Automatic tuning of a fuzzy visual system using evolutionary algorithms: single-objective vs. multiobjective approaches

Rafael Muñoz-Salinas, Eugenio Aguirre, Oscar Cordón, Miguel García-Silvente
Department of Computer Science and Artificial Intelligence
ETS Ingeniería Informática. University of Granada
18071-Granada (Spain). Phone: +34.958.249562. Fax: +34.958.243317
e-mail:{salinas,eaguirre,ocordon,m.garcia-silvente}@decsai.ugr.es

Abstract— **One of the main pros of fuzzy systems is their ability to design comprehensible models of real-world systems, thanks to the use of a fuzzy rule structure easily interpretable by human beings. This is especially useful for the design of fuzzy logic controllers, where the knowledge base can be extracted from expert knowledge. Even more, the availability of a readable structure allows the human expert to customize the fuzzy controller to different environments by manually tuning its components. Nevertheless, this tuning task is usually a time consuming procedure when done manually, especially when several measures are considered to evaluate the controller performance, and thus the interest in the design of automatic tuning procedures for fuzzy systems has increased along the last few years.**

In this paper we tackle the tuning of the fuzzy membership functions of a fuzzy visual system for autonomous robots. This fuzzy visual system is based on a hierarchical structure of three different fuzzy classifiers, whose combined action allows the robot to detect the presence of doors in the images captured by its camera. Although the global knowledge represented in the fuzzy system knowledge base makes it perform properly in the door detection task, its adaptation to the specific conditions of the environment where the robot is operating can significantly improve the classification accuracy. However, the tuning procedure is complex as two different performance indices are involved in the optimization process (true positive and false positive detections), thus becoming a multiobjective problem. Hence, in order to automatically put the fuzzy system tuning into effect, different single and multiobjective evolutionary algorithms are considered to optimize the two criteria, and their behavior in the problem solving is compared.

I. Introduction

Landmark detection is a fundamental task in the autonomous mobile robot navigation using topological approaches [1], [2], [3], [4]. It is used for creating topological maps [5] indicating the structure of an environment, and for localization purposes. Among the landmarks that can be detected, those that give information about the structure of the environment are more relevant. In that sense, a door is a common object that can be found in indoor environments. Doors are important places regarding the structure of man-made environments because they indicate access points between rooms. Therefore, they can been employed not only for localization but also for navigation purposes [6], [7].

A fuzzy visual system was designed in a previous work [8] to be used in autonomous mobile robots for navigating, map-building and positioning purposes. The fuzzy visual system is comprised by a hierarchy of three fuzzy classifiers and is able to detect the doors present in the images captured by our robot. The use of fuzzy logic allows the system to detect doors under strong perspective deformations and at different distances. The variables used, the number and shapes of the membership functions for each variable and the rule base of the three fuzzy systems were determined based on expert knowledge. Furthermore, the system was manually tuned to the specific conditions of our environment, i.e., to the usual conditions of height and size of the doors of the environment and to the distances and orientations under which they are seen by our robot.

Nevertheless, the manual tuning of the visual system is a tedious task and should be repeated in case of translating the system to other working enviroment with different doors' dimensions or camera height. An additional problem of tuning our fuzzy visual system is that its performance is evaluated by a multiobjective function. Although the global goal of the fuzzy visual system can be enunciated as *"to detect the doors present in the images captured by the robot's camera"*, this imply both (a) detect doors in the images where doors are actually present, and (b) not to indicate the false presence of a door in an image without any door. The former case is evaluated using the *True Positive Fraction* (TPF) and the latter by the *True Negative Fraction* (TNF). The two objectives are independent and in conflict, but both must be maximized in the tuning process. For these reasons, we consider interesting to develop an automatic mechanism for tuning the fuzzy visual system employing training images captured at the particular environment in which it is going to be used.

In this work we propose a methodology to automatize the tuning of the fuzzy visual system using an evolutionary approach. We decided to use evolutionary algorithms (EAs) [9] for tuning the system instead of classical tuning approaches [10] because it is possible to consider the whole fuzzy visual system as a unique system to optimize. Therefore, it is only necessary to define one error function to evaluate the performance of the whole system, instead of a different error function for each one of the three composing fuzzy systems. Besides, the multiobjective nature of the problem can be managed in a proper way by means of multiobjective EAs

(MOEAs) [11], [12].

Within the optimization field using EAs, we find the possibility to choose among single-objective algorithms, involving the aggregation of the different objectives into an scalar single-objective function, and multiobjective algorithms, that considers the joint optimization of all of them [11], [12]. In this work we have tested both approaches. On the one hand, we have used two single-objective EAs: a generational Genetic Algorithm [13] and the CHC algorithm [14]. On the other hand, we have tested three MOEAs: SPEA [15], SPEA2 [16] and NSGA-II [17]. All of them are run under similar conditions and their results compared at the end of this article.

The remainder of the paper is structured as follows: Section II is devoted to report a brief state of the art on genetic tuning of fuzzy systems and door detection. Section III reviews our fuzzy visual system designed for detecting doors. Sections IV and V describe the basis of the single-objective and multiobjective EAs used, respectively. Section VI shows the coding scheme, genetic operators and fitness functions employed for our tuning approach. Section VII shows the experiments carried out and Section VIII exposes some conclusions. Finally, we have added an Appendix that explains the basis of the multiobjective topic and the metrics used to compare the algorithms employed in this work.

II. Preliminaries

A. Genetic Tuning of Fuzzy Systems

Tuning of fuzzy systems provides an automatic way to increase their performance using a validated data set describing the problem. Three main tuning approaches can be found in the literature [18]: tuning of the scaling functions that maps the input and output variables into the ranges in which the fuzzy variables are defined; tuning of the membership functions by moving, stretching or narrowing them; and tuning of the fuzzy rules modifying the fuzzy labels in the THEN-part of the rules. Although the tuning process has been performed using different optimization techniques [10], the use of genetic algorithms (GAs) has become very popular in this field [19], [20], [21], [22], [23]. The term *Genetic Fuzzy System* has been used in the literature to refer to those fuzzy systems that somehow rely on GAs either to define their structure or to adapt some of their parameters [18], [24].

Although a great effort has been done on the optimization of fuzzy systems using single-objective approaches, there is usually a need in real problems for fulfilling several objectives at the same time. Those problems are referred to as *multiobjective* in the literature (an interesting survey on the topic can be found in [25]). The concept of *Pareto-optimum*, formulated by Vilfredo Pareto [26], constitutes the origin of the research in *Pareto-based* multiobjective optimization. In a typical multiobjetive framework, there is a set of solutions that are superior to the remainder when all the objectives are considered, the *Pareto* set. These solutions are known as *non-dominated solutions*. Since none of the Pareto set solutions is absolutely better than the other non-dominated solutions, all of them are equally acceptable as regards the satisfaction of all the objetives. Appendix A gives a deeper understading of the concept providing a mathematical descripion of it.

The term EMO (Evolutionary Multiobjective Optimization) refers to a set of evolutionary techniques that have been used to solve multiobjective optimization problems. The main advantage of EMO algorithms, so called *MOEAs*, is the ability of simultaneously searching for several non-dominated solutions. MOEAs find different non-dominated solutions to a problem in a single run while several runs of a normal algorithm would be required to obtain similar results.

MOEAs have also been used for Genetic Fuzzy Systems. Much of the work has been focused on the learning of the database or the rule base of the fuzzy system [27], [28], [29], [30]. Nevertheless, there are also approaches to tune the membership functions of the fuzzy system. In [31], a fuzzy system that controls a missile is tuned using a MOEA. It allowed to simultaneously optimize the rising and settling time, the steady state error and the overshot of the controller. In [32], a fuzzy system to control a MAS (Mass Rapid Transit) is tuned using a multiobjective approach based on Dynamic Evolution. The system was used to control the operation of a train dedicated to the transport of people from one station to another. It was composed of 16 parameters and the results showed that the method was able to optimize the system according to the objectives of punctuality, energy and passenger comfort.

B. State of the art on door-detection

Door detection has been done in the literature using different kind of sensors like laser [33] and sonar [34], [35], [36]. The door-detection problem using computer vision has also been previously tackled using different approaches.

In the work developed in [37], a pair of neural networks is used to detect doors in color images. One net is employed to detect the lateral and vertical bars of the door and the other to detect the corners. The nets receive as input subwindows of the image of size 18×18 centered in each pixel with the hue and saturation components. After the classification process, an analysis of the components found is performed to detect if there is any door present in the image. The system has the disadvantage of requiring a high computational effort in processing each image. Furthermore, it can not detect fully opened doors and it is dependent on the color of the doors used for training.

Based on a functionality-based approach, a method for generic object recognition used for robot navigation is presented in [38]. A door is defined as an inverted U that can be crossed by people. A trinocular vision system is used in order to detect segments in the images of the environment. The segments are analyzed to check if they accomplish a set of size and height restrictions typical of its indoor environment doors. The trinocular vision system makes possible to know the real position of the segments in the space and thus check the imposed restrictions. The system has the disadvantage of the cost of the perceptual system.

Other related works use neural networks in order to classify the segments of the doors [39], add information provided by other sensors [40] or focus on the door-detection problem in corridors [41].

III. Our approach for door-detection

In [8] we presented a new method for visual door-detection based on the extraction of the segments from images and the definition of several fuzzy concepts.

Our system is able to detect typical doors in grey-level images and can be used for real-time applications. Three fuzzy systems are employed to analyze the segments extracted from an image looking for doorframes in different situations. The system created is able to detect doors under the strong perspective deformations caused by the two degrees of freedom (DOFs) allowed for the camera of our robot (a Nomad200). The proposed method is valid for different image sizes since all the parameters have been set with independence of the size of the image.

The use of fuzzy logic brings several advantages when dealing with the problem. It allows us to define concepts in a flexible way. Properties like *vertical* or *horizontal* are defined as linguistic variables allowing to manage perspective deformations and vagueness in the segment extraction in a natural way. Another advantage of using fuzzy logic is the facility for combining the information provided by the visual system with other previously developed perceptual model based on ultrasound [5], [42].

Our approach is based on the detection of the image segments that form part of the doorframe. In the first step of the processing, segments in the image are detected using computer vision techniques. Then, the first fuzzy system analyzes them and selects those that are more likely to belong to doorframes. When a doorframe is captured in an image, it can form two frame edges with its surrounding. Firstly, there is the frame edge formed by the doorframe and the wall. Secondly, there is the frame edge formed by the doorframe and either its leaf (if their colors are different) or the gap of the door (if it is open). Therefore, based on our experience we have identified two possible cases in which a doorframe can be detected. In the first case, only one of the frame edges is present. This case has been formally defined by the *Frame Edge* (FE) fuzzy concept. In the second case, both frame edges are detected. This case has been formally defined by the *Complete DoorFrame* (CDF) fuzzy concept. We have designed two fuzzy systems (one for each case) able to detect these segment configurations.

Both cases can be better understood by looking at Figure 1. Although Figure 1 shows the ideal case in which the segments of the doorframe are completely vertical and horizontal, this only happens when the camera is completely in front of the door and aligned with it. Nevertheless, due to the two DOFs that provide the pan-tilt unit over which is placed our camera, the segments will normally appear in different directions and with different sizes. Our aim is to detect doors despite these perspective deformations and scale changes.

The detection process starts applying the Canny edge detector [43] on a grey-level image. The result is a binary image $I(x, y)$ where pixels labeled as *true* belong to edges in the original image. In order to ease the notation we assume that the dimension of the image is $N_{img} \times N_{img}$. The edge pixels detected are used to compute the Hough Transform [44] and then the segments from the image are extracted [45]. Let us

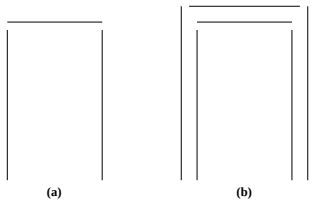


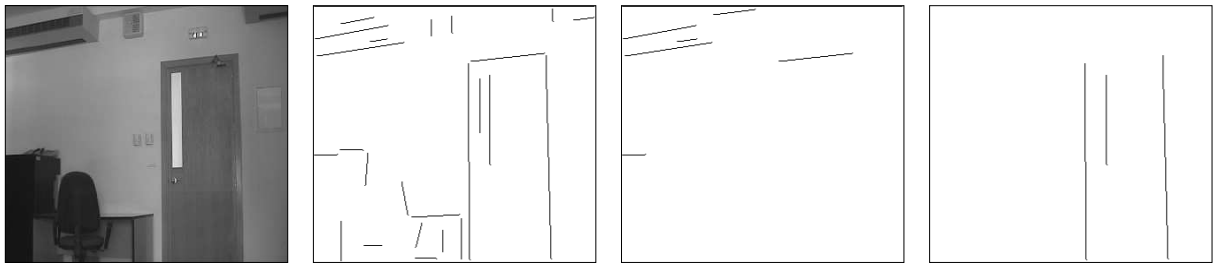
Fig. 1. (a) FE and (b) CDF Fuzzy concepts

denote the set of extracted segments by $S = \{S^0, S^1, ..., S^m\}$. Each segment is composed of two points, $S^i = \{p_0^i, p_1^i\}$, where $p_j^i = (x_j^i, y_j^i)$ (coordinates in the image plane). When the segments are extracted, the analysis to detect possible doors is performed in three phases. In the first phase, a fuzzy system classifies the segments and rejects the less promising according to their direction, size and position in order to reduce the computational effort required for the posterior phases. In the second phase, a fuzzy system searchs for the FE fuzzy concept among the segments selected in the previous phase. Finally, in the third phase, a fuzzy system analizes the FEs found in the previous phase looking for CDFs. The three phases of the process are explained in detail below. Then, the need of an automatic tuning mechanism for the fuzzy systems is justified.

A. Segment classification

The total number of extracted segments could be high and it is desirable to reduce it and select only those that could belong to a doorframe. Therefore, an initial classification is performed. Two classes of segments are of interest, the vertical segments that belong to the lateral bars of a doorframe, and the horizontal segments that belong to the upper part of the doorframe. Nevertheless, it is important to remember that we wish to detect the doors under the perspective deformations that make the segments appear neither completely vertical nor horizontal. Therefore, we have defined two fuzzy concepts to represent both cases and manage these deformations. The *Vertical Segment* (VS) fuzzy concept refers to those segments that belong to the lateral bars of the doorframe, while the *Horizontal Segment* (HS) fuzzy concept regards to those segments that belong to the upper part of the doorframe. The two rule bases shown in Table I, extracted from expert knowledge, are used to calculate the membership degree of a segment S^i to the VS and HS concepts in the $[0, 1]$ range. Both values are calculated by a fuzzy inference process and its corresponding defuzzification. We shall denote the membership degree of a segment S^i to the VS and HS fuzzy concepts by $VS(S^i)$ and $HS(S^i)$, respectively. The fuzzy sets related to the VS concept are three (*low, medium* and *high*) and are identical to the fuzzy sets related to the HS concept. The HS and VS fuzzy concepts are defined based on three fuzzy variables (*Direction*, *Size* and *YPosition*) that evaluate three different features of a segment.

The fuzzy variable Direction refers to the direction in the image plane of a segment $S^i = \{p_0^i, p_1^i\}$; $p_j^i = (x_j^i, y_j^i)$. It has three possible values (*horizontal, medium* and *vertical*) and its input value is given by Equation 1. The fuzzy variable Size refers to the size of a segment in the image. It has three



(a) Original image **(b) All extracted segments** **(c) Horizontal segments** **(d) Vertical segments**

Fig. 2. Segment classification process

IF			THEN	
Direction	Size	YPosition	VS(S^i)	HS(S^i)
horizontal	small	high	low	high
horizontal	medium	high	low	high
horizontal	big	high	low	high
medium	small	high	low	medium
medium	medium	high	low	high
medium	big	high	low	high
vertical	small		medium	low
vertical	medium		high	low
vertical	big		high	low

TABLE I

RULE BASES FOR CLASSIFICATION OF SEGMENTS IN VERTICAL OR HORIZONTAL

possible values (*small, medium* and *big*) and its input value is given by Equation 2. Finally, the fuzzy variable YPosition refers to the position of a segment in the *y*-plane of the image. It has three possible values (*low, medium* and *high*) and its input value is given by Equation 3. We should remind that N_{img} stands for the image size in the latter equations. For more details about the meaning of these variables and equations, the interest reader is referred to [8].

$$Direction(S^i) = \frac{2}{\pi}\arctan\left(\frac{|y_1^a - y_0^a|}{|x_1^a - x_0^a|}\right). \quad (1)$$

$$Size(S^i) = \frac{dist(p_0^i, p_1^i)}{N_{img}\sqrt{2}}. \quad (2)$$

$$YPosition(S^i) = \frac{y_0^i + y_1^i}{2N_{img}}. \quad (3)$$

Each one of the extracted segments is analyzed and its corresponding membership degrees $VS(S^i)$ and $HS(S^i)$ to the fuzzy concepts VS and HS are calculated. The aim of this process is dual, on the one hand, to classify the segments into horizontal and vertical segments, and on the other hand, to eliminate those segments that, according to their features, do not seem to belong to a doorframe. Therefore, the values $VS(S^i)$ and $HS(S^i)$ are computed and only those segments whose membership degree to one of the two concepts exceed a certain threshold t_1 are used in the following phases. Those whose membership degrees to the fuzzy concepts VS and HS are below t_1 are considered to have a low possibility of belonging to a doorframe and are thus removed to speed up the further processing. If S^i is selected for the next phases then it is classified as a vertical segment if $VS(S^i) > HS(S^i)$ or as a horizontal segment otherwise. Let us denote the set of vertical segments selected as $V = \{V^0, ..., V^n / VS(S^i) > t_1 \wedge VS(S^i) > HS(S^i)\}$ and the horizontal one as $H = \{H^0, ..., H^n / HS(S^i) > t_1 \wedge VS(S^i) \leq HS(S^i)\}$.

The appropriate selection of t_1 is not a trivial task. A low value causes an acceptance of all the segments detected, thus not increasing the speed of the further processing. On the other hand, the use of very high values could reject segments actually belonging to frame edges of a doorframe.

Figure 2 shows an example of the classification process with an image of our environment. Figure 2(a) shows an image captured with the camera of our robot. In Figure 2(b) all the segments extracted in the image are depicted. Figures 2(c) and 2(d) show the segments classified as horizontal and vertical using the previous method. Once the interesting segments have been selected and classified, they are analyzed in order to find if their relationships show that they compose a doorframe.

B. Frame Edge Detection

The next step is to analyze if there is any frame edge (FE) in the set of extracted segments. The FE fuzzy concept expresses the idea that there is a horizontal segment (HS) joined to a pair of vertical segments (VS) in its upper part. The HS and VS fuzzy concepts have been previously calculated. Nevertheless, it is already necessary to analyze if there is a trio of segments (two vertical and one horizontal) whose distances reveal that they belong to a frame edge. The *Frame Edge Cohesion* fuzzy concept (FEC) is defined for that purpose.

The detection process starts selecting for each horizontal segment $H^i \in H$, a pair of vertical segments $\{V^j, V^k\} \in V$ and analyzing them. Let us denote the trio by $F^i = \{L^i, Sup^i, R^i\}$, being $L^i \in V$ the leftmost vertical segment, $Sup^i \in H$ the horizontal segment, and $R^i \in V$ the rightmost vertical segment of the selected trio.

The membership degree $FEC(F^i) \in [0, 1]$ of F^i to the fuzzy concept FEC is calculated using the rule base of Table II (also extracted from expert knowledge) by a fuzzy inference process and its corresponding defuzzification. The $SegDistVV(F^i)$ and $SegDistVH(F^i)$ fuzzy variables are used to measure the distance between the segments.

FEC expresses if a trio of segments are close enough to be part of a frame edge. Altough $FEC(F^i)$ indicates that the degree of separation between the segments of F^i is appropriate

(a) FE **(b) CDF** **(c) CDF**

Fig. 3. Examples of the FE and CDF fuzzy concepts detected in images of our environment

to belong to a frame edge, it does not take into account the corresponding membership degree of each individual segment to the VS and HS concepts. Therefore, the membership degree $FE(F^i) \in [0,1]$ of F^i to the FE fuzzy concept is calculated (as expressed in Equation 4) to take into account how well the individual segments accomplish the corresponding HS and VS fuzzy concepts.

$SegDistVV(F^i)$	$SegDistVH(F^i)$				
	VL	L	M	H	VH
VL	L	L	L	L	L
L	H	M	M	L	L
M	H	H	M	L	L
H	H	M	L	L	L
VH	M	L	L	L	L

TABLE II

RULE BASE FOR LINGUISTIC VARIABLE $FEC(F^i)$

$$FE(F^i) = min\{FEC(F^i), VS(L^i), HS(Sup^i), VS(R^i)\} \quad (4)$$

Only those trios F^i whose membership degree $FE(F^i)$ exceeds a threshold t_2 are used in the following phase. Let us denote this set as $F = \{F^0, ..., F^n / FE(F^i) > t_2\}$. Image 3(a) shows an example of an FE detected in an image of our environment.

C. Double Frame Detection

In some cases, it is possible to see the two frame edges of a door. For example, when the door is seen open from the side that does not contain its leaf, or when the door is closed but the color of its leaf is different from the color of the frame. Therefore, the set of frame edges F selected in the previous phase is analyzed in order to detect if two of them belong to the same door. The *Complete Door Frame* (CDF) fuzzy concept evaluates if two frame edges belong to the same door (see Figure 1(b)). The previous phase allows the detection of frame edges by the FE concept. A new fuzzy concept, *Frame Edges Similarity* (FES), evaluates the degree in which two frame edges F^i and F^j are parallel and close. The definition of this concept is based on the philosophy employed to create the variables and rule sets of the previously described concepts and the interest reader in referred to [8] for further information. Then, using both concepts (FE and FES), the membership degree of the two frame edges to the CDF fuzzy concept is calculated using fuzzy logic operations. A threshold t_3 is used, as in the previous cases, to select those pairs whose membership degree to the CDF concept is high enough to be considered for a posterior analysis. Figures 3(b) and 3(c) shows examples of CDFs detected in images of our environment.

D. Neccesity of an automatic tuning mechanism

The number of variables, the number of linguistic labels for each of them and the rule bases have been created based on expert knowledge. Furthermore, the appropriate ranges for the linguistic labels and the values for the parameters t_1, t_2 and t_3 have been manually selected based on the experimentation developed to achieve a good performance of the fuzzy visual system in our working environment. Nevertheless, if we wish to use the same fuzzy visual system to detect doors of different sizes or with cameras placed at different heights, it is neccesary to tune the system again to these particular conditions.

Manual tuning of the fuzzy systems is a tedious task, aggravated in our case by the fact that the performance of our system depends on two different aspects. On the one hand, it is required that the system detects the doors present in images with doors. However, it is also important that the system does not indicate the false presence of a door when the robot is looking at scenes without doors. The first objective is measured by the True Positive Fraction (TPF) and the second by the True Negative Fraction (TNF) (the functions employed to measure these values are later explained in Section VI-C). These reasons have lead us to consider the development of an automatic mechanism to perform the tuning. The aim is to be able to automatically tune the fuzzy visual system by just providing a set of images captured at the desired environment both with and without doors manually labeled.

Automatic tuning of fuzzy systems has been treated using both classical optimization approaches (like gradient descent)

Generational Genetic Algorithm(N,p_{select},$p_{crossover}$):

1) $N_{select} = N * p_{select}$
2) $N_{crossover} = N * p_{crossover}$
3) $N_{mutate} = N * (1 - p_{select} - p_{crossover})$
4) $P_{t=0}$=Create Initial Population of N individuals
5) While not achieved the termination condition
   - 5.1 Select N_{select} individuals using binary tournament and copy them to P'
   - 5.2 Select $N_{crossover}$ individuals using binary tournament and cross over them. Add the $N_{crossover}$ offsprings to P'
   - 5.3 Select N_{mutate} individuals and mutate them. Add the mutated individuals to P'
   - 5.4 Assign $P_{t+1} = P'$ and empty P'
   - 5.5 t=t+1

Fig. 4. Generational GA

and evolutionary approaches. In our particular case, the latter offers the advantage of allowing us to consider the whole fuzzy visual system as a unique system to optimize. Therefore, it is possible to measure its performance by just analyzing its final results over a data set, instead of analyzing the individual performance of each one of its three composing fuzzy systems (that is the case if we had used classical approaches).

Since the tuning of our fuzzy visual system is a multiobjective problem, we can use both single-objective approaches (combining the different functions into a single escalar function) or multiobjective approaches. In this work we test both in order to select the most appropriate one. In the two following sections, the basis of the employed EAs are explained.

IV. Single-Objective approaches for tuning the fuzzy visual system

The two single-objective EAs considered are described in the next two subsections.

A. Generational GA

EAs have proved to be a useful tool for solving a broad class of difficult optimization problems [46]. Among EAs, GAs are a particular instance based on the natural evolution concept. In that area, the generational GA (GGA) [13] is the classic exponent of this type. A GGA starts with random population that is evolved each iteration using selection, crossover and mutation operators, and the offspring population directly substitutes the parent one for the next generation. Each individual is ranked with a fitness accordingly to its goodness in the problem solving.

In our implementation, the selection operator chooses a set of individuals to be part of the following generation without any modification. This operator introduces the selective pressure concept, making the good individuals to more probably be in the next generation. In our case, we have used a binary tournament that involves randomly choosing two individuals of the current population and selecting the one with the best fitness for the next generation population.

The crossover operator exploits the idea that from the combination of good individuals (parents) it is possible to generate good new individuals (offsprings). The parents are selected using binary tournament and the resulting offsprings are used for the next population.

Finally, the mutation operator works on an individual by randomly altering it. The use of this operator allows to explore undiscovered parts of the search space and prevents the algorithm to get stuck in local minima.

Figure 4 outlines the GGA operation. In order to run the algorithm, it is neccesary to specify the number of individuals of the population N and the percentage of those individuals that are to be selected, crossed over and mutated. At each iteration, a new population P' is created using the aforementioned operators over the individuals of the current population P_t. The current population is entirely replaced by the new one. Notice that the proposed GA does not consider elitism. We implemented the two versions, elitist and non elitits, and the latter performed better for the current problem. This can be due to the higher diversity induced by the non elitist version.

B. CHC

The CHC algorithm [14] is an evolutionary approach that introduces an appropriate trade-off between diversity and convergence. For that purpose, it uses a high selective pressure based on an elitist scheme in combination with a highly disruptive crossover and a re-start when the population is stagnated. It is based on four distinguishing components [14]:

- HUX crossover operator. The original algorithm was designed to be used with binary coding and this crossover operator ensures the obtaining of the most diverse offsprings from their parents.
- Incest prevention. Two parents are not crossed over if they are too similar. This ensures diversity.
- Elitist selection. A temporary population is obtained by joining the parents and offspring generated, and the best N individuals of it compose the new population.
- Re-start. When the population reaches an stagnated state, it is re-started keeping the best individual.

The algorithm pseudo-code is shown in Figure 5. In the first step, the population is created using a perturbation operator over the initial chromosome. In our case, the initial chromosome is created from the original definition of the fuzzy visual system previously explained (based on expert knowledge). Then, the algorithm measures the mean distance of the population (D_{mean}) in order to estimate when two individuals are too close to be crossed and in this way avoiding a possible incest. Incest prevention forces to cross over separated

CHC:

1) P_0=Create initial population
2) D_{mean}=Calculate mean distance of the population
3) D_{max}=Calculate maximum distance between individuals
4) $Decrement = DecFactor * D_{max}$
5) While not achieved the termination condition
   - 5.1 Pair up randomly the individuals of the population to be used as parents of the new offspring population
   - 5.2 Create a new offspring population $Offspring_i$ with M individuals ($M \leq N$) using the crossover operator. If the distance between the two parents is smaller than D_{mean} do not generate the offspring
   - 5.3 Replace the individuals of P_i with the individuals of $Offspring_i$ that are better than them
   - 5.4 If no new offspring are generated $D_{mean} = D_{mean} - Decrement$
   - 5.5 If $D_{mean} < 0$ re-start keeping the best individual and recalculate D_{mean} and $Decrement$

Fig. 5. CHC evolutionary algorithm

elements causing an exploration of the area covered by the individuals of the population. D_{mean} is decremented each time the incest prevention mechanism makes impossible to generate any new offspring in one iteration. *DecFactor* allows us to select the degree of convergence of the algorithm, the smaller it is, the greater the level of convergence allowed and *vice versa*. When D_{mean} is below zero, the population is re-started keeping the best individual found. The new population is generated using the perturbation operator on the best individual with probability 35%. It means that only the 35% of the chromosome is altered.

V. MultiObjective approaches for tuning the fuzzy visual system

The three MOEAs used are reviewed in the next three subsections.

A. SPEA

The SPEA algorithm [15] is a Pareto-based MOEA that employs an elitist scheme. In order to maintain the elitism, the algorithm keeps an external population P^e with the non-dominated solutions found since the start of the run. The outline of the algorithm can be seen in Figure 6.

The algorithm starts creating a random population P of N elements. The non-dominated solutions of P are stored in the external elitist population P^e that is updated each generation.

A fitness value is assigned to all the elements of P and P^e. The fitness of the elements from P is calculated using a different criteria than the elements of P^e. An individual $\vec{x}^i$ from P^e is evaluated using Equation 5. The value n_i is the number of solutions in P dominated by $\vec{x}$. Equation 5 assigns low values to those non-dominated solutions that do not dominate other solutions in order to enforce the search in this area that is not appropriately covered by the population. On the other hand, an individual $\vec{x}^j$ from P is evaluated using Equation 6. This equation gives low values to those individuals that are weakly dominated.

$$s(\vec{x}^i) = \frac{n_i}{N+1} \tag{5}$$

$$s(\vec{x}^j) = 1 + \sum_{\vec{x}^i \in P^e \ and \ \vec{x}^i \ dominates \ \vec{x}^j} s(\vec{x}^i) \tag{6}$$

Once a fitness value is calculated for all the individuals, those that are going to be mutated and crossed over are selected using binary tournament among all the elements of $P \cup P^e$. After applying the variation operators, the non-dominated solutions existing in the new population are copied to P^e, removing the dominated and duplicated ones. Therefore, the new elitist population is comprised by the best non-dominated solutions found so far, including new and old elitist solutions.

In order to limit the growth of the elitist population, a clustering algorithm is employed if the number of non-dominated solutions found is greater that a threshold value N^e. The clustering algorithm (showed in Figure 7) selects the solutions closer to the center of each cluster.

B. SPEA2

SPEA2 [16] is a MOEA that uses a Pareto-based elitist approach but it is modified to eliminate the potential weaknesses of SPEA. SPEA2 mainly differs from SPEA in three points:

- The fitness assignment scheme is modified to take into account both the number of individuals that an individual dominates and is dominated by. The problem appears in SPEA when P^e has only one non-dominated solution. In that case, all the dominated solutions have the same fitness and the algorithm behaves as a random search algorithm.
- In order to avoid the grouping of the individuals of the population (P and P^e), a density estimation value is introduced in the fitness function. This value assigns a better fitness to those solutions that are far from other solutions, thus enforcing an extended Pareto front.
- The management of the external population is modified. In SPEA2, P^e has fixed size and if the number of non-dominated solutions is not enough to fill P^e then the best dominated solutions from P are copied to P^e. On the other hand, if the number of non-dominated solutions exceeds the limit, a truncation operator is employed instead of the clustering technique of SPEA.

The outline of the algorithm is shown in Figure 8. It starts creating the initial random population P_0. All the elements are ranked according to the fitness assignment scheme, that in

SPEA:

1) P=Initial Population
2) P^e=Non-dominated solutions of P
3) While not achieved the termination condition
   - 3.1 Assign a fitness value to the elements of P and P^e
   - 3.2 Select N individuals from $(P \cup P^e)$ by binary tournament
   - 3.3 Apply crossover and mutation
   - 3.4 Copy the non-dominated solutions in P^e and remove the dominated solutions
   - 3.5 If $|P^e| > N^e$ then reduce P^e using clustering

Fig. 6. SPEA procedure

SPEA Clustering:

1) Assign each element to a cluster
2) While $|P^e| > N^e$ do
   - 2.1 For each pair of clusters
     - i) Calculate the distance between clusters, like the average distance among all their elements

$$D_{ij} = \frac{1}{|c_1| \cdot |c_2|} \cdot \sum_{i_1 \in c_1, i_2 \in c_2} \|i_1 - i_2\|$$

   - 2.2 Merge the two clusters with minimum distance between them
3) Choose a solution of each cluster, that with minimum average distance to the remaining cluster elements

Fig. 7. Clustering in SPEA

SPEA2:

1) Set t=0 and P_t=Initial Population.
2) Assign a fitness value to the elements of P_t
3) Copy all the non-dominated solutions of P_t in P_t^e. If $|P_t^e| > N^e$ then reduce P_t^e by means of the truncation operator. If $|P_t^e| < N^e$ then fill P_t^e with the best dominated individuals of P_t
4) If the termination condition is reached stop
5) Create a mating pool P^m by binary tournament selection with replacement on P^e
6) Use crossover and mutation operators to create the new population P_{t+1}.
7) Set $t = t + 1$ and go to step 2

Fig. 8. SPEA2 procedure

contrast to SPEA, is the same for all the elements in P and P^e.

The fitness value of an element $\vec{x}^i$ is based on the sum of two values, the raw fitness $R(\vec{x}^i)$ and a density estimation $D(\vec{x}^i)$. The raw fitness (calculated using Equation 7) is determined based on the strength of its dominators $S(\vec{x}^i)$. The function $S(\vec{x}^i)$ is a strength value that indicates the number of individuals that $\vec{x}^i$ dominates as showed in Equation 8.

$$R(\vec{x}^i) = \sum_{\vec{x}^j \in \{P_t \cup P_t^e\} \wedge \ \vec{x}^j \ dominates \ \vec{x}^i} S(\vec{x}^j) \tag{7}$$

$$S(\vec{x}^i) = |\{\vec{x}^j | \vec{x}^j \in \{P_t \cup P_t^e\} \wedge \ \vec{x}^i \ dominates \ \vec{x}^j\}| \tag{8}$$

When most of the solutions are non-dominated, the raw fitness might be the same for all of them and this value might be useless. Therefore, a density estimator measure $D(\vec{x}^i)$ is calculated using Equation 9. The value σ_i^k is the distance of $\vec{x}^i$ to the k-th nearest neighbor in the objective space. $D(\vec{x}^i)$ assigns a better fitness value to those individuals that are isolated, thus enforcing an extended Pareto front.

$$D(\vec{x}^i) = \frac{1}{\sigma_i^k + 2} \tag{9}$$

A truncation operator is used to reduce the number of non-dominated solutions in P^e keeping the extent of the Pareto front. The individual with the minimum distance to other individual is the first selected for removal; if several individuals are in the same situation, the second smallest distance is considered and so forth.

C. NSGA-II

The NSGA-II MOEA [17] appears to solve the weaknesses of its predecesor NSGA [47]. NSGA-II is a Pareto-based evolutionay algorithm that employs an elitist approach (one of the drawbacks of NSGA is that it does not use elitism). In contrast to SPEA and SPEA2, NSGA-II does not mantain an external population of non-dominated solutions but the new population is comprised by the best individuals of the parent and offspring populations (like CHC).

The ranking scheme employed consists in sorting the individuals of a population in different Pareto fronts (or levels) $\mathcal{F}_i$. The first level ($\mathcal{F}_1$) is comprised by the non-dominated individuals of the population. Once the individuals of the first level have been found, they are temporarily removed. The second level ($\mathcal{F}_2$) is comprised by the remaining nondominated individuals. This process is repeated until no individual

NSGA-II:

1) P_0=Initial Population.
2) Fast-nondominated-sorting(P_0)
3) $t = 0$
4) While the termination condition is not reached
   - 4.1 Use crossover and mutation operators to create the new population Q_t
   - 4.2 $R_t = P_t \bigcup Q_t$
   - 4.3 $\mathcal{F}$=fast-nondominated-sorting(R_t)
   - 4.4 $P_{t+1} = 0$ and $i = 1$
   - 4.5 While $|P_{t+1}| \neq N$
     - i) If $|P_{t+1}| + |\mathcal{F}_i| < N$ add all the solutions from $\mathcal{F}_i$ into P_{t+1}
     - ii) Else Sort($\mathcal{F}_i$) and insert the first solutions of $\mathcal{F}_i$ into P_{t+1} to fill it
     - iii) $i = i + 1$
   - 4.6 $t = t + 1$

Fig. 9. NSGA-II procedure

remains in the population. This ranking scheme is also used in its precedesor NSGA and is called *Nondominated Sorting* (naming the algorithm). Nevertheless, the original proposal is computionally inefficient ($O(MN^3)$) and the authors provides NSGA-II with a faster version called *Fast Nondominated Sorting*. Once the different levels $\mathcal{F}_i$ have been found, their individuals are ranked with a fitness according to the level they belong to. The individuals from $\mathcal{F}_1$ are ranked with lower values than the solutions of $\mathcal{F}_2$. Thus, minimization of fitness is assumed.

In order to enforce spread Pareto fronts, a second sorting mechanism is employed among the individuals of a level $\mathcal{F}_i$. For that purpose, NSGA-II uses a density estimation value to assign a better ranking to those individuals that are far from other solutions. For each individual in a level $\mathcal{F}_i$, the mean distance to the two nearest individuals is computed. The higher this distance, the more isolated the individual and the lower the density around it. Therefore, those individuals with lower density value are better ranked within a level $\mathcal{F}_i$.

In Figure 9 the algorithm is outlined. It starts creating an initial random parent population $P_{t=0}$ (with N individuals) that is ranked using the Fast Nondominated Sorting. Then, an offspring population Q_t (of size N) is generated using the mutation and crossover operators over the individuals of P_t selected using binary tournament. The individuals of P_t and Q_t are merged into R_t and the Fast Nondominated Sorting is performed. The next generation P_{t+1} is comprised by the N best individuals of R_t. Since R_t contains all the individuals of P_t, elitism is ensured. If the first level of R_t ($\mathcal{F}_1$) has less than N individuals, the best individuals of $\mathcal{F}_2$ are used to fill P_{t+1}. If it is not possible to accomodate all the individuals of a level $\mathcal{F}_i$ into P_{t+1}, the density estimation value is used to select the best ones.

VI. Coding Scheme, Genetic Operators and Objective Functions

In this section it is explained first how the fuzzy visual system has been encoded to be adapted by the EAs. Then, the basis of the genetic operators employed are introduced. Finally, the objective functions used to measure the performance of an individual are also described.

A. Coding Scheme

In our approach, the whole fuzzy visual system is represented using a single chromosome composed of the joining of its membership function parameters. Each fuzzy variable is encoded based on the crossing points of its membership functions and the separation between them using a real coding scheme.

We shall denote the set of m variables of the fuzzy visual system as $Z = \{Z_i \mid i = 0 \dots m\}$ and let us denote the set of membership functions of variable Z_i as $L_i = \{L_i^j | \ j = 0 \dots n+1\}$, where L_i^j denotes the $j-th$ label of the $i-th$ fuzzy variable. Notice that $n+1$ indicates the number of labels for the $i-th$ variable and that this number can be different for each variable. As our system is entirely composed of trapezoidal functions, a membership function can be defined by four parameters as in Equation 10, where $[Left_i, Right_i]$ is the range of the input variable Z_i.

$$L_i^j = [a_i^j \ b_i^j \ c_i^j \ d_i^j] \ / \ a_i^j, b_i^j, c_i^j, d_i^j \in [Left_i, Right_i] \quad (10)$$

In order to reduce the search space, we limit the set of possible configurations forcing the membership functions to cross at level 0.5. This approach has been previously employed in [48] with triangular membership functions. In our case, the use of this constraint is forced to be accomplished by the following restriction:

$$a_i^{j+1} = c_i^j \ \wedge \ b_i^{j+1} = d_i^j$$

Consequently, the crossing point P_i^j of two consecutive labels L_i^j and L_i^{j+1} is calculated as expressed in Equation 11:

$$P_i^j = \frac{c_i^j + d_i^j}{2} = \frac{a_i^{j+1} + b_i^{j+1}}{2} \quad (11)$$

Therefore, for each variable Z_i, there is a set of n crossing points $P_i = \{P_i^j \mid j = 1 \dots n\}$ that have to be learned by the EA to obtain the maximum performance. The variable P_i^j refers to the $j-th$ crossing points of the $i-th$ variable of the system. A range of allowed positions $[Pleft_i^j, Pright_i^j]$ is defined for each crossing point P_i^j instead of allowing EAs freely select any value. This range is calculated using Equations 12

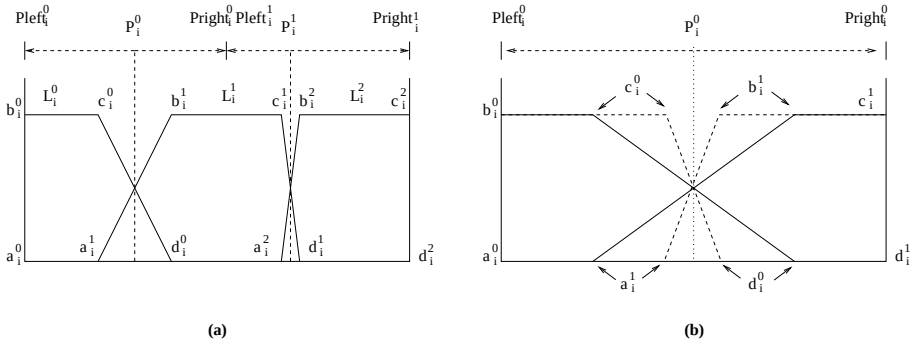


Fig. 10. (a) Representation of a variable with 3 labels. (b) Representation of a variable for different values of s_i^j

and 13 and it limits the definition interval of each crossing point to an interval around its position in the initial fuzzy system created by the expert. This restriction has two purposes. On the one hand, it limits the search space of each crossing point to a range that the expert considered appropriate when he created the knowledge base. On the other hand, it avoids both the relative displacement and the overlap of the membership functions given by the initial system, thus preserving the initial meaning given by the expert to each label. In Figure 10(a), it can be seen an example of a fuzzy variable Z_i with three membership functions $\{L_i^0, L_i^1, L_i^2\}$. The variable has two crossing points $\{P_i^0, P_i^1\}$. As it can be seen, the search space for each P_i^j is independent, thus avoiding both the overlap and the relative displacement between the membership functions. The range of each crossing point is calculated at the start of the process, with respect to the initial fuzzy partition values (as usual in genetic tunning approaches [18]), and remains the same during the whole evolution process.

$$Pleft_i^j = \begin{cases} Left_i & if\ j = 0 \\ \frac{P_i^j + P_i^{j-1}}{2} & otherwise \end{cases} \quad (12)$$

$$Pright_i^j = \begin{cases} Right_i & if\ j = n \\ \frac{P_i^j + P_i^{j+1}}{2} & otherwise \end{cases} \quad (13)$$

In order to increase the capability for adjusting the membership functions, a parameter that indicates the separation between them is used. It represents the separation of the upper points (c_i^j and b_i^{j+1}) of two consecutive membership functions L_i^j and L_i^{j+1} to its crossing point P_i^j. We shall call this parameter $s_i^j \in [0,1]$ and it will be calculated as expressed in Equation 14.

$$s_i^j = \frac{P_i^j - c_i^j}{min\{P_i^j - Pleft_i^j, Pright_i^j - P_i^j\}} \quad (14)$$

When $s_i^j = 0$, there is no separation between c_i^j and b_i^{j+1}, i.e., $c_i^j = b_i^{j+1}$. On the other hand, if $s_i^j = 1$, the separation between c_i^j and b_i^{j+1} is the maximum allowed by the limits $[Pleft_i^j, Pright_i^j]$ of the crossing point P_i^j. To clarify the utility of this parameter, Figure 10(b) shows the example of a crossing point P_i^0 and the corresponding membership functions for two different values of s_i^0. While the membership functions represented by the solid lines correspond to $s_i^0 = 0.5$, the membership functions represented by dashed lines correspond to $s_i^0 = 0.25$. This parameter can be viewed as the degree of fuzzification, i.e., when the parameter s_i^j is 0 it corresponds to an interval discretization with no fuzziness at all [49]. Continuing the notation previously employed, the subscript i refers the to $i-th$ variable and the superscript j to the $j-th$ element into this variable.

Tuning of trapezoidal membership functions using real coding was also performed in [19], but the approach explained in this work reduces the number of parameters employed for each variable and thus the search space for the EA.

A complete fuzzy visual system is represented by a chromosome $\vec{x}^i$. It is encoded by joining the set of membership function definition parameters and the three threshold values $\{t_1, t_2, t_3\} \in [0,1]$. In our system, there is a total of 7 input variables and 4 four output ones. All of them produce a total of 25 crossing points and their corresponding 25 parameters for estabilishing the degree of fuzzification s_i^j. These 50 parameters plus the 3 threshold values sum up a total of 53 parameters that are required for encoding our fuzzy visual system. We will denote the elements of each $\vec{x}^i$ by $(x_0^i, ..., x_{52}^i)$.

B. Genetic Operators

Both a crossover and a mutation operator are required to apply the EAs selected for the tuning process. BLX_α [50] has been selected as crossover operator. This operator works by generating a random real value in an extended range (given by a parameter α) of the parents. Let us suppose that we want to cross over two chromosomes $\vec{x}^a = (x_0^a, ..., x_{58}^a)$ and $\vec{x}^b = (x_0^b, ..., x_{58}^b)$ to obtain an offspring $\vec{x}^c = (x_0^c, ..., x_{58}^c)$. The BLX_α operator generates for each x_i^c a random value in a extended range $[BLX_{Inf}^i, BLX_{Sup}^i]$ given by the parent values x_a^i and x_b^i as shown in Figure 11. Nevertheless, in order to keep the coherence in the values of the offspring, the range $[BLX_{Inf}^i, BLX_{Sup}^i]$ can not exceed the range imposed by the coding scheme for each element x_c^i. It means that if x_c^i is either an $\{t_1, t_2, t_3\}$ or a s_k^j parameter then $[BLX_{Inf}^i, BLX_{Sup}^i]$ must not exceed the range $[0,1]$. Similarly, if x_c^i is a crossing point P_j^k then $[BLX_{Inf}^i, BLX_{Sup}^i]$ must not exceed the range $[Pleft_j^k, Pright_j^k]$. This operator is applied twice in each pair of parents to generate two offsprings.

The mutation operator has been implemented by randomly selecting an element of $\vec{x}^a$ and assigning a random number in

Fig. 11. Operation mode of the $BLX\alpha$ operator.

the range of possible values. In the case of the CHC algorithm, the perturbation operator has been implemented by randomly altering the 35% of an individual.

C. Objective Functions

In order to evaluate the performance of the solutions generated, a set of validated patterns is required. These patterns must be images of the enviroment where the system is going to work, containing scenes with and without doors. Let us denote by $I = \{I^0, ..., I^n\}$ the images in the image set I. For each image, all the segments must be extracted and those that belong to doorframes manually labeled. For that purpose we have created a specific application that helps in this process. Let us denote all the segments extracted from an image I^i by IS^i (*Image Segments*) and those that belong to doorframes by $DS^i \in IS^i$ (*DoorFrame Segments*). When IS^i is passed to a fuzzy visual system $\vec{x}$, it returns only the set of segments that considers belonging to a door. Let us denote them by SS^i (*Solution Segments*). If the system correctly classifies all the segments, then $SS^i = DS^i$. Otherwise, there may be a subset of SS^i with segments that actually belong to the doorframe, and there may be another subset of SS^i with segments that have been incorrectly considered as belonging to a doorframe. Let us denote the former subset by $CC^i = \{SS^i \bigcap DS^i\}$ (*Correctly Classified*) and the latter by $IC^i = \{SS^i - DS^i\}$ (*Incorrectly Classified*).

In our problem there are two different objectives to be optimized. As first objective, it is desired to achieve a maximum positive detection (TPF), i.e., we want to detect all the segments that belong to doorframes. It can be measured using the function $f_1(\vec{x}) \in [0, 1]$ defined in Equation 15 that is to be maximized. This function is 1 when $\vec{x}$ correctly returns all the segments of I that really belong to doorframes. On the other hand, if it is 0, it means that $\vec{x}$ does not return any correct segment at all.

$$f_1(\vec{x}) = \frac{1}{n} \sum_{i=0}^{n} \frac{|CC^i|}{|DS^i|} \tag{15}$$

As second objective, it is desired that the fuzzy visual system rejects all those segments that are in the image but do not belong to doorframes (TNF). This can be measured using the function $f_2(\vec{x}) \in [0, 1]$ defined in Equation 16. This function is 1 when no incorrect segments are returned by $\vec{x}$ and it is 0 in the opposite case.

$$f_2(\vec{x}) = \frac{1}{n} \sum_{i=0}^{n} \left(1 - \frac{|IC^i|}{|IS^i - DS^i|}\right) \tag{16}$$

When using single-objective approaches, we must combine the two objective functions into a single escalar function. Therefore, we employ Equation 17 that uses a parameter λ to independently weight the importance of f_1 and f_2. The use of λ directs the search towards a single direction of the multiobjective space. This operation can be seen referred in the literature as *plain aggregating approach* [11]. In order to obtain a set of solutions with different trade-offs between their objectives, it is necessary to run the single-objective algorithm for different values of λ.

$$f(\vec{x}) = \lambda f_1(\vec{x}) + (1 - \lambda) f_2(\vec{x}) \tag{17}$$

VII. Experimental Results

The aim of our experimentation is to tune the fuzzy visual system jointly optimizing the two goals previously explained, f_1 and f_2. Therefore, we have employed two single-objective EAs, GGA and CHC, and the three MOEAs, SPEA, SPEA2 and NSGA-II. A large set of images I with and without doors, taken from different angles and distances at the fixed height of our camera, has been created. The set contains 436 images with doors and 186 images without doors. The number of images without doors is inferior because we have experimented in the manual tuning performed in [8] that images with doors are more relevant to achieve a proper tuning.

The validated data set has been split into two subsets. The first one is to be used in the evolution process to evaluate the individuals generated (training set) that contains the 80% of the whole patterns. The other set (test set) has the rest of the patterns and is used to test the final solutions generated by the algorithms.

In order to be able to compare the solutions of the different algorithms, we have set almost the same conditions for the execution of all them. The same initial fuzzy system ($\vec{x}_0$) has been used for all the algorithms. Its labels have been uniformly distributed covering the whole range of the fuzzy variables. The individual $\vec{x}_0$ has the following fitness values $f_1(\vec{x}_0) = 0.841$ and $f_2(\vec{x}_0) = 0.884$.

Both GGA and CHC have been run 10 times using the fitness function showed in Equation 17 for the values $\lambda = [0, 0.11, 0.22, .., 0.88, 1]$ in order to emulate the behavior of the MOEAs. The size of the population has been set to 100 and the number of iterations to 300. For the GGA, the percentage of individuals employed for the selection, crossover and mutation operators are 0.4, 0.5 and 0.1, respectively. In CHC, the $Decrement$ factor has been experimentally set to 0.08.

SPEA, SPEA2 and NSGA-II have been run 10 times using a population of 100 individuals. For SPEA and SPEA2, the number of individuals of the elitist population P^e has been set to 25. The total number of iterations employed for the three algorithms are 300 and the objective functions showed in Equations 15 and 16 have been used.

All the algorithms uses BLX_α with $\alpha = 0.3$ as crossover operator and the random mutation operator.

We have used five different metrics for the comparison of the algorithms: four individual metrics, $\#p$, $\mathcal{M}_2^*$, $\mathcal{M}_3^*$, and $\mathcal{S}$; and one comparison metric, $\mathcal{C}$ (see Appendix B for furhter details about the metrics employed). Metric $\#p$ is the number

Algorithm	#p	$\sigma_{\#p}$	$\mathcal{M}_2^*$	$\sigma_{\mathcal{M}_2^*}$	$\mathcal{M}_3^*$	$\sigma_{\mathcal{M}_3^*}$	$\mathcal{S}$	$\sigma_{\mathcal{S}}$
GGA	3	-	0	-	0.227	-	0.9971	-
CHC	5	-	2	-	0.910	-	0.9624	-
SPEA	19.1	2.26	2.635	0.795	0.503	0.053	0.9930	0.0071
SPEA2	17.4	2.80	3.066	0.783	0.516	0.070	0.9871	0.0120
NSGA-II	29.9	3.46	4.259	1.878	0.514	0.038	0.9972	0.0029

TABLE III

QUALITY METRICS VALUES OF THE PARETO FRONTS OBTAINED BY GGA, CHC, SPEA, SPEA2 AND NSGA-II

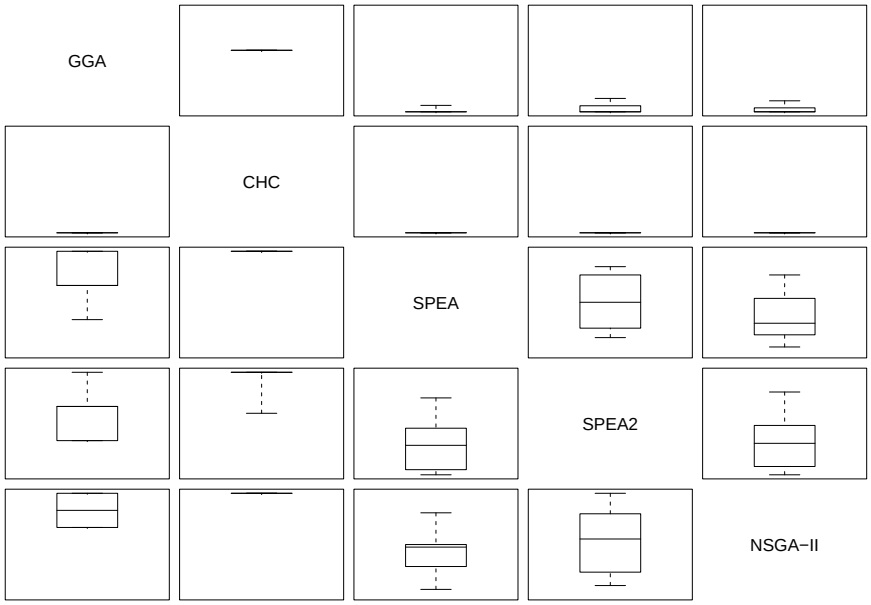


Fig. 12. $\mathcal{C}$ metric values for the different runs of GGA, CHC, SPEA, SPEA2, and NSGA-II

of non-dominated solutions in the final Pareto front. $\mathcal{M}_2^*$ is the generational distance, i.e., the distribution of non-dominated solutions. $\mathcal{M}_3^*$ is the extreme spread that measures the extent of the Pareto front. The reason of using $\mathcal{M}_2^*$ and $\mathcal{M}_3^*$ (instead of $\mathcal{M}_2$ and $\mathcal{M}_3$) comes from the fact that we are interested in that the fuzzy visual systems learned are well distributed in the objective space, to be able to obtain several fuzzy visual systems with different TPF-TNF trade-offs. Please notice that the metric $\mathcal{M}_1^*$ can not be applied because the true Pareto-optimal front is not known in our problem. $\mathcal{S}$ is the hyper-volume covered by the Pareto front. Finally, the comparison metric $\mathcal{C}$ measures the dominance of the solutions of a Pareto front over those of another.

Notice that, as our problem is composed of just two objectives, $\mathcal{M}_3^*$ corresponds to the distance between the objective vectors of the two outer solutions (hence, the maximum possible value is $\sqrt{2} = 1.4142$).

These metrics are appropriate to measure the quality of the generated Pareto fronts. Nevetheless, they can not be applied to the results of the single-objective EAs. Since it is of our interest to compare all the algorithms, the single 10 solutions obtained from the independent 10 runs of each single-objective EA have been gathered to create an aggregated Pareto front for each one of the latter two algorithms. The four metrics previously indicated are also applied to these aggregated Pareto sets.

Table III shows the metrics results for the 10 runs of the three MOEAs and for the aggregated Pareto fronts of GGA and CHC. From left to right, the columns show the average values of the abovementioned metrics in the 10 runs of the MOEAs, and their corresponding standard deviations. In the case of GGA and CHC, as there is a single Pareto front for the independent 10 runs of each algorith (a total of two Pareto fronts: one for GGA and another one for CHC), the standard deviations can not be calculated. Parameter σ^* of $\mathcal{M}_2^*$ has been set to $0.1\sqrt{2}$, i.e., the ten percent of the maximum possible distance. Figure 12 graphically shows as box-plots the values of the $\mathcal{C}$ metric for the different Pareto sets obtained in the 10 runs.

As usually done in the experimental comparison of MOEAs in the specialized literature (see, for example, [51]), we have created a global Pareto front for each MOEA by aggregating the Pareto sets found in each of the independent 10 runs performed, and removing the dominated solutions. Figure 13(a) shows the non-dominated solutions found by SPEA in the different runs. Similarly, Figures 13(b) and 14(a) show these non-dominated solutions found by SPEA2 and NSGA-II. Finally, Figure 14(a) depicts the aggregated Pareto front from the independent 10 runs of GGA (that used to calculate the metrics) and Figure 14(b) for the CHC algorithm. Note that in Figure 14(b) the range of the graph is different from the ranges of the other three graphs in order to represent all

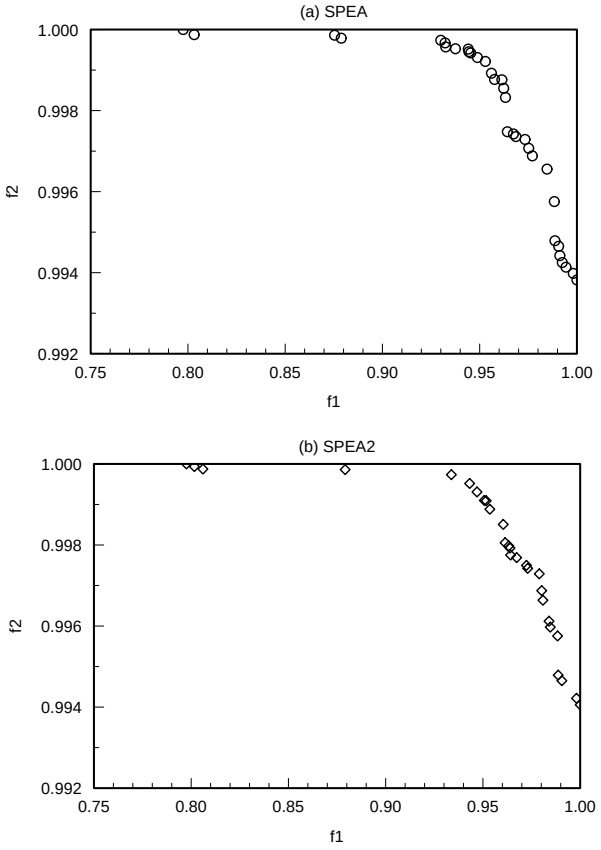


Fig. 13. Aggregated Pareto fronts obtained by (a) SPEA and (b) SPEA2

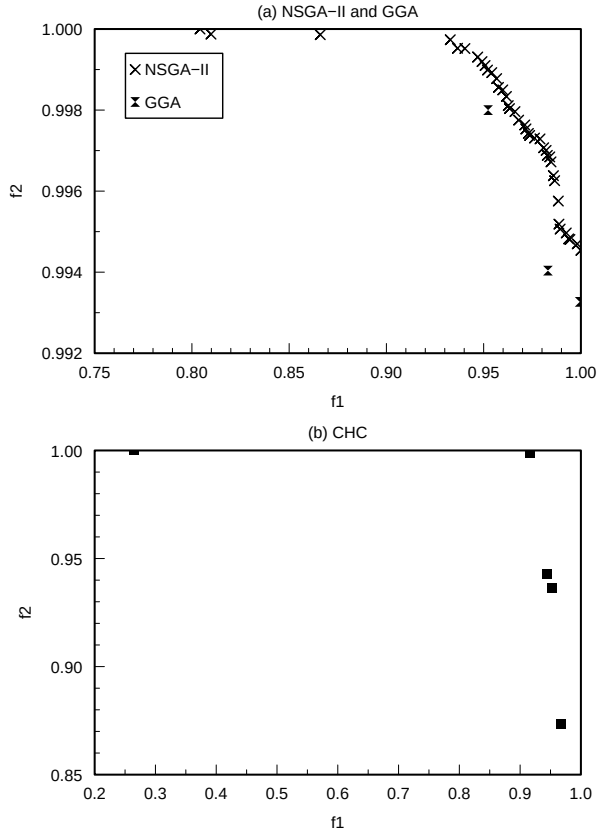


Fig. 14. Aggregated Pareto fronts obtained by (a) NSGA-II and GGA (b) CHC

Algorithm	#p	$\mathcal{M}_2^*$	$\mathcal{M}_3^*$	$\mathcal{S}$
SPEA	33	3	0.4567	0.9998
SPEA2	28	4.9629	0.4565	0.9998
NSGA-II	40	3.3333	0.4486	0.9998

TABLE IV

METRICS FOR THE AGGREGATED PARETO FRONTS OF THE MOEA'S (FIGURES 13 AND 14)

the solutions. Table IV shows the metrics $\#p$, $\mathcal{M}_2^*$, $\mathcal{M}_3^*$ and $\mathcal{S}$ for each one of the aggregated Pareto fronts of the MOEAs. We should remind that, for the case of the single-objective EAs, these metrics were shown in the first two rows of Table III.

In view of the results obtained, several conclusions can be drawn. On the one hand, it can be seen how the three MOEAs clearly outperform the two single-objective EAs considered. As expected, the direct application of a MOEA seems to be a most promising technique to obtain a Pareto set of non dominated solutions for our tuning problem than the consideration of repeated runs of a single-objective EA with different weights for the aggregated fitness function. The direct comparison between both single-objective approaches, GGA and CHC, shows better results of the latter in three of the four individual metrics: the number of solutions #p, the distribution metric $\mathcal{M}_2^*$, and the extension metric $\mathcal{M}_3^*$ (see Table III). However, the results associated to the $\mathcal{M}_3^*$ metric can be confusing if analyzed in isolation. Notice that CHC gets the best value of the five algorithms considered in this metric, with a significant difference (almost the double) with respect to the remainder. If we have a look to Figure 14, we recognize that the reason why the Pareto fronts generated by CHC are so spread in the problem space is because they have not converged properly to the real non dominated solutions, which are actually obtained by the remaining algorithms. Besides, GGA outperforms CHC in the remaining individual metric, the area $\mathcal{S}$, and in the comparison metric $\mathcal{C}$. Notice from the top left part of Figure 12 that, while no solution of CHC dominates the GGA ones in any case, the Pareto sets generated by GGA dominate CHC ones to a high degree.

Focusing the analysis on the three MOEAs, it can be first

Algorithm	$min(f_1)$	$min(f_2)$	$max(f_1)$	$max(f_2)$	μ_{f_1}	μ_{f_2}	σ_{f_1}	σ_{f_2}
GGA	0.9834	0.9865	1	0.9961	0.9944	0.9897	0.0073	0.0042
CHC	0.7107	0.8962	0.9834	1	0.9245	0.9615	0.0855	0.0305
SPEA	0.9049	0.9837	1	1	0.9826	0.9949	0.0138	0.0035
SPEA2	0.9132	0.9879	1	1	0.9836	0.9954	0.0146	0.0028
NSGA-II	0.9132	0.9900	1	1	0.9898	0.9955	0.0133	0.0027

TABLE V

RESULTS OF THE BEST INDIVIDUALS OF THE ALGORITHMS OVER THE TEST SET

noticed how SPEA and SPEA2 perform in a similar way, although SPEA seems to be a little bit more adapted to our fuzzy visual system tuning problem than its improved version SPEA2. While the latter only outperforms the former in the distribution metric $\mathcal{M}_2^*$ (3.066 versus 2.635) and also, although very slightly, in the extension metric $\mathcal{M}_3^*$ (0.516 versus 0.503) (see Table III), SPEA gets better values in the remaining three metrics. It founds more non dominated solutions (19.1 versus 17.4 in average), gets a slightly better value in the area metric $\mathcal{S}$ (0.9930 versus 0.9871), and the solutions in its Pareto sets clearly dominate those in SPEA2 ones in view of the boxplots of Figure 12.

The global best choice seems to be NSGA-II. Although the $\mathcal{C}$ metric show that the Pareto sets generated by this MOEA are of similar quality to those of SPEA, the former ones get the best results in every individual metric but in $\mathcal{M}_2^*$, in which CHC obtains the (false) best value as mentioned before. Besides, NSGA-II takes advantage of applying elitism in the main population, without needing an external one, to generate the largest Pareto sets with an average of 29.9 solutions. In Figure 14, it can be seen how this fact results in a significantly better distributed aggregated Pareto front, with a lesser number of gaps than SPEA and SPEA2 ones.

Finally, in order to verify that the solutions generated do not overfit the training data, we have evaluated their accuracy on the images of the test set. The solutions evaluated are those depicted in the Pareto fronts of Figures 13 and 14. Table V shows in columns for all the algorithms, from left to right: the minimum, maximum, average and standard deviation of the objective functions over the patterns of the test set. As it can be noticed, the results show that the solutions generated are valid and do not overfit the training data.

VIII. CONCLUSIONS

In this paper, an evolutionary methodology to tune an hierarchical fuzzy visual system for door detection has been proposed with the aim of adapting the system structure to the specific characteristics of the environment where the mobile robot using the system is located. Since the system accuracy is measured by two different, conflicting criteria, the positive and negative rates of door detection, the tuning task becomes a multiobjective problem. Two different single-objective and three different multiobjective EAs have been considered for the problem solving, showing the better performance of the latter over the former by means of a sound experimental study.

ACKNOWLEDGMENTS

This work was supported by the Spanish Ministerio de Ciencia y Tecnología under projects TIC2003-04900 and TIC2003-00877, including FEDER fundings.

REFERENCES

[1] J. Wiley, "Mobile robot navigation using artificial landmarks," *Journal of Robotic Systems*, vol. 14, pp. 93–106, 1997.

[2] H. Li and S. Yang, "A behavior-based mobile robot with a visual landmark-recognition system," *IEEE/ASME Transactions on Mechatronics*, vol. 8, pp. 390–400, 2003.

[3] K. Tashiro, J. Ota, Y. Lin, and T. Arai, "Design of the optimal arrangement of artificial landmarks," *IEEE International Conference on Robotics and Automation*, pp. 407–413, 1995.

[4] D. Scharstein and A. J. Briggs, "Real-time recognition of self-similar landmarks," *Image and Vision Computing*, vol. 19, pp. 763–772, 2001.

[5] E. Aguirre and A. González, "A fuzzy perceptual model for ultrasound sensors applied to intelligent navigation of mobile robots," *Applied Intelligence*, vol. 19, no. 3, pp. 171–187, 2003.

[6] R. Muñoz-Salinas, E. Aguirre, M. García-Silvente, and M. Gómez, "A multi-agent system architecture for mobile robot navigation based on fuzzy and visual behaviours," *To appear in Robotica*, 2005.

[7] E. Aguirre and A. González, "Fuzzy behaviors for mobile robot navigation: Design, coordination and fusion," *International Journal of Approximate Reasoning*, vol. 25, pp. 255–289, 2000.

[8] R. Muñoz-Salinas, E. Aguirre, M. García-Silvente, and A. González, "Door-detection using artificial vision and fuzzy logic," *WSEAS Transactions on Systems*, vol. 10, pp. 3047–3052, 2004.

[9] T. Bäck, D. Fogel, and Z. Michalewicz, Eds., *Handbook of Evolutionary Computation*. IOP Publishing Ltd and Oxford University Press, 1997.

[10] F. Moreno-Velo, I. Baturone, R. Senhadji, and S. Sanchez-Solano, "Tuning complex fuzzy systems by supervised learning algorithms," in *The 12th IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2003)*, vol. 1, 2003, pp. 226–231.

[11] K. Deb, *Multi-objective Optimization using Evolutionary Algorithms*. Wiley, 2001.

[12] C. A. Coello, D. A. Van Veldhuizen, and G. B. Lamant, *Evolutionary Algorithms for Solving Multi-objective Problems*. Kluwer Academic Publishers, 2002.

[13] D. E. Goldberg, "Genetic algorithms in search, optimization, and machine learning," *Addison-Wesley, Reading, MA*, 1989.

[14] L. J. Eshelman, "The CHC adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination," in *First Workshop on Foundations of Genetic Algorithms*. Morgan Kaufmann, 1991, pp. 265–283.

[15] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, 1999.

[16] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength pareto evolutionary algorithm for multiobjective optimization." in *Proc. of Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems (EUROGEN2001)*, 2001, pp. 95–100.

[17] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[18] O. Cordón, F. Herrera, F. Hoffman, and L. Magdalena, *Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knwoledge Bases*. World Scientific Publishing, 2001.

[19] F. Herrera, M. Lozano, and J. L. Verdegay, "Tuning of fuzzy controllers by genetic algorithms," *International Journal of Approximate Reasoning*, vol. 12, pp. 299–315, 1995.

[20] O. Cordón and F. Herrera, "A three-stage evolutionary process for learning descriptive and approximative fuzzy logic controller knowledge bases," *International Journal of Approximate Reasoning*, vol. 17, no. 4, pp. 369–407, 1997.

[21] L. Magdalena, "Adapting the gain of an FLC with genetic algorithms," *International journal of Approximate reasoning*, vol. 17, no. 4, pp. 327–349, 1997.

[22] H. B. Gurocak, "A genetic-algorithm-based method for tuning fuzzy logic controllers," *Fuzzy Sets and Systems*, vol. 108, no. 1, pp. 39–47, 1999.

[23] J. Casillas, O. Cordón, M. J. del Jesus, and F. Herrera, "Genetic tuning of fuzzy rule deep structures preserving interpretability and its interaction with fuzzy rule set reduction," *IEEE Transactions on Fuzzy Systems*, vol. 13, no. 1, pp. 13–29, 2005.

[24] O. Cordón, F. Gomide, F. Herrera, F. Hoffman, and L. Magdalena, "Ten years of genetic fuzzy systems: current framework and new trends," *Fuzzy sets and systems*, vol. 141, pp. 5–31, 2004.

[25] V. Changkon and Y. Y. Haimes, *Multiobjective Decision Making Theory and Methodology*. North-Holland, 1983.

[26] V. Pareto, *Cours D'Economie Politique*, Rouge, Lousanne, Switzerland, 1896-7.

[27] H. Ishibuchi, T. Murata, and I. B. Turksen, "Single-objective and two-objective genetic algorithms for selecting linguistic rules for pattern classification problems," *Fuzzy Sets and Systems*, vol. 89, no. 2, pp. 135–150, 1997.

[28] L. Gacôgne, "Multiple objective optimization of fuzzy rules for obstacles avoiding by an evolution algorithm with adaptive operators," in *Proc. of the Fifth International Mendel Conference on Soft Computing (Mendel'99)*, 1999, pp. 236–242.

[29] O. Cordón, F. Herrera, M. J. del Jesus, and P. Villar, "A multiobjective genetic algorithm for feature selection and granularity learning in fuzzy-rule based classification systems," in *Joint 9th IFSA World Congress-20th NAFIPS International Conference*, vol. 3, 2001, pp. 1253–1258.

[30] H. Ishibuchi, T. Nakashima, and T. Murata, "Three-Objective Genetic-Based Machine Learning for Linguistic Rule Extraction," *Information Sciences*, vol. 136, no. 1-4, pp. 109–133, 2001.

[31] A. L. Blumel, E. J. Hughes, and B. A. White, "Fuzzy autopilot design using a multiobjective evolutionary algorithm," in *Congress on Evolutionary Computation*, 2000, pp. 54–61.

[32] C. S. Chang, D. Y. Xu, and H. B. Quek, "Pareto-optimal set based multiobjective tuning of fuzzy automatic train operation for mass transit system," in *IEE Proc. Electric Power App*, vol. 145, no. 5, 1999, pp. 577–583.

[33] U. Nehmzow and C. Owen, "Robot navigation in the real world: Experiments with manchester's forty two in unmodified, large enviroments," *Robotics and Autonomous Systems*, vol. 33, pp. 223–242, 2000.

[34] P. Cariñena, C. Regueiro, A. Otero, A. Bugarin, and S. Barro, "Landmark detection in mobile robotics using fuzzy temporal rules," *IEEE Transactions on Fuzzy Systems*, vol. 12, pp. 423–435, 2004.

[35] M. Masliah and R. Albrecht, "The mobile robot surrogate method for developing autonomy," *IEEE Transactions on Robotics and Automation*, vol. 14, pp. 314–320, 1998.

[36] A. Otero, P. Félix, C. Regueiro, M. Rodríguez, and S. Barro, "A model to perform knowledge-based temporal abstraction over multiple signals," in *TIME-ICTL 2003*, 2003, pp. 128–136.

[37] G. Cicirelli, T. D'orazio, and A. Distante, "Target recognition by component for mobile robot navigation," *Journal of Experimental and Theoretical Artificial Intelligence*, vol. 15, no. 3, pp. 281–297, 2003.

[38] D. Kim and R. Nevatia, "A method for recognition and localization of generic objects for indoor navigation," *Image and Vision Computing*, vol. 16, no. 11, pp. 729–743, 1998.

[39] O. Cetin and A. Erden, "Improvements of an image processing system for autonomous mobile robot navigation," in *Proc. 5^{th} Int. Conf. on Mechatronics Design and Modelling*, 2001.

[40] I. Monasterio, E. Lazkano, I. Rañó, and B. Sierra, "Learning to traverse doors using visual information," *Mathematics and Computer in Simulation*, vol. 60, pp. 347–356, 2002.

[41] S. A. Stoeter, F. L. Mauff, and N. P. Papanikolopoulos, "Real-time door detection in cluttered enviroments," *Proceeding of the 15th IEEE International Symposium on Intelligent Control (ISIC 200)*, pp. 187–191, 2000.

[42] M. A. Abidi and R. C. Gonzalez, *Data Fusion in Robotics and Machine Intelligence*. Academic Press, 1992.

[43] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, pp. 679–698, 1986.

[44] P. Hough, "Method and means for recognizing complex patterns," *U.S. Patent 3069654*, 1962.

[45] G. Foresti, "A real-time hough-based method for segment detection in complex multisensor images," *Real-Time Imaging*, vol. 6, pp. 93–111, 2000.

[46] T. Bäck, U. Hammel, and H.-P. Schwefel, "Evolutionary computation: Comments on the history and current state," *Evolutionary Computation*, vol. 1, no. 1, pp. 3–17, 1993.

[47] N. Srinivas and K. Deb, "Multi-Objective function optimization using non-dominated sorting genetic algorithms," *Evolutionary Computation*, vol. 2, pp. 221–248, 1995.

[48] C. Karr, "Genetic algorithms for fuzzy controllers," *AI Expert*, vol. 6, no. 2, pp. 26–33, 1991.

[49] H. Ishibuchi, T. Nakashima, and M. Nii, *Classification and Modeling with Linguistic Information Granules: Advances Approaches to Linguistic Data Mining*. Springer, 2004.

[50] L. J. Eshelman and D. Schaffer, "Real-coded genetic algorithms and interval-schemata," in *Second Workshop on Foundations of Genetic Algorithms*. Morgan Kaufmann, 1993, pp. 187–202.

[51] E. Zitzler, K. Deb, and L. Thiele, "Comparison of Multiobjective Evolutionary Algorithms: Empirical Results," *Evolutionary Computation*, vol. 8, no. 2, pp. 173–195, 2000.

APPENDIX

A. Evolutionary Multiobjective Optimization

Generally speaking, multiobjective problems can be formulated as: *"finding the best solution $\vec{x} = [x_1, ..., x_n]^T$ to a problem achieving the maximum value of the objective function $\vec{f}(\vec{x}) = [f_1, ..., f_n]^T$"* . The term $\vec{x}$ refers to a vector solution, $\vec{f}(\vec{x})$ refers to the objective function that evaluates $\vec{x}$ according to some criteria and let us denote the set of objectives to evaluate by O.

When using single-objective algorithms to solve the problem, it is usual to transform the different objectives functions into an real scalar value (*plain-aggregating* approach [11]) in the following way:

$$\vec{f}(\vec{x}) = \beta_1 f_1 + \beta_2 f_2 + \cdots + \beta_n f_n$$

where

$$\sum_{i=1..n} \beta_i = 1$$

The different β_i values independently weight one objective against the others. This approach, from the multiobjective point of view, consist in directing the search process to only one direction of the multiobjective space.

Nevertheless, it is also possible to employ a *Pareto-based* approach to solve the problem. In that approach, the goal of the optimization process is to find the ideal vector $\vec{x}^i$ that optimizes all the elements of the objective function. The optimization concept depends on the way the problem is formulated. It could be either a minimization or a maximization of $\vec{f}(\vec{x})$. If the optimization process is a maximization one, a solution $\vec{x}^i$ of the set of found solutions F *dominates* another solution $\vec{x}^j$ if:

$$\forall_{k \in O} f_k(\vec{x}^i) \geq f_k(\vec{x}^j) \wedge \exists_k : f_k(\vec{x}^i) > f_k(\vec{x}^j)$$

The set of non-dominated solutions found is denominated *Pareto set*. Hence, Pareto-based techniques allow us to find a set of non-dominated solutions, each one with a different trade-off of the objectives, thus avoiding to use an importance factor

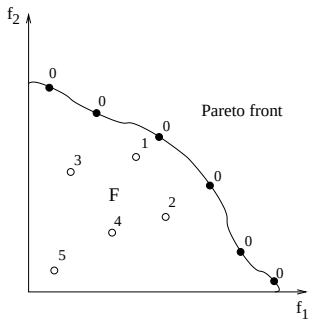


Fig. 15. An example of ranking several solutions in EMO.

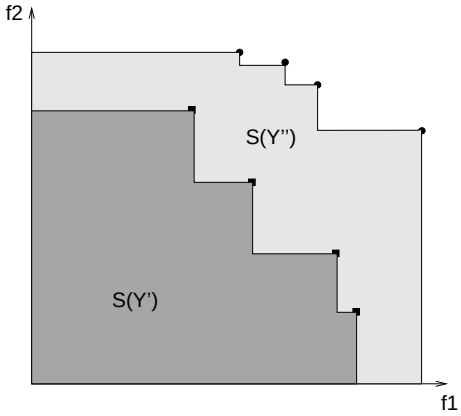


Fig. 16. Example of metric $\mathcal{S}$ applied to two Pareto fronts

β_i for each objective. Once a set of non-dominated solutions is given by the multiobjective technique, they can be analyzed and it is possible to select a unique solution with the most suitable trade-off at the light of the results.

EAs have been employed to solve multiobjective problems using plain-aggregating approaches as well as both *population-based non-Pareto* approaches and *Pareto-based* approaches. This framework has been referred in the literature by the term *EMO* (Evolutionary Multiobjective Optimization) and much work has been done since then to the Pareto-optimum.

Pareto-based approaches can be divided into two different groups: first and second generation [12]. The difference between them is in the use of elitism. First generation algorithms are non-elitist multiobjective algorithms like Niched Pareto Genetic Algorithm (NPGA), Non-dominated Sorting Genetic Algorithm (NSGA) and Multiple-Objective Genetic Algorithm (MOGA). On the other hand, second-generation algorithms employ elitist approaches in order to speed up the performance of the algorithm. Examples of this group are Strength Pareto Evolutionary Algorithm (SPEA) and SPEA2, NSGA-II and NPGA2. For more information about all these algorithms, the interested reader is referred to [11], [12].

MOEAs are able to optimize $\vec{f}(\vec{x})$ ranking each individual according to its proximity to the Pareto front, i.e., non-dominated solutions have the highest rank and the rest of the solutions are ranked according to some criteria. In Figure 15, there is depicted a typical situation in a multiobjective optimization search. The filled circles represent the individuals of the population that belong to the Pareto set and the empty circles represent dominated solutions of the discovered space F. The numbers under each solution represent a possible ranking employed by a MOEA. While the elements of the Pareto set are ranked with the lowest values, the rest of solutions are ranked with higher values

B. Measuring the performance of MOEAs: EMO metrics

In multiobjective optimization, the definition of quality is more complex than for single-objective optimization problems. The multiobjective optimization process involves several objectives itself:

- The distance of the resulting Pareto front to the Pareto-optimal front must be minimized.
- A good distribution of the solutions found is desirable. The assessment of this criterion might be based on a certain distance metric.
- The extent of the obtained Pareto front must be maximized.

Several quantitative metrics have been proposed in the literature to formalize the above definition (or parts of it) [11], [12], [51]. Those employed in this paper are defined below.

Given a set of pairwise non-dominated decision vectors $X' \subseteq X$, a neighborhood parameter $\sigma > 0$ (to be chosen appropriately), and a distance metric $\|\cdot\|$:

1) The function $\mathcal{M}_1$ gives the average distance to the Pareto-optimal set $(\bar{X}) \subseteq X$:

$$\mathcal{M}_1(X') := \frac{1}{|X'|} \sum_{a' \in X'} min\{\|a' - \bar{a}\|; \bar{a} \in \bar{X}\} \quad (18)$$

2) The function $\mathcal{M}_2$ takes the distribution in combination with the number of non-dominated solutions found into account:

$$\mathcal{M}_2(X') := \frac{1}{|X' - 1|} \sum_{a' \in X'} |\{b' \in X'; \|a' - b'\| > \sigma\}| \quad (19)$$

3) The function $\mathcal{M}_3$ considers the extent of the front described by X':

$$\mathcal{M}_3(X') := \sqrt{\sum_{i=1}^{m} max\{\|a'_i - b'_i\|; a', b' \in X'\}} \quad (20)$$

Analogously, [51] defines three metrics $\mathcal{M}_1^*$, $\mathcal{M}_2^*$, and $\mathcal{M}_3^*$ on the objective space. Let $Y', \bar{Y} \subseteq Y$ be the sets of objective vectors that correspond to X' and $\bar{X}$, respectively, and $\sigma^* > 0$ and $\|\cdot\|^*$ as before:

$$\mathcal{M}_1^*(Y') := \frac{1}{|Y'|} \sum_{p' \in Y'} min\{\|p' - \bar{p}\|^*; \bar{p} \in \bar{Y}\} \quad (21)$$

$$\mathcal{M}_2^*(Y') := \frac{1}{|Y' - 1|} \sum_{p' \in Y'} |\{q' \in Y'; \|p' - q'\|^* > \sigma^*\}| \quad (22)$$

$$\mathcal{M}_3^*(Y') := \sqrt{\sum_{i=1}^{n} max\{\|p'_i - q'_i\|^*; p', q' \in Y'\}} \quad (23)$$

Additionally to the latter, in [15] Zitzler et al. proposed the metric $\mathcal{S}(Y')$, also called size of the space covered, that measures the volume enclosed by the Pareto front Y'. In a maximization problem where the optimal objective values are equal to 1, $\mathcal{S}(Y') = 1$ means that the algorithm has reached the optimal solution and lower values for this metric indicates a lower quality of the Pareto front. In our case, as there are only two objectives, $\mathcal{S}(Y')$ measures the area covered by the Pareto front by adding the areas covered by each individual point. In Figure 16, the areas covered by two possible Pareto fronts are shown. It can be noticed that the outer Pareto front (Y'') dominates all the solutions of the inner one (Y'), thus the area $S(Y'') > S(Y')$.

Finally, the $\mathcal{C}$ metric [15] is introduced in order to evaluate the dominance of one Pareto set over another. This metric is used to evaluate the degree in which the solutions of a Pareto set cover the solutions of another. Given two Pareto sets Y' and Y'', the function $\mathcal{C}$ can be calculated using Equation 24. When $\mathcal{C}(Y', Y'') = 1$ it means that all solutions in Y'' are dominated by solutions in Y'. The value $\mathcal{C}(Y', Y'') = 0$ means that none of the solutions of Y'' are dominated by the set Y'. It is important to point out that this function is not conmutative, therefore it is neccessary to calculate both $\mathcal{C}(Y', Y'')$ and $\mathcal{C}(Y'', Y')$.

$$\mathcal{C}(Y', Y'') = \frac{|\{p'' \in Y''; \exists\ p' \in Y' : p'\ dominates\ p''\}|}{|Y''|} \tag{24}$$