networks. The training data of each module is one of three parts of the edge vectors divided after the edge detection step. The outputs of the training phase are the modular neural networks with the weights modified during the training step.

The recognition phase consists of four steps (Fig. 2):

1. The edge detection of a particular image to recognize:
   In order of test the recognition phase, the input must be a particular image at a time, but all the images on the database can be used for testing. This particular image can belong to the training set or not.
2. The simulation of the edge vector in the modular neural network.
3. The estimation of each module's relevance using a fuzzy inference system with the values of simulation vectors as input:
   The problem then becomes how to combine the simulation vectors of the three modules in order to recognize the maximum number of images possible. The recognition phase is completely automatic because the method estimates the relevance of each module using a fuzzy inference system without human intervention.
4. Selection of the recognized image from the fusion of the simulation vectors using the Sugeno integral with the relevance of each module as fuzzy densities:
   The final decision is made using the Sugeno integral, which is used to combine the simulation vectors into a unique vector, then at the end of the method the system decides the best choice of recognition in the same manner than a monolithic neural network does it, but with the problem of complexity reduced. All the algorithms are coded with Matlab© [10].

The target data for the supervised training algorithm is an identity matrix, where the elements with a value equal to 1 represent the position of each subject to recognize. In the simulation phase, if a particular module (neural network) does not contain any element with value near 1; this shows a module with weak recognition performance.

We use the maximum simulation value of each module as the input to the FIS for estimating its relevance. The outputs of the FIS are the relevance rates of the modules, which is similar to rating a recognition expert. The benefit of using the Sugeno integral is that the relevance rates can be used as fuzzy densities for the modules.

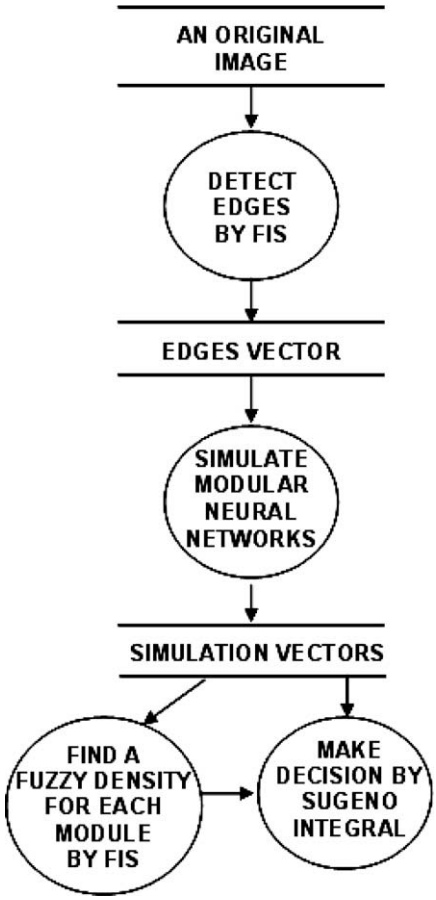


Fig. 2. Steps in the recognition phase.

Another important feature of the Sugeno integral is the use of fuzzy λ measures, because they perform the generalization of lower and upper probability. Lower probability occurs when the sum of the fuzzy densities of all the information sources is less than 1, and upper probability occurs when the sum of the fuzzy densities of all the information sources is greater than 1. This feature is very useful because both cases can occur and the Sugeno integral is the ideal operator to manage this efficiently.

The goal is to find the fuzzy densities for the Sugeno integral to rank the relevance of the modules in the decision process in order to obtain the best image recognition rate. This problem is solved by building a FIS to estimate the fuzzy densities using only the simulation vectors as input data. This step is tested with two fuzzy logic systems, using type-1 and interval type-2 fuzzy logic respectively.

In this paper, fuzzy logic help us model the problems of edge detection and estimating the optimal fuzzy densities in the decision process, which are two different problems that can be solved with the same approach to improve the results of the problem of image recognition.

The paper is organized as follows: In section 2, some image processing fundamentals are mentioned, and three different edge detection methods are presented: Sobel, type-1 fuzzy inference system and type-2 fuzzy inference system. In Section 3, a modular neural network method for image recognition using the Sugeno integral as aggregation operator is presented. In this section, the method to estimate the relevance of each module using fuzzy logic is also detailed. In Section 4, the results obtained with the recognition method are presented. In Section 5, we compare our results with the recognition rates reported by some authors of existing methods.

2. Edge detection

A digital image can be defined as a function of two real variables $I(x,y)$ or two discrete variables $I[m,n]$. The fundamental operations in digital image processing can be classified into four categories: operations based on the image histograms, on simple mathematics, on convolution, and on mathematical morphology. Further these operations can also be described in terms of their implementation. A summary of these operations is given below [11].

- *Histogram-based operations:* If an image contains pixel values that do not use the available dynamic range, then it can be easily observed in the histogram of brightness. This situation can be corrected by stretching the histogram over the available dynamic range. To compare two or more images on a specific basis, such as texture, it is common to first normalize their histograms to a standard histogram. This can be especially useful when the images have been acquired under different conditions. The most common histogram normalization technique is histogram equalization that consists of changing the histogram through the use of a function into a histogram that is constant for all brightness values. This would correspond to a brightness distribution where all values are equally probable.
- *Mathematics-based operations:* These operations can be done with binary arithmetic and ordinary arithmetic. In the binary case, there are two brightness values 0 and 1. The binary operations are based on Boolean arithmetic like NOT, AND, OR, XOR and SUB. In the ordinary arithmetic case, the gray value point operations that form the basis for image processing are based on ordinary mathematics and include: +, −, *, /, log, exp, sqrt, sin, cos, tan, inversion.
- *Convolution-based operations:* Convolution, a mathematical local operation is widely applied with filters to digital images. This operation is applied when a window of finite size and shape is scanned across the image. The output pixel value is the weighted sum of the input pixels within the window where the weights are the values of the filter assigned to every pixel of the window itself. The window with its weights is called the convolution kernel.
- *Smoothing operations:* These algorithms are applied in order to reduce noise and/or to prepare images for further processing such as segmentation. We distinguish between linear and non-linear algorithms where the former are amenable to analysis in the Fourier domain and the latter are not. We also distinguish between implementations based on a rectangular support for the filter and implementations based on a circular support for the filter.
- *Derivative-based operations:* Just as smoothing is a fundamental operation in image processing, so is the ability to take one or more spatial derivatives of the image. The fundamental problem is that, according to the mathematical definition of a derivative, this cannot be done. A digitized image is not a continuous function $a(x,y)$ of the spatial variables but rather a discrete function of the integer spatial coordinates. As a result the used algorithms can only be seen as approximations to the true spatial derivatives of the original spatially continuous image. The derivative multiplies the signal spectrum by either u or v. This means that high frequency noise will be emphasized in the resulting image. The general solution to this problem is to combine the derivative operation with one that suppresses high frequency noise, in short, smoothing in combination with the desired derivative operation.
- *Morphological-based operations:* These operations are based in an alternative definition of an image: The image consists of a set of either continuous or discrete coordinates. In a sense the set corresponds to the points or pixels that belong to the objects in the image.

Edge detection in digital images has the opposite effect of noise elimination; consists in emphasize pixels with gray tone that are different than its neighbors. In an image the edges corresponds to the objects boundaries in the image.

There are many ways to perform edge detection. However, most of them can be grouped into two categories, gradient and Laplacian methods. The gradient method detects the edges by looking for the maximum and minimum in the first derivative of the image. The Laplacian method searches for zero crossings in the second derivative of the image to find the edges. Methods based on the gradient: find places where the first derivative of the intensity is greater than a defined threshold, like Sobel, Prewit, Roberts and isotropic. Methods based on the Laplacian: find pixels where the second derivative of the intensity has a zero crossing, like the method of Marrs–Hildreth [12].

Some of the previously mentioned methods have been enhanced with type-1 fuzzy logic [8,9,13,14], Fractal theory [15] or neural networks [16–18]. In this paper, we show the implementation and tests with an edge detector based on gradient magnitude, but enhanced with type-1 and interval type-2 fuzzy inference systems. The tests are shown using the ORL database [19] of faces in order to prepare the training data for an image recognition system based in modular neural networks [20].

2.1. Sobel's method for edge detection

The Sobel operator is applied to a digital image in gray scale, calculating the gradient of the brightness intensity of each pixel, giving the direction of the greater possible increase of black to white, in addition calculates the amount of change in that direction. The operator uses masks to perform a two dimensional spatial gradient measurement on an image. Typically it is used to find the approximate absolute gradient magnitude at each point in an input grayscale image.

The Sobel operator is a pair of 3×3 convolution masks, one estimating the gradient in the x-direction (columns) (1) and the other estimating the gradient in the y-direction (rows) (2).

A convolution mask is usually much smaller than the actual image. As a result, the mask is slid over the image, manipulating a square of pixels at a time [8]

$$\mathrm{Sobel}_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, \tag{1}$$

$$\mathrm{Sobel}_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}. \tag{2}$$

If we define $I_{m,n}$ as the matrix with m rows and n columns, where the image source is stored, then g_x and g_y are two matrices with same dimension than I, which at each element contain the horizontal and vertical derivative approximations. The latter are computed by the following equations:

$$g_x = \sum_{i=1}^{i=3}\sum_{j=1}^{j=3} \text{Sobel}_{x,i,j} * I_{r+i-2,c+j-2} \quad \begin{matrix}\text{for } r = 1,2,\ldots,m \\ \text{for } c = 1,2,\ldots,n,\end{matrix} \tag{3}$$

$$g_y = \sum_{i=1}^{i=3}\sum_{j=1}^{j=3} \text{Sobel}_{y,i,j} * I_{r+i-2,c+j-2} \quad \begin{matrix}\text{for } r = 1,2,\ldots,m \\ \text{for } c = 1,2,\ldots,n.\end{matrix} \tag{4}$$

The operation performed with Eqs. (3) and (4) is called convolution. Eqs. (3) and (4) calculate the gradients along x-axis and y-axis, r and c are the indexes of rows and columns of the pixels position on matrix I. In the Sobel method the gradient magnitude g is calculated by Eq. (5) as is shown in Fig. 3, where $\otimes$ represents the convolution operator.

$$g = \sqrt{g_x^2 + g_y^2}. \tag{5}$$

2.2. Type-1 fuzzy inference system for edge detection

The type-1 fuzzy inference system (FIS1) for edge detection uses as inputs the images generated by the Sobel operators, as in Sobel's method, as we can see in Fig. 4 [8,9].

2.2.1. Inputs to the type-1 FIS

For the type-1 fuzzy inference system, 3 inputs can be used, 2 of them are the gradients with respect to the x-axis and y-axis, calculated with (3) and (4), which we call DH and DV, respectively. The third variable M is the image after the application of a low-pass filter hMF in Eq. (6); this filter allows to detect image pixels belonging to regions of the input were the mean gray level is lower. These regions are proportionally affected more by noise, which is supposed to be 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 edge detection by effect of noise

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

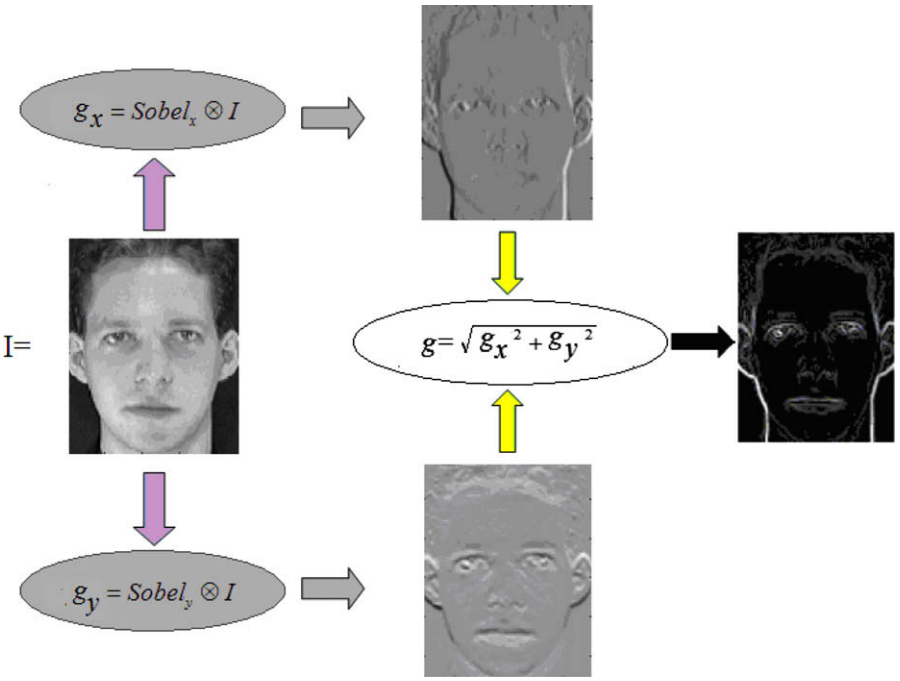


Fig. 3. The process of edge detection using the Sobel method.

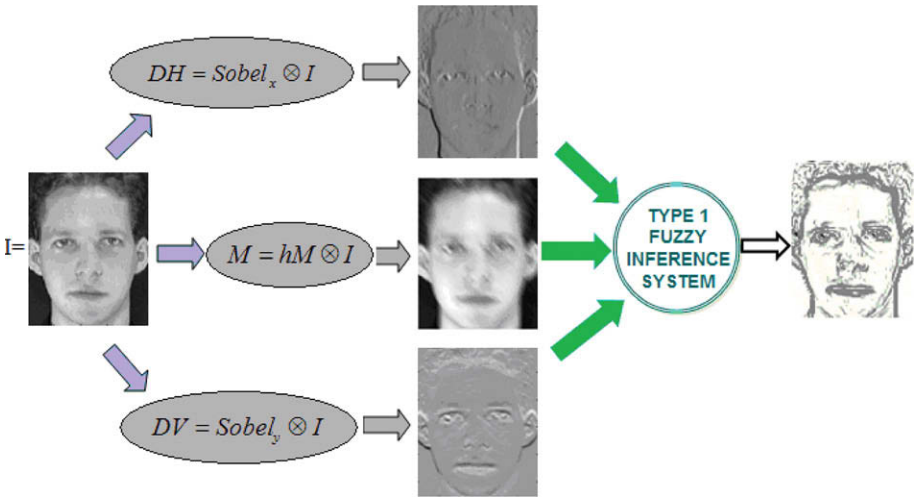


Fig. 4. The type-1 FIS for edge detection implemented in the Matlab fuzzy logic toolbox.

The inputs to the type-1 FIS are defined with the expressions (7)–(9):

$$DH = \text{Sobel}_x \otimes I, \tag{7}$$

$$DV = \text{Sobel}_y \otimes I, \tag{8}$$

$$M = \text{hMF} \otimes I, \tag{9}$$

where $\otimes$ is the convolution operation, and I the image to be applied to the filter.

2.2.2. Fuzzy variables

For all the fuzzy variables, the membership functions are of Gaussian type. According to the performed tests, the values of DH and DV, are between 0 and 500. These values correspond to the magnitude of change in any direction. To obtain a better estimation of the amount of change we use its absolute values, which means considering only the magnitude of change without its direction. The ranges in the *x*-axis are adjusted as it is shown in Figs. 5 and 6. The membership functions corresponding to the fuzzy system are: LOW: Gaussmf(60,0), MEDIUM: Gaussmf(60,230), HIGH: Gaussmf(50,365) [21]. The membership function LOW corresponds to a magnitude of change near 0, the membership function MEDIUM to a magnitude of change near 230, and the membership function HIGH to a magnitude of change near 365. It is possible to find few magnitude changes greater than 365, but we prefer to ignore it, with the purpose of considering more edge areas even if the magnitude of change is not the largest.

In the case of variable *M*, the tests gave values in the range from 0 to 255, and thus the range in the *x*-axis is adjusted, as it is appreciated in Fig. 7, in which the membership functions corresponding to the fuzzy system are shown: LOW: Gaussmf(50,0), MEDIUM: Gaussmf(50,127), HIGH: Gaussmf(50,255) [21].

The output variable EDGES is used to find and normalize the edges to any range of required values. In Fig. 8, the output variable EDGES is shown, which also has been adjusted to have a range between −4.5 and 1.5, since it is the better range of

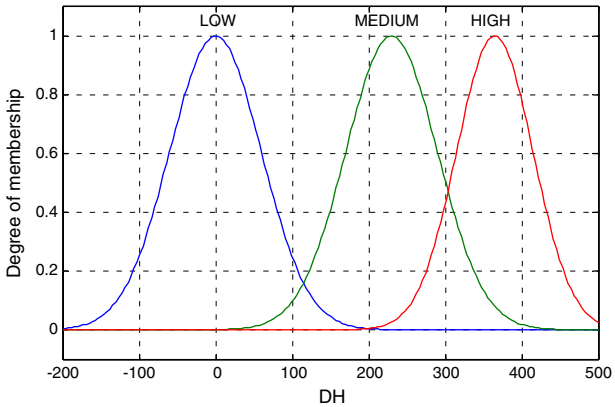


Fig. 5. Type-1 fuzzy logic input variable DH.

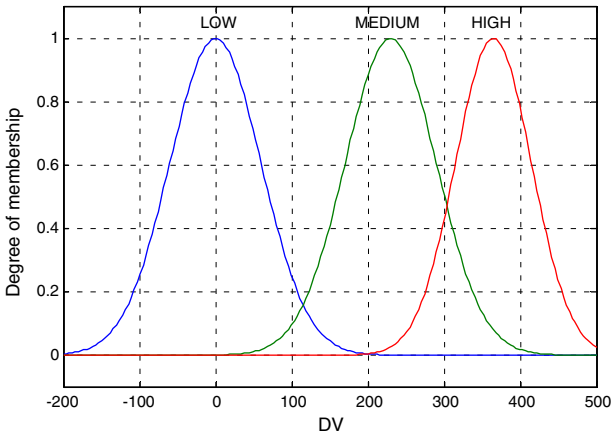


Fig. 6. Type-1 logic input variable DV.

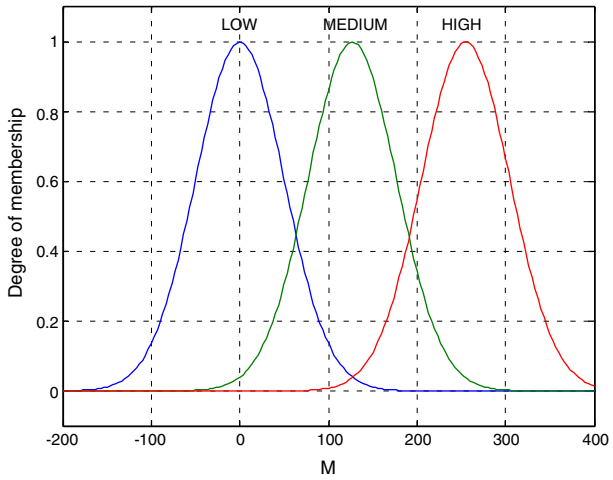


Fig. 7. Type-1 fuzzy logic input variable *M*.

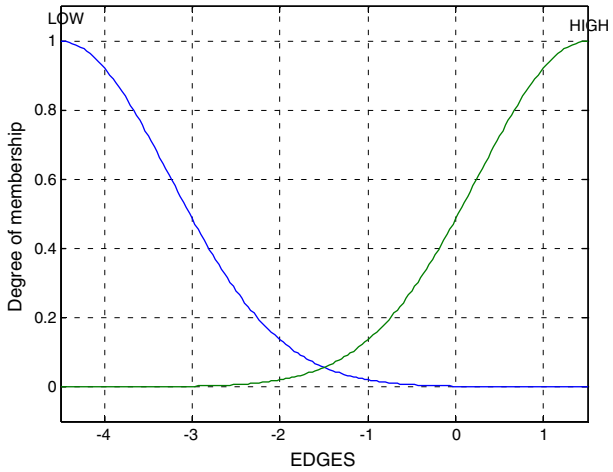


Fig. 8. Type-1 fuzzy logic output variable EDGES.

values to normalize the edges matrix for a good performance in next step of the method, which is the training with neural networks.

2.2.3. Fuzzy inference rules

The five fuzzy rules that allow the evaluation of the input variables, so that the output image displays the edges of the image in gray tones are near black (tone LOW), whereas the background is in tones near white (HIGH tone).

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)

The solution surface for DV, DH and EDGES is shown in Fig. 9. The plot describes high values for EDGES when the magnitudes of DH and DV are near 0, which means the clear gray tones or background of the image, and low values for the EDGES when the magnitudes of DH and DV are between 200 and 400, that means the dark gray tones or EDGES of the image.

The solution surfaces for *M*, DH and EDGES and *M*, DV and EDGES are shown in Figs. 10 and 11, respectively.

2.3. Interval type-2 fuzzy inference system for edge detection

The interval type-2 fuzzy inference system (FIS2) for edge detection has the structure parameters and rules based on the FIS1, which is required to be able to make a comparison of both results (Fig. 12). The tests with the FIS2 are produced using a Matlab program, which creates an interval type-2 inference system (Mamdani) [22,23].

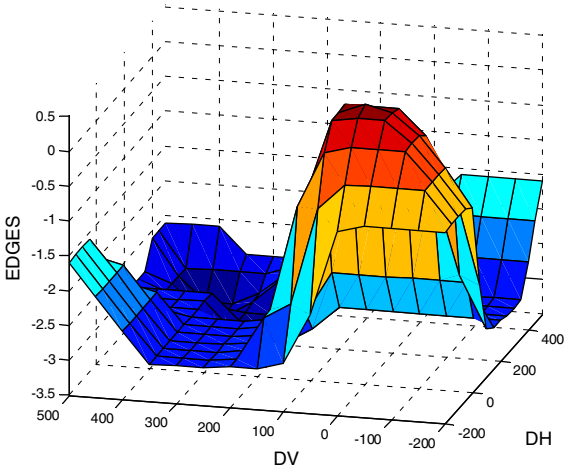


Fig. 9. Solution surface for DV, DH and EDGES in type-1 FIS for edge detection.

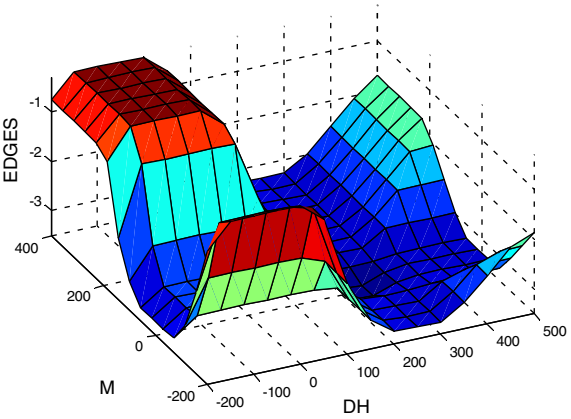


Fig. 10. Solution surface for *M*, DH and EDGES in type-1 FIS for edge detection.

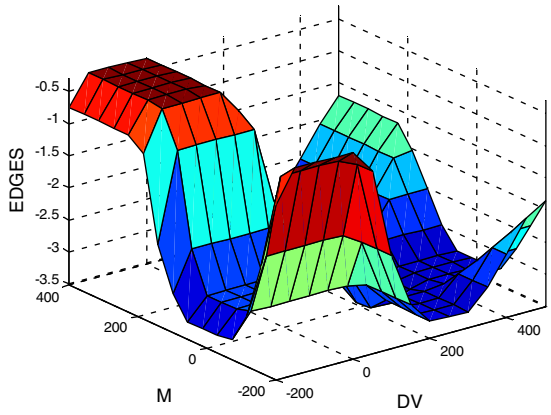


Fig. 11. Solution surface for *M*, DV and EDGES in type-1 FIS for edge detection.

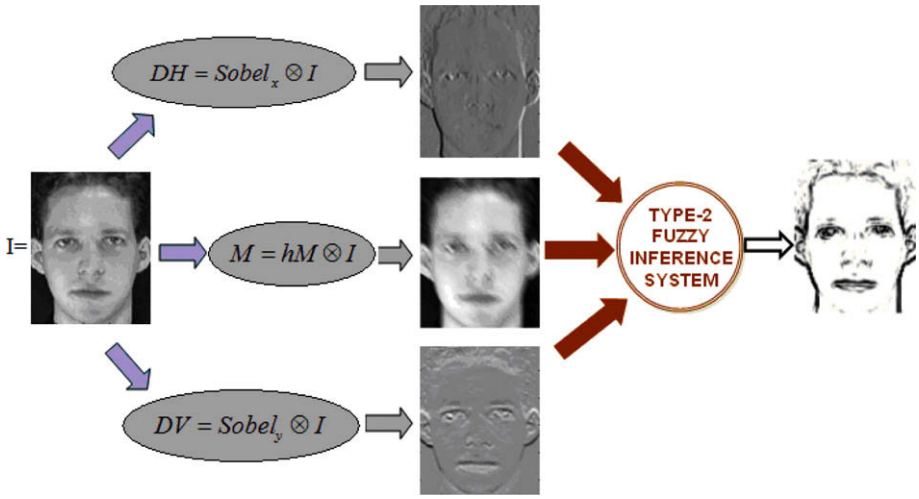


Fig. 12. Type-2 FIS for edge detection performed using the interval type-2 fuzzy logic toolbox.

The above-mentioned program is part of the interval type-2 fuzzy logic Toolbox developed in our research group [24]. This toolbox allows us to create interval type-2 fuzzy variables, as shown in Fig. 13. The toolbox is completely interactive in a graphic environment similar to the Matlab type-1 fuzzy logic toolbox, for this reason the experimentation is very accessible, because we only select the Mamdani or Sugeno method, different types of membership functions and parameters, reduction method or defuzzification algorithms and the fuzzy inference system can be created and tested until the required results are achieved.

Particularly for the fuzzy logic edge detector we can select the defuzzification method and parameters making many tests until the edge image achieves an homogenous and clear background without losing fashion details, and finding the gray tones range to obtain a mean near 0 and standard deviation near 1, which is needed for better performance in the training data of the neural networks.

The width of the footprint of uncertainty (FOU) chosen for each membership function is the one that produced better results after several experiments [25]. The solution surfaces of the rules are shown in Figs. 14–16.

Let us remember that the purpose of the edge detector is not only to discard the background area, we also need that the training data have the values designed for an easy training of the neural networks. The images obtained using the Sobel detector have the problem of a very wide range of values, between 0 and 700; even discarding the values near 0, which means the darker background, the rest of the edges values are distributed between 100 and 700, which is a range very hard to manage with a neural network using some supervised back-propagation training algorithm. Using the FIS1 or the FIS2 as edge detectors we can obtain edges images with more detail and with the ideal values for an excellent performance in neural networks training.

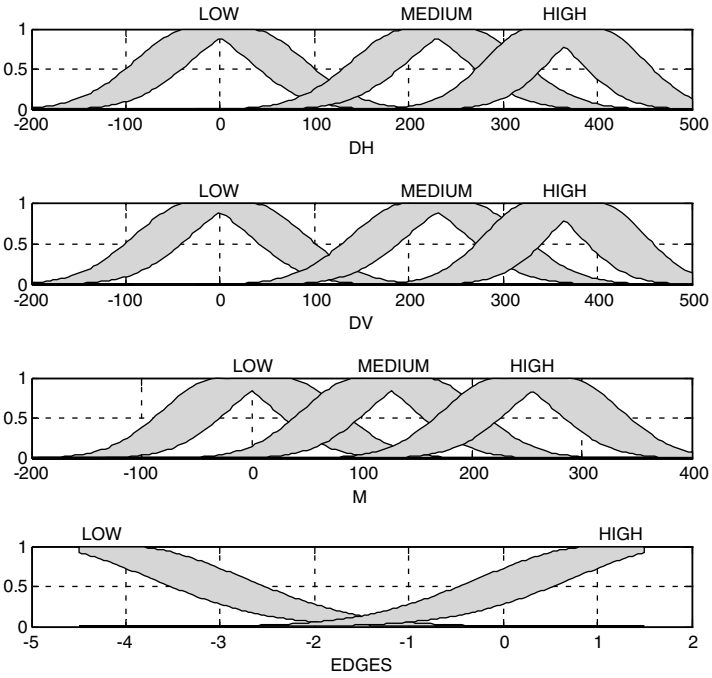


Fig. 13. Membership functions for variables DH, DV, *M* and EDGES in the type-2 FIS edge detector.

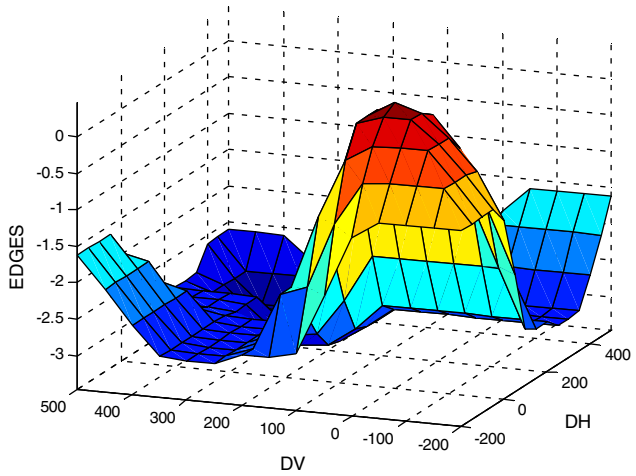


Fig. 14. Solution surface for DV, DH and EDGES in the type-2 FIS for edge detection.

3. Modular neural networks

3.1. Modular structure

The modular structure is designed for a database of images, like the Olivetty Research Laboratory database of faces (ORL) [19], but it is not limited to this data. To measure the recognition rate in an objective form, the modular neural networks must be trained with the set of images in a random ordered fashion, this process is called a random permutation.

The design of the modular neural network consists of three monolithic feed-forward neural networks, and each one is trained with a supervised method with the first seven samples of the 40 images of ORL [26,27].

The edge vector for each image is accumulated into a matrix, as shown in the scheme of Fig. 17. The complete matrix of images is divided into three parts, each module is trained with a corresponding part, with the some rows for overlapping.

The target to the supervised training method consists of one identity matrix with dimensions 40×40 for each sample, building one matrix with total dimensions $(40 \times 40 \times 7)$, as shown in (Fig. 18) [28].

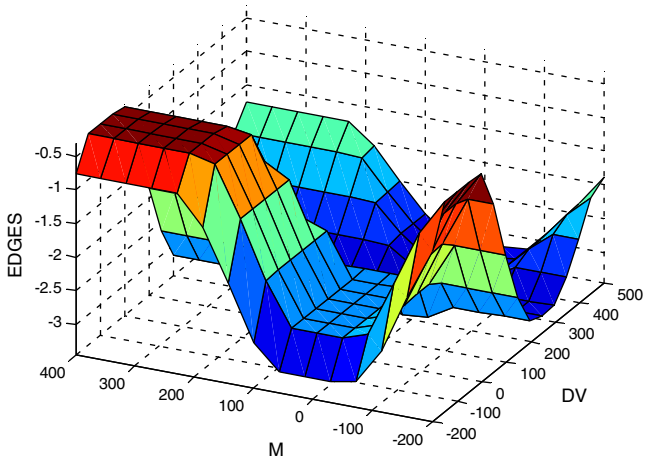


Fig. 15. Solution surface for *M*, DV and EDGES in type-2 FIS for edge detection.

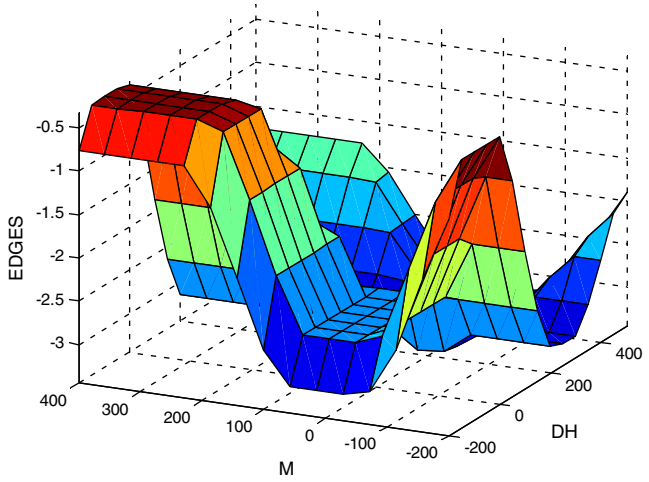


Fig. 16. Solution surface for *M*, DH and EDGES in type-2 FIS for edge detection.

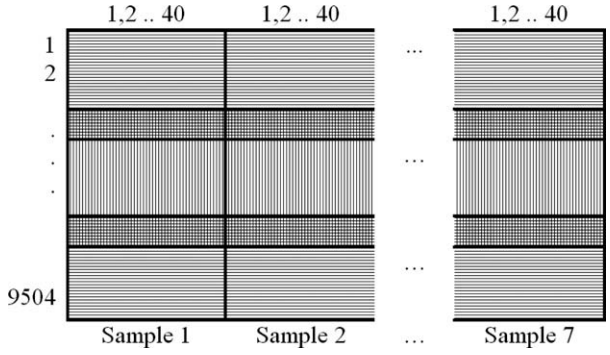


Fig. 17. Input: 7 images for each person.

Each monolithic neural network has the same structure (Fig. 19). Training is done under the same conditions using the neural network toolbox of Matlab:

1. Three hidden layers with 200 neurons and *tansig* transfer functions are used.
2. The output layer with 40 neurons and *purelin* transfer functions is used.
3. The training function is gradient descent with momentum and adaptive learning rate back-propagation (*traingdx*).

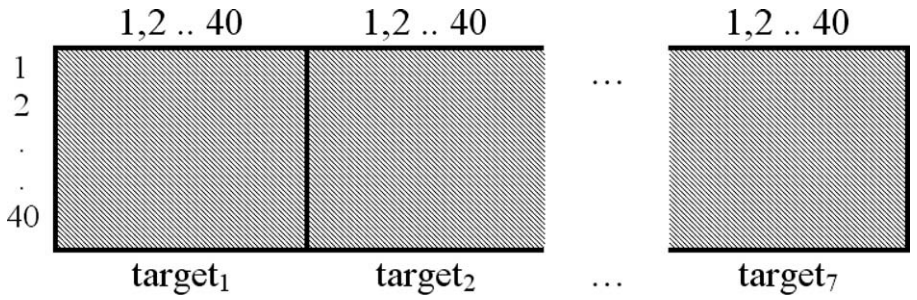


Fig. 18. Target: one identity matrix with dimensions 40×40 for each sample.

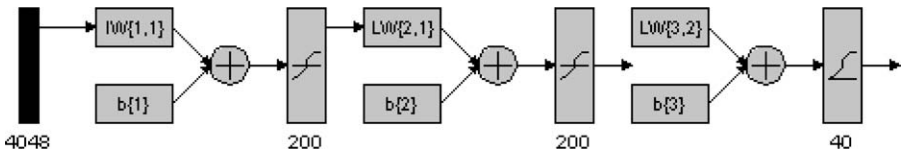


Fig. 19. Structure of each monolithic neural network.

3.2. Modules training

The next code segment is used to train each module, where *newff* is a Matlab neural network toolbox function to create a feed-forward back-propagation network with the specified parameters [29].

```
layer1=200; layer2=200; layer3=40;
net=newff(minmax(p),[layer1,layer2,layer3],{'tansig','tansig','logsig'},'traingdx');
net.trainParam.goal=1e-5;
net.trainParam.epochs=1000;
```

3.3. Modules fusion

We use Matlab© to develop all the computer programs. The simulation of each module includes the 400 images of the ORL database, building a matrix with the results of the simulation of each one, as it is shown in Fig. 20. These matrices are stored on a file to be analyzed later for the combination of results [30].

In the simulation, columns corresponding to the images in the training data, the position with a value near one is always selected correctly. However, some outputs of the training data have very low values in all positions, reason why it is very important to have a good combination method to recognize more images.

3.3.1. The Sugeno integral for modules fusion

The fuzzy integral is an operator introduced in 1974 by Sugeno. This operator is used to solve problems of multi-criteria decision making, where the information that is combined is based on fuzzy measures determined by an expert.

The goal is the simulation of the human process for the integration of different source of information.

Fuzzy measures are functions applied to fuzzy sets and they consist of different coefficients $\mu(x_n)$ called fuzzy densities. Each fuzzy density rates the relevance of the different sets and their combinations, in order to satisfy certain hypothesis.

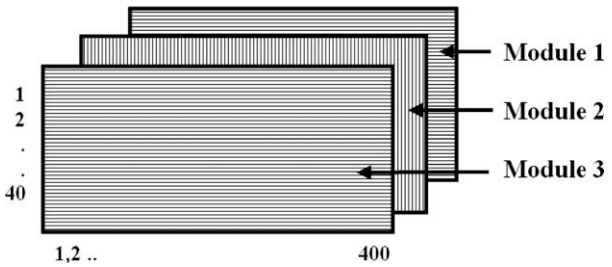


Fig. 20. Simulation matrices for the three modules.

There are two types of fuzzy integral: Choquet fuzzy integral (10) and Sugeno fuzzy integral (11) [31,32]

$$h(\sigma_1,\ldots,\sigma_n)=\sum_{i=1}^{n}(\sigma_i-\sigma_{i-1})\mu(\{x_i,\ldots,x_n\}), \tag{10}$$

$$h(\sigma_1,\ldots,\sigma_n)=\max_{i=1}^{n}(\min(\sigma_i,\mu(x_i,\ldots,x_n))), \tag{11}$$

where $\sigma_i=\sigma(x_i)$ and $0\leqslant\sigma_1\leqslant\cdots\leqslant\sigma_n\leqslant 1$.

A fuzzy measure μ, with respect to the dataset X, must satisfy the following conditions:

(1) $\mu(X)=1$, $\mu(\emptyset)=0$
(2) If $S\subseteq T$, then $\mu(S)\leqslant\mu(T)$

In condition (2) S and T are subsets of X.

A fuzzy measure is a Sugeno measure or λ-fuzzy, if it satisfies the condition (12) of addition for some $\lambda>-1$

$$\mu(S\cup T)=\mu(S)+\mu(T)+\lambda\mu(S)\mu(T) \tag{12}$$

λ can be calculated with the following Eqs. (13) or (9):

$$\mu(S)=\left[\prod_{x\in S}(1+\lambda\mu(\{x\}))\right]\Big/\lambda, \tag{13}$$

$$\lambda+1=\prod_{i=1}^{n}(1+\lambda\mu(\{x_i\})). \tag{14}$$

The value of the parameter λ is determined by the conditions of the following theorem [33–35]:

Theorem 1. *Let $\mu(\{x\})<1$ for each $x\in X$ and let $\mu(\{x\})>0$ for at least two elements of X. Then Eq. (3) determines a unique parameter λ in the following way:*

- *If $\sum_{x\in X}\mu(\{x\})<1$, then λ is equal to a unique root of the equation in the interval $(0,\infty)$.*
- *If $\sum_{x\in X}\mu(\{x\})=1$, then $\lambda=0$; that is the unique root of the equation.*
- *If $\sum_{x\in X}\mu(\{x\})>1$, then λ is equal to a unique root of the equation in the interval $(-1,0)$.*

Any λ-measure is thus completely determined by its values on all singletons $\mu(\{x\})$, for all $x\in X$, the values of λ are determined with Eq. (3). According to Theorem 1, three situations are distinguished:

- *If $\sum_{x\in X}\mu(\{x\})<1$, which means that μ qualifies as a lower probability, $\lambda>0$.*
- *If $\sum_{x\in X}\mu(\{x\})=1$, which means that μ is a classical probability, $\lambda=0$.*
- *If $\sum_{x\in X}\mu(\{x\})>1$, which means that μ qualifies as an upper probability, $\lambda<0$.*

The method to calculate Sugeno measures, carries out the calculation in a recursive way, using (11) and (12).

$$\mu(A_1)=\mu(x_1), \tag{15}$$

$$\mu(A_i)=\mu(x_i)+\mu(A_{i-1})+\lambda\mu(x_i)\mu(A_{i-1}), \tag{16}$$

where $1<i\leqslant n$, and the values of $\mu(xi)$ correspond to the fuzzy densities determined by an expert.

A fundamental constraint to use the recursive formulas (11) and (12) is the reordering of the fuzzy densities. The fuzzy densities must be ordered with respect to the descendent order of the respective values to combine [36,37].

3.4. Fuzzy inference systems to determine fuzzy densities

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 this is the unique available information. For this reason we analyze the simulation matrix in many tests and decided that each of the inputs to the FIS corresponds to the maximum value of each column corresponding to the simulation of each module of the 400 images. The variables m_1, m_3 and m_3 correspond to the simulation values to combine, $\max_1$, $\max_2$ and $\max_3$ correspond to the maximum values of the simulation vector, and d_1, d_2 and d_3 the fuzzy densities for each module, as shown in Fig. 21.

Each output corresponds to one fuzzy density for ranking each module to perform the fusion of results later with the Fuzzy Sugeno Integral. The inference rules find 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 listed below.

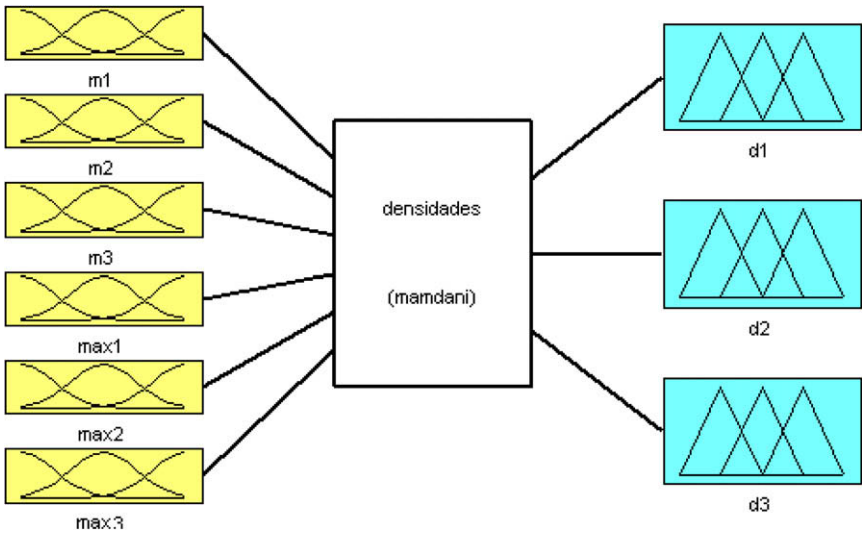


Fig. 21. Variables for the FIS to find the fuzzy densities.

1. If (ml is LOW) and (maxl is LOW) then (dl is LOW) (1)
2. If (ml is MEDIUM) and (maxl is MEDIUM) then (dl is MEDIUM) (1)
3. If (ml is HIGH) and (maxl is HIGH) then (dl is HIGH) (1)
4. If (m2 is LOW) and (max2 is LOW) then (d2 is LOW) (1)
5. If (m2 is MEDIUM) and (max2 is MEDIUM) then (d2 is MEDIUM) (1)
6. If (m2 is HIGH) and (max2 is HIGH) then (d2 is HIGH) (1)
7. If (m3 is LOW) and (max3 is LOW) then (d3 is LOW) (1)
8. If (m3 is MEDIUM) and (max3 is MEDIUM) then (d3 is MEDIUM) (1)
9. If (m3 is HIGH) and (max3 is HIGH) then (d3 is HIGH) (1)
10. If (ml is LOW) and (maxl is MEDIUM) then (dl is LOW) (1)
11. If (ml is MEDIUM) and (maxl is HIGH) then (dl is MEDIUM) (1)
12. If (m2 is LOW) and (max2 is MEDIUM) then (d2 is LOW) (1)
13. If (m2 is MEDIUM) and (max2 is HIGH) then (d2 is MEDIUM) (1)
14. If (m3 is LOW) and (max3 is MEDIUM) then (d3 is LOW) (1)
15. If (m3 is MEDIUM) and (max3 is HIGH) then (d3 is MEDIUM) (1)
16. If (ml is LOW) and (maxl is HIGH) then (dl is LOW) (1)
17. If (m2 is LOW) and (max2 is HIGH) then (d2 is LOW) (1)
18. If (m3 is LOW) and (max3 is HIGH) then (d3 is LOW) (1)

According to exhaustive tests made in the simulation matrices, we know that the recognition rate of the images corresponding to the training set is 100%. Therefore the interest is focused on the recognition of the samples that do not belong to the training set, in other words samples 8–10 of the ORL database [38].

The parameters for the Sugeno Fuzzy Integral to be estimated are the Fuzzy Densities, with values between 0 and 1 for each module, which determine the relevance of each module [39,40].

The parameter λ, according to the fuzzy measures theory depends on the values of the fuzzy densities, and can be obtained finding the roots of a function:

$$f(\lambda) = (1 + \mu(x_1)\lambda)(1 + \mu(x_2)\lambda)\ldots(1 + \mu(x_n)\lambda) - (1 + \lambda).$$

3.4.1. FIS1 optimization to find fuzzy densities

The membership functions in Fig. 22 and the solution surface in Fig. 23 show the system's behavior.

3.4.2. FIS2 to find fuzzy densities

To compare the FIS1 and FIS2 to find the fuzzy densities, we add a FOU = 0.2, to the same fuzzy variables in FIS1, as we can see in Figs. 24 and 25.

3.4.3. Calculation of the Sugeno integral

After the simulation of each image divided into three modules we have three simulation vectors of length 40 to combine. The value for each element of the resulting vector is the Sugeno Integral for the values corresponding to the same position in

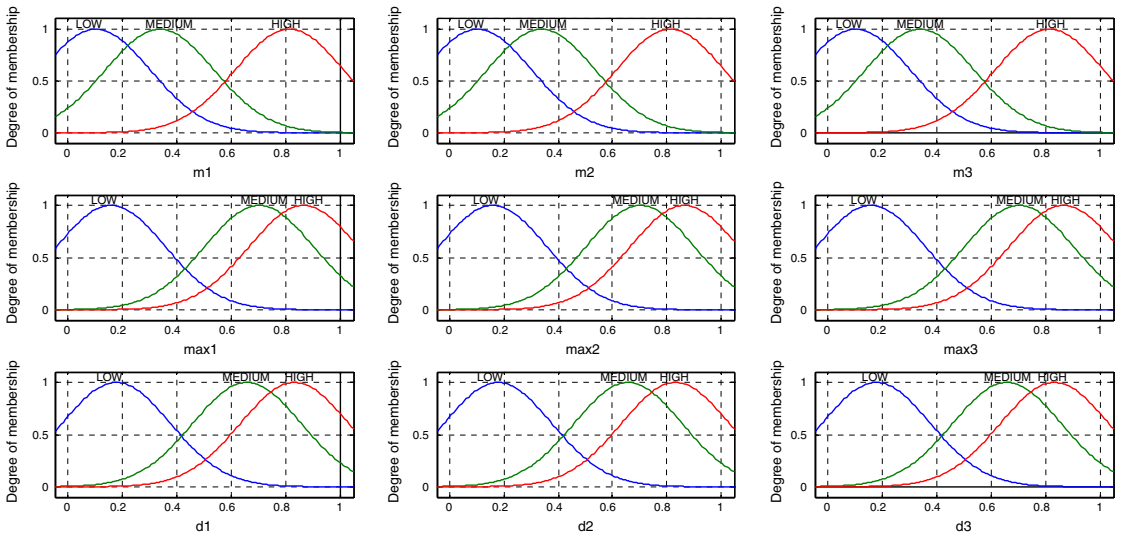


Fig. 22. Variables of the FIS1 for the estimation of fuzzy densities.

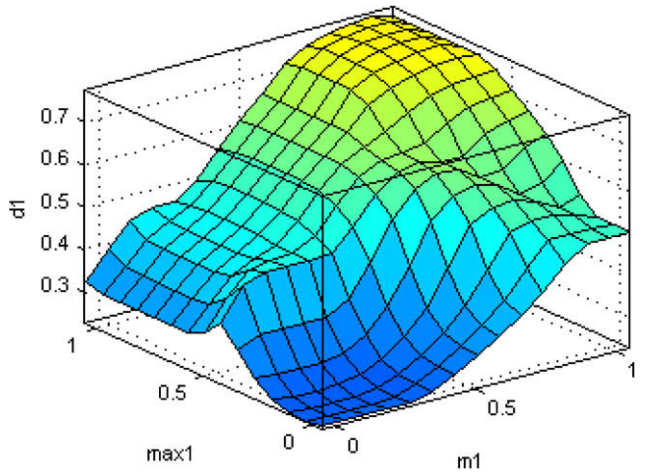


Fig. 23. Solution surface for variables $\max_1$, m_1 and d_1 in Type-1 FIS for the estimation of fuzzy densities.

the three vectors to combine. Consider the following values corresponding to the simulation of sample 8 of person number 13, this sample is not on the training data.

	m1	m2	m3
1			
2			
3			
:			
13	0.1575	0.0094	0.0247
14			
:			
40			

max1	max2	max3
0.1575	0.0286	0.0574

The fuzzy densities are inferred by the FIS as follows: First the FIS determines the fuzzy densities for each module. Consider the example were the fuzzy densities are $d_1 = 0.3069$, $d_2 = 0.1788$, and $d_3 = 0.1890$. Next we need to solve the function $f(\lambda) = (1 + 0.3069\lambda)(1 + 0.1788\lambda)(1 + 0.1890\lambda) - (1 + \lambda)$ according with Eq. (11) (Fig. 26).

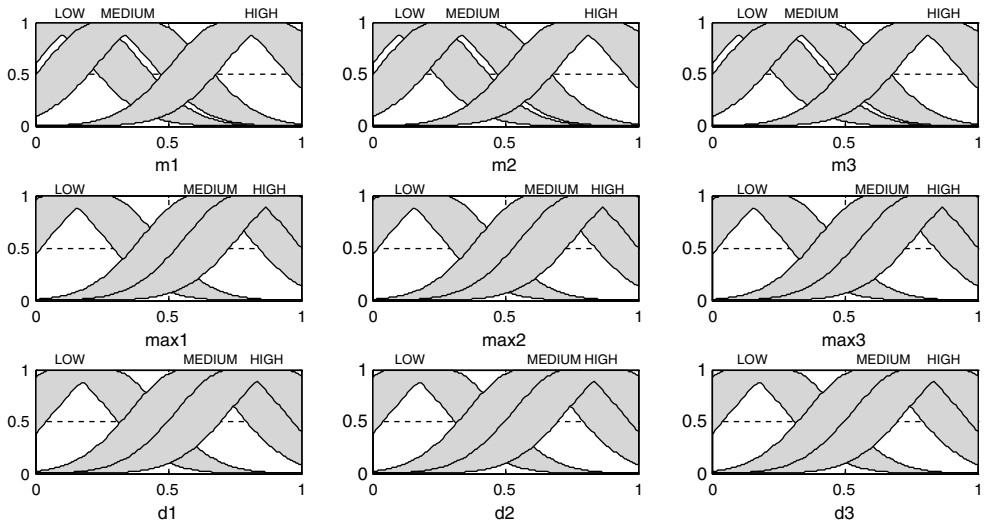


Fig. 24. Variables for the FIS2 for find fuzzy densities, using the same membership functions than the FIS1 adding a FOU = 0.2

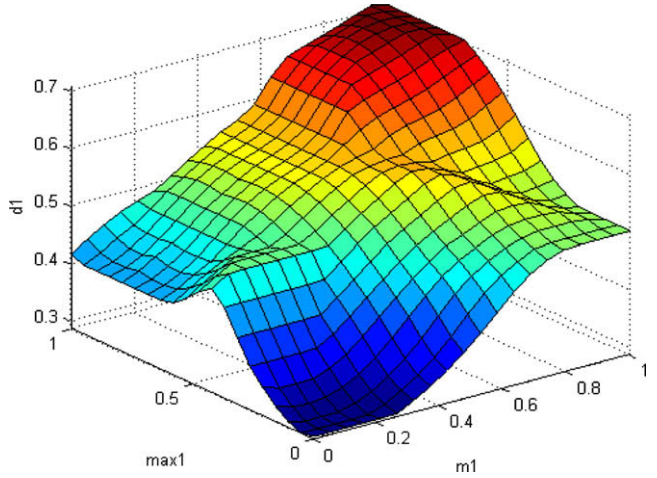


Fig. 25. Solution surface for $\max_1$, m_1 and d_1 in type-2 FIS to find fuzzy densities.

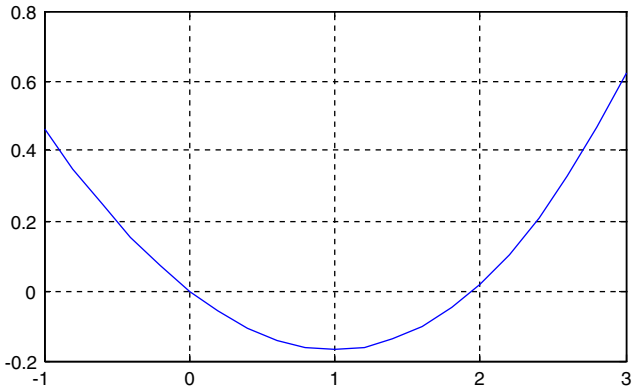


Fig. 26. Plot for $f(\lambda) = (1 + 0.3069\lambda)(1 + 0.1788\lambda)(1 + 0.1890\lambda) - (1 + \lambda)$.

The value of λ can be calculated by Eq. (11), using a numerical method such as Newton–Raphson or bisection, to find the root of $f(\lambda)$ shown in Fig. 26. To solve this equation we used the Matlab function "fzero" that finds a root of a continuous function of one variable. The solution is $\lambda = 1.9492$ in this example, and the Sugeno Measures can be constructed using the recursive formulas (12) and (13), but before this calculation the fuzzy densities must be sorted descendent by $\sigma(x_i)$, that is shown in Table 1

$$\mu(A_1) = \mu(x_1),$$
$$\mu(A_i) = \mu(x_i) + \mu(A_{i-1}) + \lambda\mu(x_i)\mu(A_{i-1}),$$

$$\mu(A_1) = 0.3069,$$
$$\mu(A_2) = 0.1890 + 0.3069 + (1.9492)(0.1890)(0.3069) = 0.6090,$$
$$\mu(A_3) = 0.1788 + 0.6090 + (1.9492)(0.1788)(0.6090) = 1.$$

The Fuzzy Sugeno Integral, can now be calculated using (6):

$$h(0.9989, 0.0492, 0.0249) = \max(\min(0.9989, 0.3069), \min(0.0492, 0.6090), \min(0.0249, 1)),$$
$$h(0.9989, 0.0492, 0.0249) = \max(0.3089, 0.0492, 0.0249) = 0.3089.$$

As can be seen in Fig. 27, person number 13 obtained the highest value of the Sugeno integral and is correctly selected.

3.5. Program for combining the modules

In order to measure in an objective form the final results, we use 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 three times and the net weights and permutation order are saved for posterior tests, which is implemented by the following algorithm:

1. Build a matrix containing one column for each image after the edge detection.
2. Divide the columns in n parts with 10 rows of overlapping, to obtain n matrices.
3. Repeat steps 4 and 5, 5 times
4. Reorder randomly the samples of each subject.

Table 1
The values sorted descendent by x before normalization.

x_i	$\sigma(x_i$ = trapmf([0 max(i) max(i) 2],x_i)	$\mu(x_i)$
0.1575	0.9989	0.3069
0.0247	0.0492	0.1890
0.0094	0.0249	0.1788

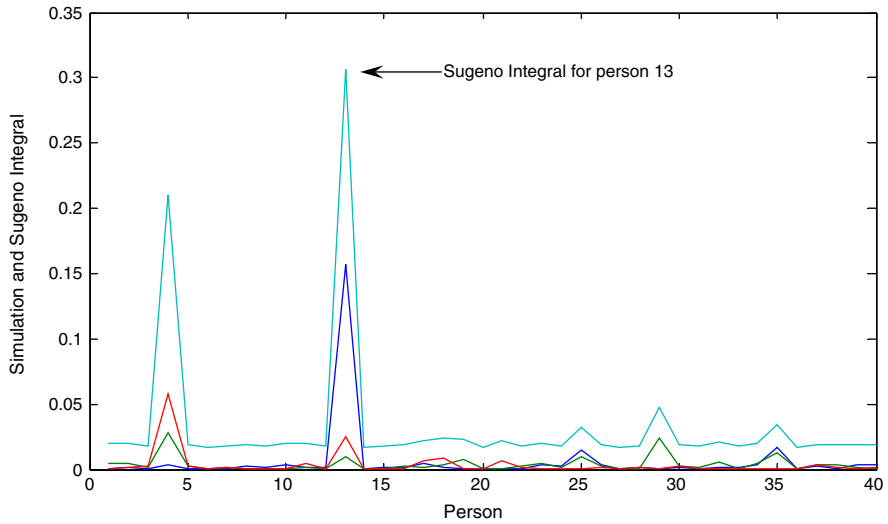


Fig. 27. Simulation of the modules and the Sugeno integral.

5. Repeat steps 6 and 7, 3 times.
6. Train the modular neural networks.
7. Save the networks.

4. Simulation results

4.1. Edge detection

4.1.1. Images generated by the Sobel operators

The first image of subject number one of the ORL database is used as an example (Fig. 28). The gray tone of each pixel of this image has a value between 0 and 255.

In Fig. 29 the image generated by g_x is shown, and in Fig. 30 the image generated by g_y is presented.

An example of the maximum and minimum values of the matrix given by g_x, g_y and g from the image 1.pgm is shown in Table 2. As we can see, the range values are very wide, and this is a training dataset difficult to calculate for a neural network if the supervised method has a target with the value near 1 for good classification and near 0 for bad classification.

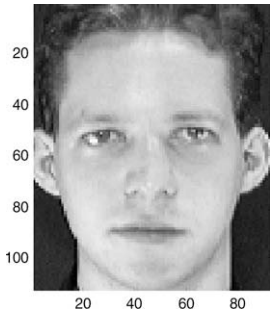


Fig. 28. Original image 1.pgm.

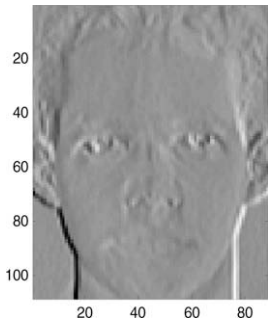


Fig. 29. Image given by g_x.

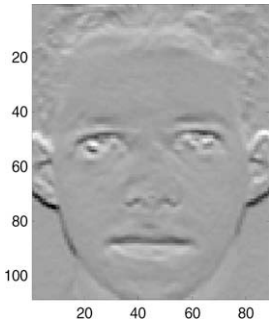


Fig. 30. Image given by g_y.

Table 2
Maximum and minimum values from 1.pgm, g_x, g_y, g.

Tone	1.pgm	g_x	g_y	g
Minimum	11	−725	−778	0
Maximum	234	738	494	792

The useful Edges image is smaller than the original because the result of the convolution operation is a central matrix where the convolution has a value. In our example, each image with dimension 112×92 is reduced to 108×88.

4.1.2. Edge detection by Sobel, FIS1 and FIS2 methods

The inference rules and membership function parameters allow the calculation of a gray value between −4.5 and 1.5 for each pixel, where the most negative values correspond to the dark tones in the edges of the image. Reviewing the values of each pixel, we see that all fall in the rank from −4.5 to 1.5, which is not obtained with the method of gradient magnitude.

Another advantage of the FIS Edge Detector is that values can be normalized to a homogenous range, independently of the light, contrast or background tone in each image. We fit the values for the EDGES variable in order that the network reaches the training goal faster, because the mean of the vector values for each edges image is near 0 and the standard deviation is near 1.

In order to obtain an objective comparison of the images, frequency histograms corresponding to the resulting matrices of edges matrices of the Sobel, FIS1 and FIS2 methods, are shown in Table 3.

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

The results of both FIS1 and FIS2 are better because the output can be normalized to the ideal range for training data. When we compare the images, clearly the results by FIS2 are better than FIS1, because the contrast between background and edges is greater, and the background is more uniform than with the FIS1. For this reason, the rest of tests corresponding to the training and recognition steps were made only with the edges images obtained with the FIS2 detector.

4.2. Recognition rates with the complete method

Once the dataset of images are as shown in Fig. 31, the recognition method is applied, however, some of the images don't reach a sufficient value in the simulation of the three modules, in these cases, there is not enough information to select an image at the modules combination, and the image is wrongly selected. In Tables 4 and 5, we show the recognition rates for FIS1 and FIS2, for each permutation (P), explained in Section 3.5.

5. Comparison with existing pattern recognition methods

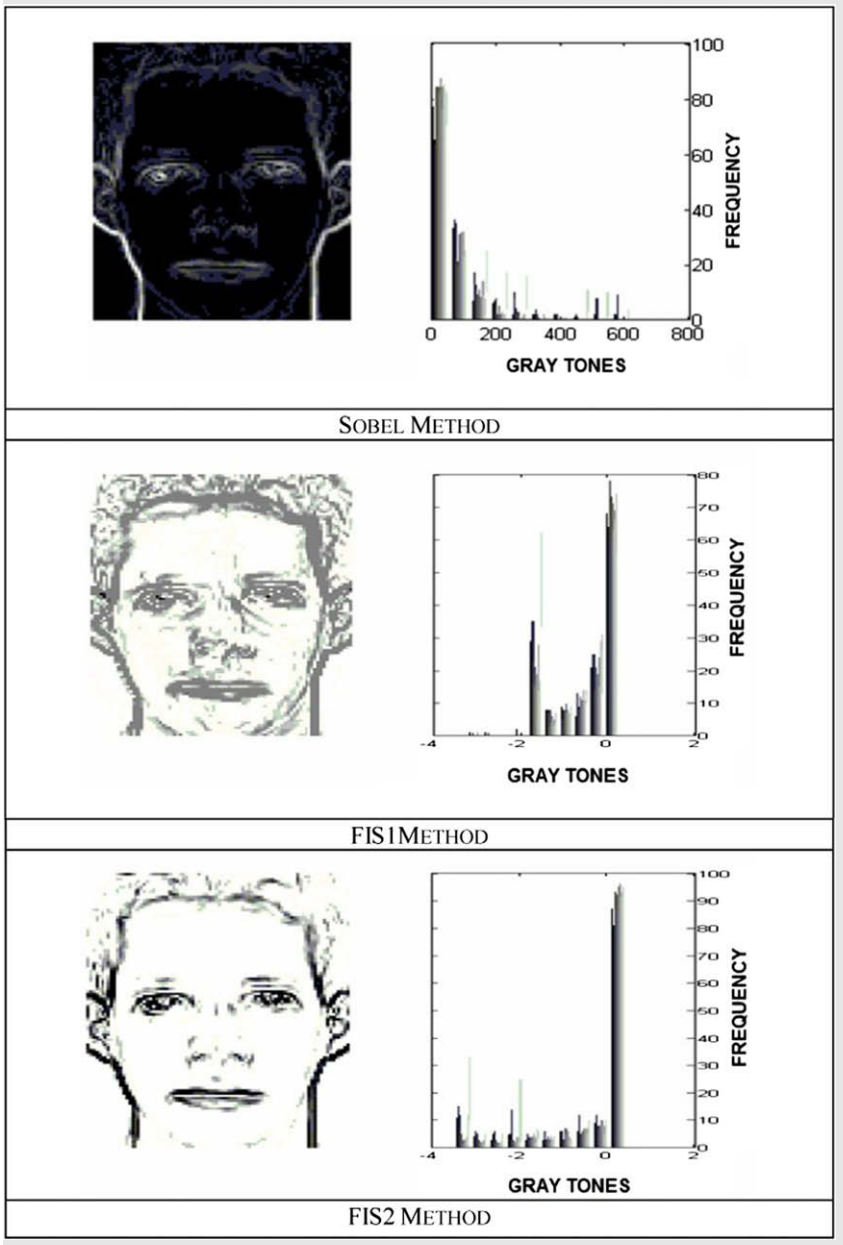
Many methods for image recognition are designed to detect special features of the human face, some of them are based on the gray distribution in the eye and brow region, the nose localization, etc. [1,2]. We propose a method for any type of images, but in this paper we illustrate it with the human face, because it does not need special information about the sample images, then eventually we can adapt this method to classify industrial products, fruits, etc.

The performance of the proposed image recognition method including its two variants: Type-1 Fuzzy Logic plus Modular Neural Networks (T1FL + MNN) and Interval Type-2 Fuzzy Logic plus Modular Neural Networks (IT2FL + MNN), is compared to the performance of other benchmark methods. Our method presents the following relevant advantages:

- The capability to ignore the background gray tones, without losing image detail, then if the test data have a different gray scale, the performance of the recognition is the same.
- The flexibility of a Fuzzy Inference System as Edge Detector to obtain the edges vectors in the best gray scale for the data training set with the purpose of improve the neural network performance.
- The possibility of applying the method to recognize any type of image, because we do not need any extra information, only the original image files.
- The Modular Neural Network structure is trained with segments of images after the edge detector, and each module can be trained with different size of vectors, and allows the overlapping of few rows, that adds flexibility to the image division.
- The Fuzzy Sugeno Integral as a decision operator gives an extra advantage, because is possible to assign different criteria values to calculate the relevance level of each module.

We now present results of some existing Pattern Recognition Methods, all of them need extra information of the image, and/or not consider the possibility of error when the background color, contrast or brightness is modified. All the results correspond to the ORL face database.

Table 3
Histograms of the resulting images of the edges by Sobel, FISI and FIS2 methods.



- Face Recognition with Radial Basis Function (RBF) Neural Networks [5], comprises the following parts:

1. The number of input variables is reduced through feature selection by the principal component analysis (PCA) and the Fisher's linear discriminant (FLD) is then implemented to generate the most discriminant features.
2. A clustering algorithm concerning information of training samples is proposed so that homogeneous data could be clustered and a compact structure of an RBF neural classifier with limited mixed data could be achieved.
3. Two important criteria are proposed to estimate the initial widths of RBF units which control the generalization of RBF neural classifier.
4. A hybrid learning algorithm is presented to train the RBF neural networks so that dimension of the search space is significantly reduced in the gradient paradigm. The results obtained by others and their method are shown in Table 6, and the last two rows correspond to T1FL + MNN and IT2FL + MNN best results.

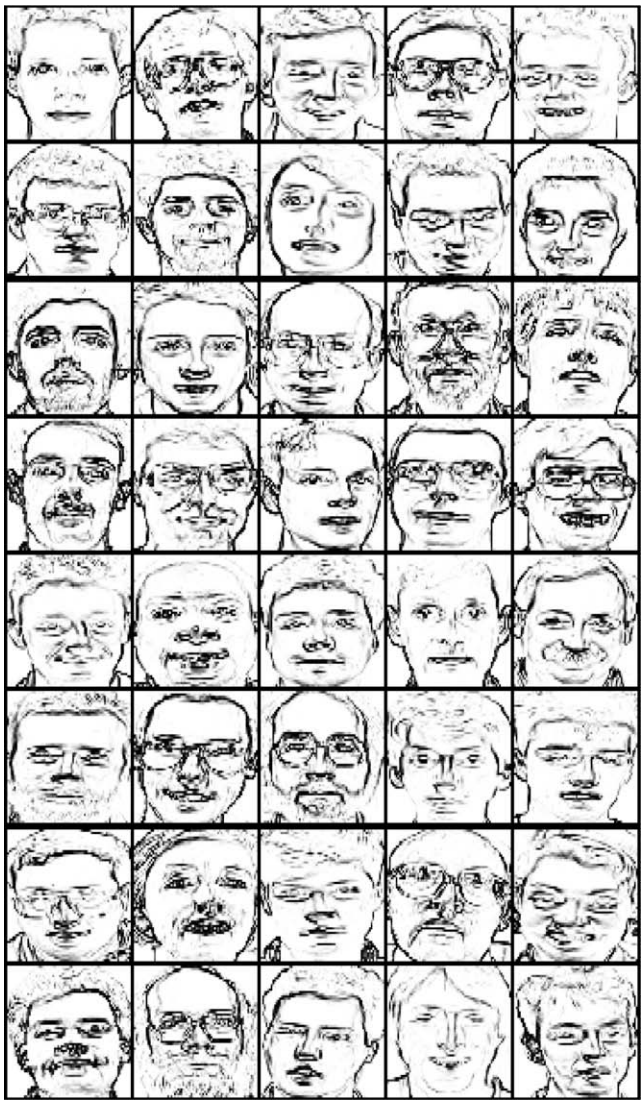


Fig. 31. Set of sample 1 of the 40 subjects in ORL database of faces after FIS2 edge detector.

Table 4
Recognition rates with FIS1 fuzzy densities finder.

P	Image recognition (%)				
	Train 1	Train 2	Train 3	Avg.	Max
1	94.00	95.75	94.50	94.75	95.75
2	94.25	94.75	94.25	94.41	94.75
3	94.25	94.25	95.25	94.58	95.25
4	94.00	93.25	93.50	93.58	94.00
5	94.75	94.75	94.00	94.36	94.75
				94.36	**95.75**

- The technique BBPCA plus LDA (Bidirectional Principal Component Analysis plus Linear Discriminant Analysis) [6] performs feature extraction for face recognition in original images without edge detectors, then the background tones are data included as part of the image pattern. The results obtained by others and their method are shown in Table 7, and the last two rows correspond to T1FL + MNN and IT2FL + MNN best results.

Table 5
Recognition rates with FIS2 fuzzy densities finder.

P	Image recognition (%)				
	Train 1	Train 2	Train 3	Avg.	Max
1	97.25	96.25	95.00	96.17	97.25
2	94.75	95.25	95.75	95.25	95.75
3	95.50	97.50	96.00	96.33	97.50
4	95.25	95.00	95.50	95.25	95.50
5	96.50	97.00	96.00	96.50	97.00
				95.90	**97.50**

Table 6
Error rates reported by the RBF method by other authors and the proposed T1FL + MNN, IT2 FL + MNN methods.

Method	Error (%)	Year
PDBNN	4	1997
Point-matching	16	1998
Pseudo2-D HMM + DCT	0	1999
LVQ + RBF + FEC	0.5	2000
PCA + RBF	4.9	2000
FND	1.1	2001
UDT	2.5	2001
Wavelet + RBF	3.7	2001
PCA + moment invariant	4	2001
RBF	1.92	2002
T1FL + MNN	4.25	2007
IT2FL + MNN	2.5	2007

Table 7
Technique: bidirectional principal component analysis plus linear discriminant analysis (BBPCA + LDA) and the proposed T1FL + MNN, IT2 FL + MNN methods.

Method	Recognition rate (%)	Year
Complete PCA + LDA	97.00	2003
DF-LDA	95.80	2003
NKFDA	95.10	2004
ELDA	95.85	2004
BDPCA LDA	97.10	2006
T1FL + MNN	95.75	2007
IT2FL + MNN	97.50	2007

- The method Uncertainty Estimation Using Fuzzy Measures for Multiclass Classification (FMMC) [7] proposes a modification to the monotonic function model to estimate the uncertainty associated with a new observation for multi-class classification. The model, therefore, overcomes a limitation of traditional classifiers that base decisions on sharp classification boundaries. This approach is based on the transformation of the input pattern vector relative to each classification class. Separate, monotonic, single-output neural networks are then used to represent the degree-of-similarity between each input pattern vector and each class. An algorithm for the implementation of this approach is proposed and tested with publicly available face recognition datasets. The results indicate that the suggested approach provides similar classification performance to conventional principle component analysis (PCA) and linear discriminant analysis (LDA) techniques for multiclass pattern recognition problems as well as providing uncertainty information caused by misclassification. The best results reported by FMMC method authors are shown in Table 8, and the last two rows correspond to T1FL + MNN and IT2FL + MNN best results.

Table 8
Method: uncertainty estimation using Fuzzy Measures for Multiclass Classification (FMMC).

Distance method (1997)	Recognition rate (%)
Manhattan	93.65
Euclidean	97.10
Chevychev	83.95
Cosine	94.40
Correlation	94.70
T1FL + MNN (2007)	95.75
IT2FL + MNN (2007)	97.50

6. Conclusions

The results in this paper show the flexibility of fuzzy logic to model the behavior of different systems, in particular: edge detection and estimation of fuzzy densities. The method does not need extra parameters for achieving success, because all of the parameters are calculated using the fuzzy inference systems, of which the inputs are only the original images and the simulation matrix in each phase.

The method in this paper is based on the combination of Soft Computing techniques that allow the improvement of intelligent systems with different hybrid approaches. The method using a Modular Neural Network for image recognition, where Type-2 Fuzzy Logic help us improve the performance results in image recognition is a good alternative for this kind of problems. The first FIS2 is used for feature extraction in the training data, and the second one to find the ideal parameters for the integration method of the modular neural network. In this method, the hybrid fuzzy neural approach can be considered a good alternative for improving the performance of the modular neural model.

References

[1] P. Yang, B. Du, S. Shan, W. Gao, A novel pupil localization method based on Gabor eye model and radial symmetry operator, ICT-ISVISION Joint R&D Laboratory for Face Recognition, Beijing, China, 2004.
[2] V. Starovoitov, D.I. Samal, D.V. Briliuk, Three Approaches for face recognition, in: The Sixth International Conference on Pattern Recognition and Image Analysis, Velikiy Novgorod, Russia, 2002, pp. 707–711.
[3] J.B. Dowdalla, I. Pavlidisa, G. Bebisb, Face Detection in the Near-IR Spectrum, Department of Computer Science, University of Nevada, Reno, NV, 2003.
[4] Mao-Meng Chuang, Ruey-Feng Chang, Yu-Len Huang, Automatic facial feature extraction in model-based coding, Journal of Information Science and Engineering 6 (2000) 447–458.
[5] M. Joo, S. Wu, J. Lu, H. Lye, Face recognition with radial basis function (RBF) neural networks, IEEE Transactions on Neural Networks 13m (3) (2002) 697–710.
[6] W. Zuo, D. Zhang, J. Yang, K. Wang, BBPCA plus LDA: a novel fast feature extraction technique for face recognition, IEEE Transactions on Systems, Man, and Cybernetics 36 (4) (2006) 946–953.
[7] K. Graves, R. Nagarajah, Uncertainty estimation using fuzzy measures for multiclass classification, IEEE Transactions on Neural Networks 18 (1) (2007) 128–140.
[8] C. Jacques, A. Bauchspiess, Fuzzy inference system applied to edge detection in digital images, in: V Brazilian Conference on Neural Networks, Brazil, 2001.
[9] O. Mendoza, P. Melín, G. Licea, Fuzzy inference systems type-1 and type-2 for digital images edges detection, Engineering Letters, International Association of Engineers, EUA, vol. 15, issue 1, 2007. Available from: <http://www.engineeringletters.com/issues_v15/issue_1/EL_15_1_7.pdf>.
[10] The MathWorks, Inc., 1994-2008, Fuzzy Logic Toolbox, available on the web page: <http://www.mathworks.com/products/fuzzylogic/>.
[11] I.T. Young, J.J. Gerbrands, L.J. Van-Vliet, Fundamentals of Image Processing, PH Publications, Delft, The Netherlands, 1995.
[12] M. Heath, S. Sarkar, T. Sanocki, K.W. Bowyer, A robust visual method for assessing the relative performance of edge-detection algorithms, IEEE Transactions on Pattern Analysis and Machine Intelligence 19 (12) (1997) 1338–1359.
[13] M. Heath, S. Sarkar, T. Sanocki, K.W. Bowyer, A high performance edge detector based on fuzzy inference rules, Information Sciences: An International Journal 177 (21) (2007) 4768–4784.
[14] K. Yau-Hwang, L. Chang-Shing, L. Chao-Chin, A new fuzzy edge detection method for image enhancement, in: Proceedings of the Sixth IEEE International Conference on Fuzzy Systems, Spain, vol. 2, nos. 1–5, 1997, pp. 1069–1074.
[15] S.E. El-Khamy, M. Lotfy, N. El-Yamany, A modified fuzzy Sobel edge detector, in: Radio Science Conference, 17th NRSC'2000. Seventeenth National, Minufiya, Egypt, 2000, pp. C32/1–C32/9.
[16] K. Revathy, S. Lekshmi, R. Prabhakaran, Fractal-based fuzzy technique for detection of active regions from solar, Journal of Solar Physics 228 (2005) 43–53.
[17] G. Rong, W. Li-qun, Y. Shu, C. Yu-hua, An edge detection method by combining fuzzy logic and neural network, Advances in Machine Learning and Cybernetics, Springer, Berlin, 2006. pp. 930–937.
[18] T.Q. Pham, L.J. Van-Vliet, Blocking artifacts removal by a hybrid filter method, in: Proceedings of the 11th Annual Conference of the Advanced School for Computing and Imaging, EUA, 2005, pp. 372–377.
[19] K. Suzuki, I. Horiba, N. Sugie, M. Nanki, Contour extraction of left ventricular cavity from digital subtraction angiograms using a neural edge detector, Systems and Computers Japan, Wiley, Japan, 2003, pp. 55–69.
[20] AT&T Laboratories Cambridge, The ORL database of faces. <http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html>.
[21] P. Melin, O. Mendoza, D. Soto, M. Gutierrez, D. Solano, An intelligent system for pattern recognition and time series prediction using modular neural networks, in: IEEE World Congress on Computational Intelligence, Vancouver, Canada, 2006, pp. 8197–8193.
[22] J. Mendel, Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions, Prentice-Hall, USA, 2001.
[23] J.R. Castro, O. Castillo, P. Melin, L.G. Martínez, S. Escobar, I. Camacho, Building fuzzy inference systems with the interval type-2 fuzzy logic toolbox, in: Analysis and Design of Intelligent Systems using Soft Computing Techniques, Springer, Germany, 2006, pp. 53–62.
[24] J.R. Castro, O. Castillo, P. Melin, An interval type-2 fuzzy logic toolbox for control applications, in: IEEE International Fuzzy Systems Conference, London, UK, 2007, pp. 1–6.
[25] J. Mendel, Advances in type-2 sets and systems, Information Sciences 177 (2007) 84–110.
[26] A. Sharkey, Ensemble and modular multi-net systems, in: Combining Artificial Neural Nets, Springer, Germany, 1999, pp. 1–25.
[27] P. Melin, A. Mancilla, M. Lopez, O. Mendoza, A hybrid modular neural network architecture with fuzzy Sugeno integration for time series forecasting, Journal of Applied Soft Computing 7 (4) (2007) 1217–1226.
[28] O. Mendoza, P. Melin, The fuzzy Sugeno integral as a decision operator in the recognition of images with modular neural networks, in: Hybrid Intelligent Systems: Design and Analysis, Springer-Verlag, Germany, 2007. pp. 299–310.
[29] The MathWorks, Inc., 1994–2008, Neural Network Toolbox, available on the web page: <http://www.mathworks.com/products/neuralnet/>.
[30] P. Melin, O. Castillo, Hybrid Intelligent Systems for Pattern Recognition, Springer-Verlag, Heidelberg, Germany, 2005.
[31] M. Detyniecki, Mathematical Aggregation Operators and Their Application to Video Querying, PHD Artificial Intelligence, Pierre and Marie Curie University, Paris, 2000.
[32] T. Vincent, N. Yasuo, A view of averaging aggregation operators, IEEE Transactions on Fuzzy Systems 15 (6) (2007) 1063–10067.
[33] J.C. Bezdek, J. Keller, N.R. Pal, Fuzzy Models and Algorithms for Pattern Recognition and Image Processing, Springer, USA, 2005.
[34] G. Klir, Uncertainty and Information, Wiley & Sons, Inc., USA, 2005.
[35] V. Torra, Y. Narukawa, Modeling Decisions, Information Fusion and Aggregation Operators, Springer-Verlag, Germany, 2007.
[36] A. Verikas, A. Lipnickas, K. Malmqvist, M. Bacauskiene, A. Gelzinis, Soft combination of neural classifiers, Pattern Recognition Letters (1999) 429–444.
[37] D. Dubois, J. Marichal, H. Prade, M. Roubens, The use of the discrete Sugeno integral in decision-making: A survey, in: A. Colorni, M. Paruccini, B. Roy (Eds.), A6MCD-A: Multiple Criteria Decision Aiding Joint Research Center, The European Commission, Luxembourg, 2001, pp. 81–98.

[38] O. Mendoza, G. Torres, P. Melín, An intelligent system for modeling and simulation with modular neural networks, in: International Conference on Artificial Intelligence, Las Vegas, EUA, 2005, pp. 406–411.
[39] O. Mendoza, P. Melín, 2005, 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, Tijuana, México, pp. 105–116.
[40] O. Mendoza, P. Melin, O. Castillo, G. Licea, Type-2 fuzzy systems as a method for improving training data and decision making in modular neural networks for pattern recognition, International Journal of Information Technology And Intelligent Computing 1(4). Available at: <http://itic.wshe.lodz.pl/index.php>.