J Intell Robot Syst (2013) 69:189–205
DOI 10.1007/s10846-012-9791-5

Cross-Entropy Optimization for Scaling Factors of a Fuzzy Controller: A See-and-Avoid Approach for Unmanned Aerial Systems

Miguel A. Olivares-Mendez · Luis Mejias · Pascual Campoy · Ignacio Mellado-Bataller

Received: 21 August 2012 / Accepted: 2 October 2012 / Published online: 14 November 2012

Abstract The Cross-Entropy (CE) is an efficient method for the estimation of rare-event probabilities and combinatorial optimization. This work presents a novel approach of the CE for optimization of a Soft-Computing controller. A Fuzzy controller was designed to command an unmanned aerial system (UAS) for avoiding collision task. The only sensor used to accomplish this task was a forward camera. The CE is used to reach a near-optimal controller by modifying the scaling factors of the controller inputs. The optimization was realized using the ROS-Gazebo simulation system. In order to evaluate the optimization a big amount of tests were carried out with a real quadcopter.

M. A. Olivares-Mendez (✉) · P. Campoy · I. Mellado-Bataller
CAR - Centro de Automática y Robótica, Universidad Politécnica de Madrid, Madrid, Spain
e-mail: miguelolivaresmendez@gmail.com
URL: http://www.vision4uav.eu/?q=miguel/personal

L. Mejias
Australian Research Centre for Aerospace Automation (ARCAA), Queensland University of Technology, GPO Box 2434, Brisbane, QLD 4001, Australia
e-mail: luis.mejias@qut.edu.au

Keywords UAV · UAS · Fuzzy logic · Fuzzy control · Cross-Entropy · Visual servoing · Computer vision · Sense and avoid · See and avoid · Optimization

1 Introduction

When addressing real-world applications, we typically have to deal with systems that are either not-well defined, not-modelled or with a huge solution space. Imprecision, uncertainty, partial truth, and approximation are some of the issues that are well handled by Soft Computing approaches. These capability seems very attractive in real world scenarios, where uncertainty and unmodelled dynamic seems predominant, gaining more and more importance in controlling real systems. Its original definition provided by Zadeh [1] denote systems that *"… exploit the tolerance for imprecision, uncertainty, and partial truth to achieve tractability, robustness, low solution cost, and better rapport with reality"*.

These techniques are also of interest when dealing with highly non-linear (and unmodelled) dynamics, very common in aerospace applications. In particular, they are very well suited to deal with non-linearities that makes some of the problems in the field of aerial robotics intractable. In this work, we propose the use of Soft Computing techniques to address the sense-and-avoid problem [2] for unmanned aerial systems (UAS). Before UASs are allowed to routinely fly in civil airspace, several technological hurdles need to be addressed. For example, sense-and-avoid or safe termination systems are some of the technologies that UAS require before they share the airspace and fly over populated areas [2]. The UAS sector is gaining considerable predominance among researchers nowadays. Industry, academia and general public are placing more attention in UASs to understand the potential benefits UAS could provide to society.

The onboard sense-and-avoid capability can be provided by the use of single or multiple onboard sensors [3–5]. Furthermore, self-contained and passive electro optical (EO) sense-and-avoid systems have the capability to address non-cooperative scenarios at the same it provide an alternative to the Size, Weight and Power (SWaP) limitations of many small-medium size UAS. Onboard EO or cameras have not only the capability to perform sense-and-avoid [6–8] but also they can be used for state estimation [9–11] among others applications.

This paper is structured as follows. In Section 2 the related work is presented. In Section 3 we describe the image processing front-end used in our approaches. In Section 4 we explain the visual servoing approach using fuzzy logic for heading control. The cross-entropy theory is introduced in Section 5. All the software implementation is explained in Section 6. The results of the optimization using the simulator and a comparison of the optimized and non-optimized controller with real tests are presented in Section 7. Finally, concluding remarks and future work are presented in Section 8

2 Related Work

The two of most common application of SC techniques are Fuzzy systems and neural networks. In Hunt et al. [12] a survey of Neural Networks for control systems is presented. In the same way the work of Precup and Hellendoormn [13] presents a survey of industrial control applications with Fuzzy Control. Similar to other types of controllers, SC controllers need to be tuned or optimised manually. In 1992 Zheng [14] defined a tuning sequence for manual tuning of Fuzzy controllers (FC). The optimisation process can be performed at three different scales based on the effects caused to the controller behaviour: The Macroscopic effects are caused by the modification of the scaling factors (SF), which are defined as gains of the inputs and outputs. Medium-size effects which impact the controller when the membership functions (MF) are modified, and Microscopic effects which are present when we modify the output or the weight of each rule. This sequence of effects could be easily understood is we visualize the rule base of the FC as a rule table. A modification of one scaling factor affects the entire rule table. A modified set of membership functions affects one row, one column, or one diagonal in the table. A modified rule only affects one cell of the rule table.

Malhorta et al. [15] presents a macroscopic optimization of PID and PI Fuzzy controllers using genetics algorithms. In [16] Bonissone presents the use of Genetics Algorithms for macroscopic and medium-size optimization of a PI Fuzzy controller. Wei Li [17] presents a medium-size scale optimization using neural networks. In [18], Jang presents an adaptive neural based Fuzzy inference system (ANFIS) that was used to refine the Fuzzy if-then rules, being in this case a microscopic optimization. The learning algorithm is based on the gradient descent and the chain rule proposed by Werbos [19] in the 1970's. Also of interest to the reader is the work of Bonissone et al. in [20], who presents a deep discussions of SC hybrid systems and optimization methodologies with very clear examples of industrial and commercial applications.

In this work, we present a Macroscopic optimization of a Fuzzy controller using the Cross-Entropy method. This novel optimization method is a general Monte Carlo approach to combinatorial and continuous multi-extremal optimization and importance sampling. This method was motivated by an adaptive algorithm for estimating probabilities or rare events in complex stochastic networks [21], which involves variance minimization. A simple modification of the initial algorithm allows to apply it to solve difficult combinatorial optimization problems. Several recent applications demonstrate the power of the CE method like [22] for power system reliability evaluation, and [23] to find the optimal path planning, and [24] use this method for a antenna selection to improve the radio channel capacity, and [25] for motion planning. The uses of this optimization method in control is reduced to two works in the literature. In [26] Bodur presents the use of the CE method to optimize the gains of a classic PID controller to manage the invert pendulum problem in a simulated environment. Haber et al. [27] use the CE method to optimize the scaling factors of a Fuzzy PD controller for cutting force regulation of a drilling process. Experimental results are presented in this work for this high controlled environment process. In our work, we present an optimization of the scaling factors of a Fuzzy PID controller to command a quadrotor for a see-and-avoid task. In comparison with the previous work of [27] we have to face a highly dynamic environment of a flying quadcopter in indoor tests for avoiding collision task using vision.

3 Image Processing Front-End

Visual awareness is achieved by using an onboard forward-looking camera. Images from the camera are then sent for off-board processing in a laptop ground-station. The outcome of the visual processing (and servoing commands) are then sent back to the vehicle using a wifi link.

The avoidance task aims to keep the target in the image plane at constant bearing, either right or left (as seen from image centre). When the object is first detected it is pushed to the edge of the image (far left or right side), and kept at a fixed position that represents a constant relative bearing.

The target is detected by pre-defining a color and then designing an algorithm to highlight this color. The color will be tracked along the image sequence. The tracking is performed by using the Continuously Adaptive Mean Shift [28] (CamShift). This algorithm is based on the mean shift originally introduced by Fukunaga and Hostetler [29]. This algorithm accounts for the dynamic nature of changes in lighting conditions by dynamically adapting to changes in probability distributions of color.

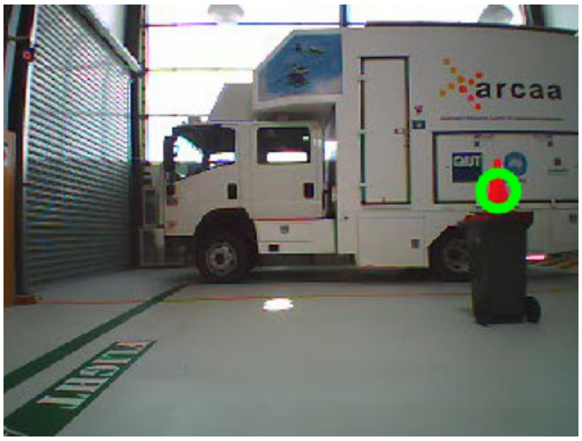

Fig. 1 Image captured with the onboard camera

Using the Camshift algorithm we track and estimate the centre of the color region that describes the object. Figure 1 shows an example of the tracking processes on a red colored object. Using the location of the target in the image we generate desired yaw commands (while keeping forward velocity constant) which in turn will modify the trajectory of the vehicle in order to keep the object at constant relative bearing.

4 Fuzzy Controller

The controller was designed using Fuzzy Logic techniques. The aim of the controller is to generate desired yaw commands for the aerial vehicle based on the location of the target in the image plane. With the commands generated by the controller, the aircraft must avoid the obstacle that is in its trajectory.

The Fuzzy PID controller was implemented using our self-developed library MOFS (Miguel Olivares' Fuzzy Software) [30]. This library has a hierarchical class definition for each part of the fuzzy-logic environment (variables, rules, membership functions, and defuzzification modes) in order to facilitate future updates and make easier the development of controllers based on Fuzzy Logic. These routines have been used in a wide variety of control applications such as autonomous

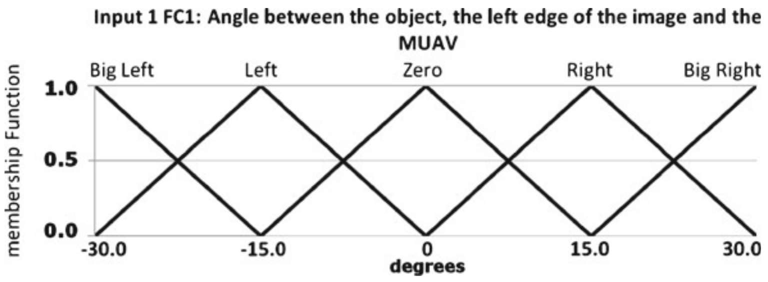

Fig. 2 PID-fuzzy controller: membership function of the first input, the error

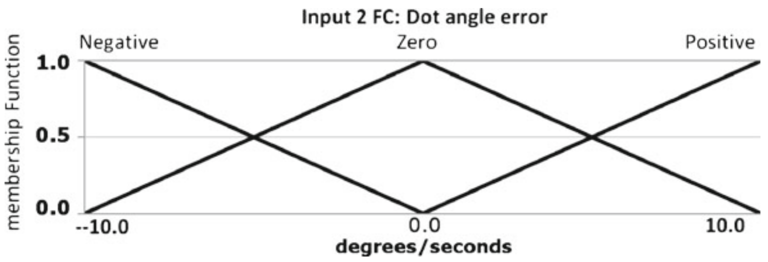

Fig. 3 PID-fuzzy controller: membership function of the second input, the derivative of the error

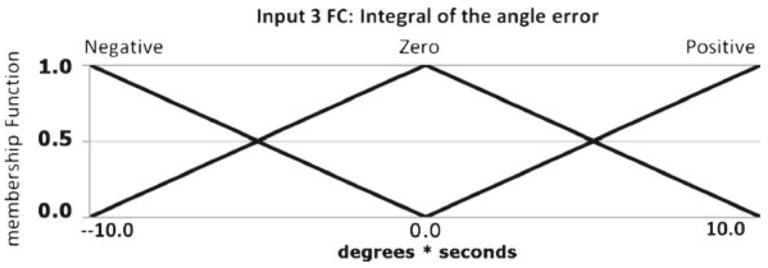

Fig. 4 PID-fuzzy controller: membership function of the third input, the integral of the error

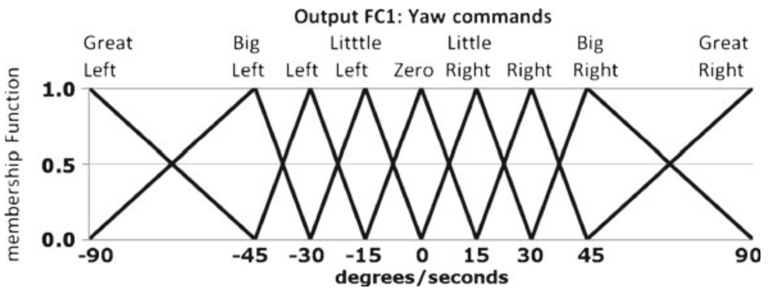

Fig. 5 PID-fuzzy controller: membership function of the output, heading degrees to turn

Table 1 Base of rules with value for the third input (integral of the error) equal to **zero**

Dot	Error				
	Big left	Left	Zero	Right	Big right
Negative	Big left	Left	Little left	Zero	Little right
Zero	Left	Little left	Zero	Little right	Right
Positive	Little left	Zero	Little right	Right	Big right

Table 2 Base of rules with value for the third input (integral of the error) equal to **negative**

Dot	Error				
	Big left	Left	Zero	Right	Big right
Negative	Left	Little left	Zero	Little right	Right
Zero	Little left	Zero	Little right	Right	Big right
Positive	Zero	Little right	Right	Big right	Great right

landing [31], and autonomous road following [32], etc.

The inputs and the outputs were defined using triangular membership functions. The product t-norm is used for the conjunction of the rules and the Height Weight method has been selected for the defuzzification phase (Eq. 1).

$$y = \frac{\sum_{l=1}^{M} \bar{y}^{l} \prod \left(\mu_{B'}(\bar{y}^{l})\right)}{\sum_{l=1}^{M} \prod \left(\mu_{B'}(\bar{y}^{l})\right)} \tag{1}$$

The Fuzzy controller was defined using three inputs and one output. The first input measures the error in degrees between the quadrotor, the object to avoid minus the reference (Fig. 2). The second, is the derivative of the error, as is shown in Fig. 3, and third input, shown in Fig. 4 represents the integral of the error. The output is the commanded yaw that the vehicle needs to turn to keep the object at the desired relative bearing, see Fig. 5. First and second outputs are equivalent to the inputs of the first approach.

The definition of the fuzzy variables uses 45 rules. By the reason that the system have 3 inputs the base of rules has a cube disposition of $5 \times 3 \times 3$. To be easy to the reader to understand the base of rule, we present three matrix of 5×3 with the relation between the first two inputs, error and dot (derivative of time) of the error. Each matrix has a static value of the third input, the integral of the error. Table 1 shows the output values for the variables error and dot, with the integral of the error value equal to zero. The Table 2 shows the output values for the variables error and dot, with the static value for the third variable equal to negative. And finally the Table 3 shows the output values for the variable error and dot, with the static value for the third variable equal to Positive.

5 Cross-Entropy Optimization Method

The Cross-Entropy (CE) method is a new approach in stochastic optimization and simulation. It was developed as an efficient method for the estimation of rare-event probabilities. The CE method has been successfully applied to a number of difficult combinatorial optimization problems. We present an application of this method for optimization of the gains of a Fuzzy controller. Next, we present the method and the Fuzzy controller optimization approach. A deeper explanation of the Cross-Entropy method is presented on [33].

5.1 Method Description

The CE method is iterative and based on the generation of a random data sample $(x_1, ..., x_N)$ in the χ space according to a specified random mechanism. A reasonable option is to use a probability density function (pdf) such as the normal distribution. Let $g(-, v)$ be a family of probability density functions in χ parameterized by a real value vector $v \in \Re$: $g(x, v)$. Let ϕ be a real function on χ, so the aim of the CE method is to find the minimum (like in our case) or maximum of ϕ over χ, and the corresponding states x^* satisfying this minimum/maximum: $\gamma^* = \phi(x^*) = \min_{x \in \chi} \phi(x)$.

In each iteration the CE method generates a sequence of $(x_1, ..., x_N)$ and $\gamma_1 ... \gamma_N$ levels such that γ converges to γ^* and x to x^*. We are concerned

Table 3 Base of rules with value for the third input (integral of the error) equal to **positive**

Dot	Error				
	Big left	Left	Zero	Right	Big right
Negative	Great left	Big left	Left	Little left	Zero
Zero	Big left	Left	Little left	Zero	Little right
Positive	Left	Little left	Zero	Little right	Right

with estimating the probability $l(\gamma)$ of an event $E_v = \{x \in \chi \mid \phi(x) \geq \gamma\}, \gamma \in \Re$.

Defining a collection of functions for $x \in \chi$, $\gamma \in \Re$.

$$I_v(x,\gamma) = I_{\{\chi(x_i)>\gamma\}} = \begin{cases} 1 & if \quad \phi(x) \leq \gamma \\ 0 & if \quad \phi(x) > \gamma \end{cases} \tag{2}$$

$$l(\gamma) = P_v(\chi(x) \geq \gamma) = E_v \cdot I_v(x, v) \tag{3}$$

where E_v denotes the corresponding expectation operator.

In this manner, Eq. 3 transforms the optimization problem into an stochastic problem with very small probability. The variance minimization technique of importance sampling is used, in which the random sample is generated based on a pdf h. Being the sample $x_1, ..., x_N$ from an importance sampling density h on ϕ and evaluated by:

$$\hat{l} = \frac{1}{N} \cdot \sum_{i=1}^{N} I_{\{\chi(x_i)>\gamma\}} \cdot W(x_i) \tag{4}$$

where $\hat{l}$ is the importance sampling and $W(x) = \frac{g(x,v)}{l}$ is the likelihood ratio. The search for the sampling density $h^*(x)$ is not an easy task because the estimation of $h^*(x)$ requires that l be known $h^*(x) = I_{\{\chi(x_i)>\gamma\}} \cdot \frac{g(x,v)}{l}$. So the referenced parameter v^*, must be selected such the distance between h^* and $g(x, v)$ is minimal, thereby the problem is reduced to a scalar case. A way to measure the distance between two densities is the Kullback–Leibler, also known as cross-entropy:

$$D(g,h) = \int g(x) \cdot ln\ g(x)dx - \int g(x) \cdot ln\ h(x)dx \tag{5}$$

The minimization of $D(g(x, v), h^*)$ is equivalent to maximize $\int h^* ln[g(x, v)]dx$ which implies that $\max_v D(v) = \max_v E_p \left(I_{\{\chi(x_i)>\gamma\}} \cdot ln\ g(x, v)\right)$, in terms of importance sampling it can be rewritten as:

$$\max_v \hat{D}(v) = \max \frac{1}{N} \sum_{i=1}^{N} I_{\{\chi(x_i)>\gamma\}} \cdot \frac{p_x(x)}{h(x_i)} \cdot ln\ g(x_i, v) \tag{6}$$

Note that h is still unknown, therefore the CE algorithm will try to overcome this problem by constructing an adaptive sequence of the parameters $(\gamma_t \mid t \geq 1)$ and $(v_t \mid t \geq 1)$.

5.2 Fuzzy Control Optimization Approach

This approach is based on a population-and-simulation optimization [34]. The CE algorithm generates a set of N fuzzy controllers $x_i = (x_{KE}, x_{KD}, x_{KI})$ with $g(x, v) = (g(x_{KE}, v), g(x_{KD}, v), g(x_{KI}, v))$ and calculates the cost function value for each controller. The controllers parameters KE, KD, KI correspond to the gains of the first, second and third input of each controller (Figs. 2–4). Then updates $g(x, v)$ using a set of the best controllers. This set of controllers is defined with the parameter N^{elite}.The process finish when the minimum value of the cost function or the maximum number of iterations is reached, as is shown in the Algorithm 1.

Algorithm 1 Cross-Entropy algorithm for Fuzzy controller optimization

```
1. Initialize $t = 0$ and $v(t) = v(0)$
2. Generate a sample of N controllers: $(x_i(t))_{1 \leq i \leq N})$ from $g(x, v(t))$, being each $x_i = (x_{KEi}, x_{KDi}, x_{KIi})$
3. Compute $\phi(x_i(t))$ and order $\phi_1, \phi_2, ..., \phi_N$ from smallest $(j = 1)$ to biggest $(j = N)$.
   Get the $N^{\text{elite}}$ first controllers $\gamma(t) = \chi_{[N^{\text{elite}}]}$.
4. Update $v(t)$ with $v(t+1) = \arg_v \min \frac{1}{N^{\text{elite}}} \sum_{j=1}^{N^{\text{elite}}} I_{\{\chi(x_i(t)) \geq \gamma(t)\}} \cdot \ln\ g(x_j(t), v(t))$
5. Repeat from step 2 until convergence or ending criterion.
6. Assume that convergence is reached at $t = t^*$, an optimal value for $\phi$ can be obtained from $g(., v(t)^*)$.
```

For this work the Normal (Gaussian) distribution function was selected. The mean μ and the variance σ are estimated for each iteration $h = 1, 2, 3$ parameters (K_e, K_d, K_i) as $\tilde{\mu}_{th} = \sum_{j=1}^{N^{\text{elite}}} \frac{x_{jh}}{N^{\text{elite}}}$ and $\tilde{\sigma}_{th} = \sum_{j=1}^{N^{\text{elite}}} \frac{(x_{jh} - \mu_{jh})^2}{N^{\text{elite}}}$ where $4 \leq N^{\text{elite}} \leq 20$.

The mean vector $\bar{\tilde{\mu}}$ should converge to γ^* and the standard deviation $\bar{\tilde{\sigma}}$ to zero. In order to obtain a smooth update of the mean and the variance we use a set of parameters (β, α, η), where α is a constant value used for the mean, η is a variable value which is applied to the variance to avert the occurrences of 0*s* and 1*s* in the parameter vectors, and β is a constant value which modify the value of $\eta(t)$.

$$\begin{aligned}&\eta(t) = \beta - \beta \cdot (1 - \tfrac{1}{t})^q\\&\hat{\mu}(t) = \alpha \cdot \tilde{\mu}(t) + (1 - \alpha) \cdot \hat{\mu}(t-1)\\&\hat{\sigma}(t) = \eta(t) \cdot \tilde{\sigma} + (1 - \eta) \cdot \hat{\sigma}(t-1)\end{aligned} \tag{7}$$

Where $\hat{\mu}(t-1)$ and $\hat{\sigma}(t-1)$ are the previous values of $\hat{\mu}(t)$ and $\hat{\sigma}(t)$. The values of the smoothing update parameters are $0.4 \le \alpha \le 0.9$, $0.6 \le \beta \le 0.9$ and $2 \le q \le 7$. In order to get an optimized controller different criterion could be chosen, such as the Integral time of the absolute error (ITAE) or the Integral time of the square error (ITSE) or the root mean-square error (RMSE).

6 Software Implementation

The simulation tests were performed using the ROS (Robotics Operative System) and the 3D simulation environment Gazebo [35]. In simulations, a quadcopter model of the starmack ros-pkg developed by Berkeley University [36] was used. The obstacle to avoid was defined by a virtual yellow balloon.

Two external software routines in C++ were developed for accomplish these tests. One is the implementation of the Cross-Entropy method, which is responsible of the optimization process. This program generates a set of controllers, selects the controller to test and when all the controllers are tested, update the pdf with the results to obtain new values of mean and variance of each pdf to generate the new set of controllers to test in the next iteration. The other one is responsible to execute iteratively the ROS-Gazebo system. In order to test all the controllers in the same conditions, the ROS-Gazebo is restarted for each test getting same initial state for all the tests. In each iteration the program send a kill command to close the ROS-Gazebo and start it again loading all the initial parameters needed by the simulator. The Fig. 6 shows the tests flowchart, in which the tasks performed by the Cross-Entropy program are represented with green boxes. The tasks performed by the iteration program are those which are represented by blue diamonds. The blue

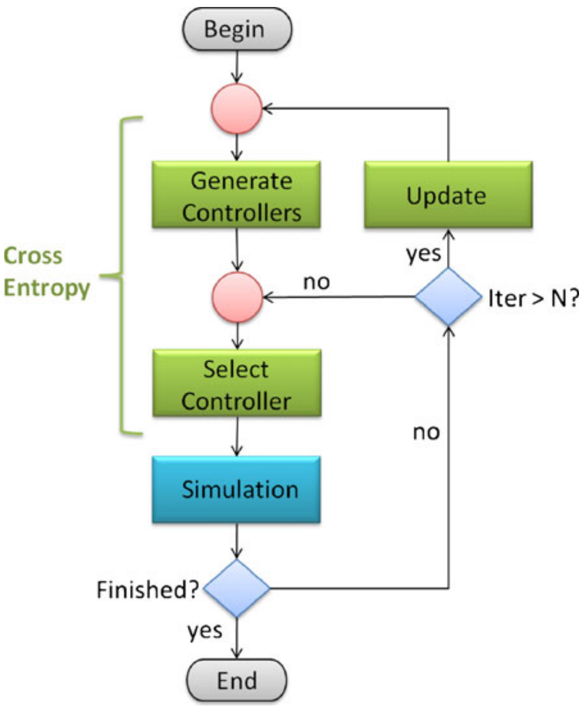

Fig. 6 Flowchart of the optimization process

box titled Simulation represents the ROS-Gazebo process.

Additionally, two nodes were added to the ROS-Gazebo, visual algorithm and Fuzzy controller nodes, respectively (Fig. 7). The visual algorithm which gets the image from the simulated camera onboard the quadcopter and converts it to an OpenCV image for further processing. After

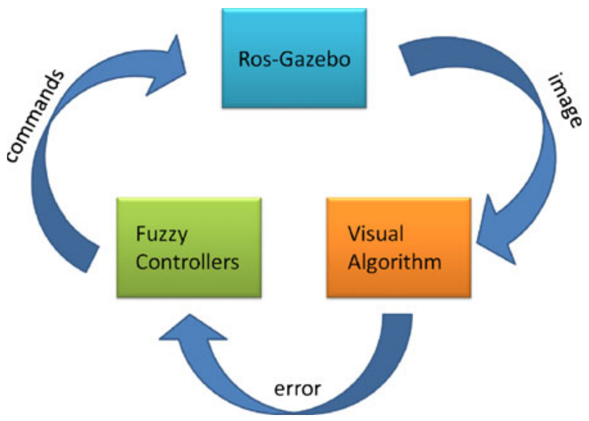

Fig. 7 Interaction between the ROS-Gazebo 3D simulator and the two other process developed for this work

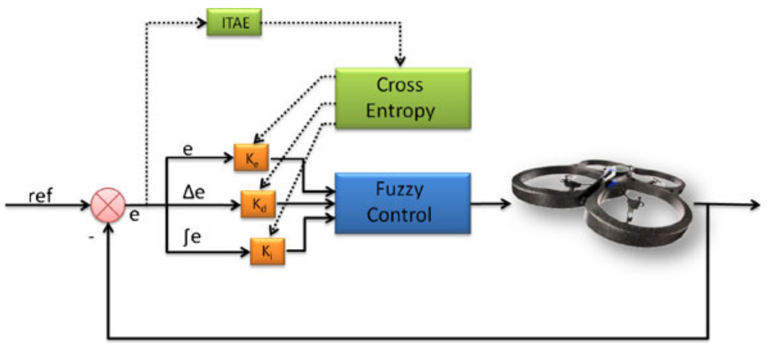

Fig. 8 Control loop with the optimization of the Cross-Entropy method

the image is processed the information obtained is sent to the Fuzzy controller node. The controller evaluates this data to obtain the correct yaw value. Finally, this command is sent to the simulated aircraft in the 3D simulator. One advantage of this simulation environment is that the detection algorithm used in this phase is the same that was used in the real tests.

7 Results

In this section we present the results of the optimization process in simulation and a set of real flight tests to compare the behaviour of the optimised and the non-optimized Fuzzy controllers.

7.1 Optimization Process Using the Simulation Environment

In order to obtain an optimal parameters for a controller we should generally test a large number of different controllers. Testing these controller in the same conditions is challenging. To do this we defined a type of test based on some fixed parameters such as fixed time for each simulation cycle; the quadcopter positioned in front of the object to avoid in a defined starting location and each test is performed sending a constant pitch command to the aircraft of 0.03 m/s. To evaluate the performance of each test we used the Integral Time Absolute Error (ITAE) criterion. Figure 8 shows the control loop during the optimization process. We also used the Root Mean Square Error (RMSE) criterion but similar results. We choice the ITAE error estimator is motivated by the error penalization it imposes at the end of the test. Being more important estimator during a optimization process. The RMSE criterion was used with the real tests because is easier to understand what the performance of the test was, because the result is given in the same unit that is used by the first input of the controller (the error). The cross-

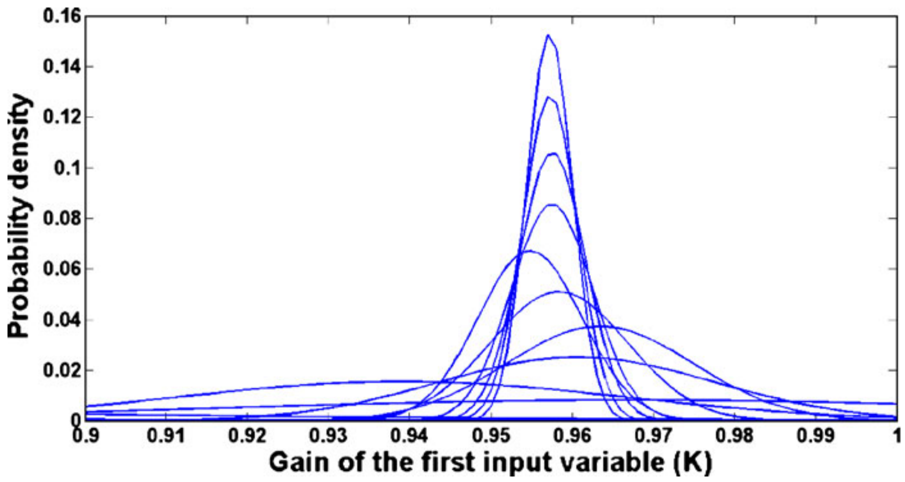

Fig. 9 Evolution of the probability density function for the first input gain. The standard variance converge in 12 iterations to a value of 0.0028 so that the obtained mean 0.9572 can be used in the real tests

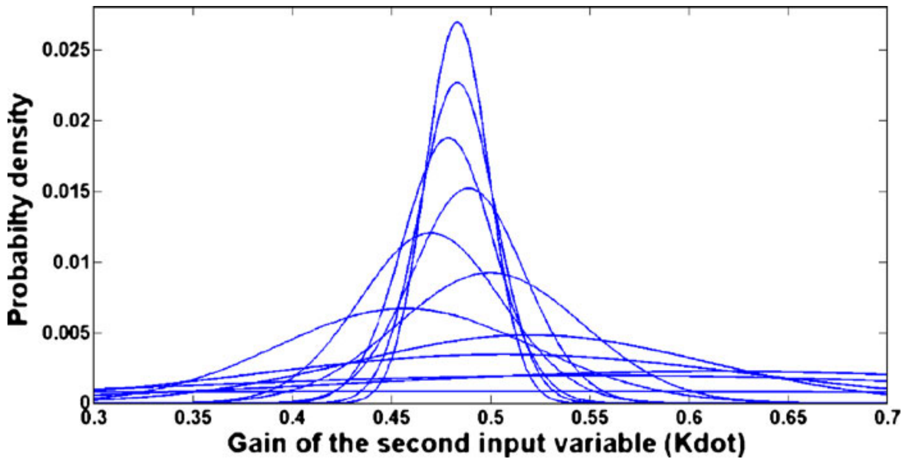

Fig. 10 Evolution of the probability density function for the second input gain. The standard variance converge in 12 iterations to a value of 0.0159 so that the obtained mean 0.4832 can be used in the real tests

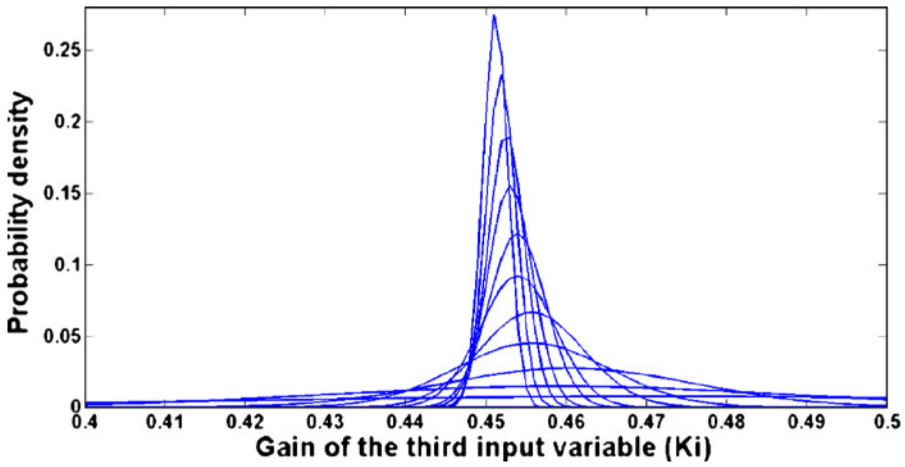

Fig. 11 Evolution of the probability density function for the third input gain. The standard variance converge in 12 iterations to a value of 0.0015 so that the obtained mean 0.4512 can be used in the real tests

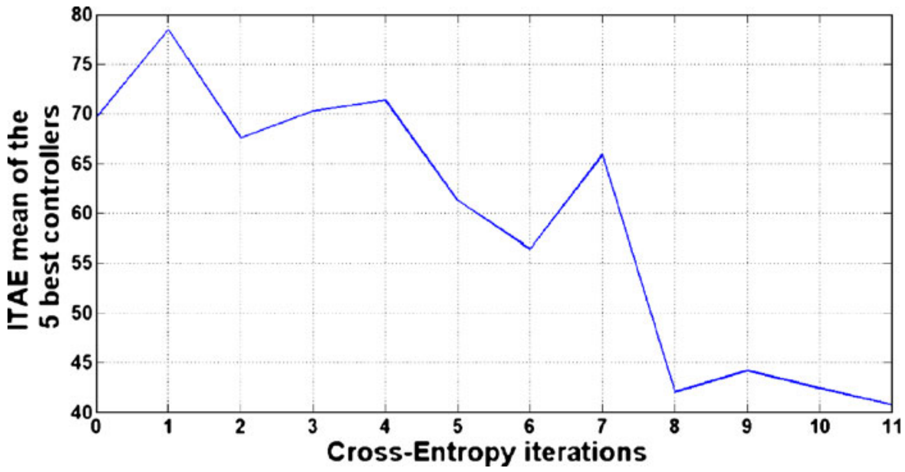

Fig. 12 Evolution of the ITAE error during the 12 Cross-Entropy iterations. The ITAE value of each iteration correspond to the mean of the first 5 of 30 controllers of each iteration

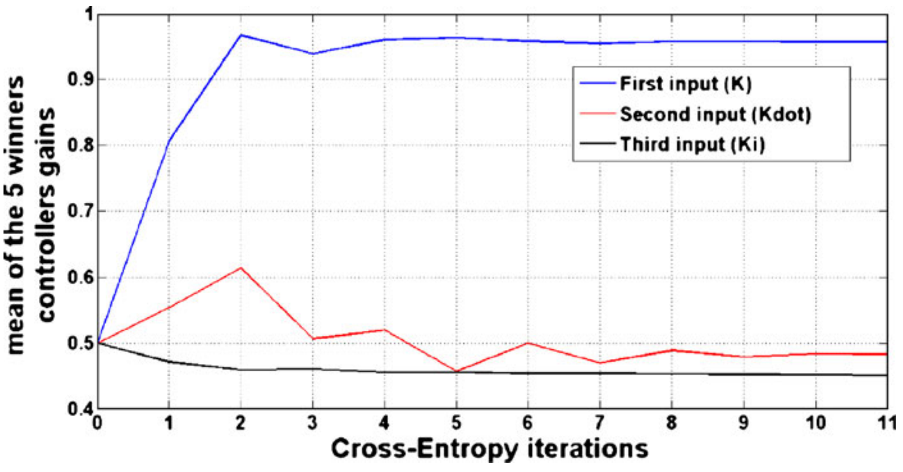

Fig. 13 Evolution of the gains of each input. The value of the gain correspond to the first 5 of 30 controllers of each iterations

entropy system generates $N = 30$ controllers per iteration based on the last update of the probability density function for each gains. From this set of controllers the five with the lowest ITAE value are selected ($Nelite = 5$) to update the next pdf parameters. The initial values for the pdf of all the gains are $\mu(0) = 0.5$, $\sigma(0) = 0.5$. The rest of the parameters of the cross-entropy method are $q = 2$, $\eta(0) = 0$, $\beta = 0.92$, $\alpha(0) = 0$. Those values are based on values reported in [27] and [34].

A number 330 tests were performed to obtain the optimal controller. This process corresponds to 11 updates of the pdf for the gains. Figure 9 shows the evolution of the probability density function of the first input of the controller. The initial mean and sigma for the three gains were 0.5 for both parameters. The final values of the pdf were $mean = 0.9572$ and $sigma = 0.0028$. Figure 10 shows the evolution for the second input with the final values of $mean = 0.4832$ and $sigma = 0.0159$. In the same way Fig. 11 shows the evolution of the pdf for the third input, which finalize with $mean = 0.4512$ and $sigma = 0.0015$. Figure 12 presents the evolution of the mean of the ITAE value of the 5 winners from each set of 30 controllers. The Fig. 13 shows the evolution of the different gains of the controller during the 330 tests.

7.2 Flight Tests

In order to validate and compare the behavior of both controllers we conducted real flights tests. We used a AR.Drone-Parrot [37] platform with our own software routines developed for this purpose [38]. Figure 14 shows the AR.Drone aircraft. A typical orange traffic cone was used as the object to avoid. We recorded the quadrotor trajectory with the maximum precision using the VICON position detection system [39]. The VICON system was used for data logging, no data was used for the control of the quadrotor.

The quadcopter system used in these tests is a commercial-off-the-shelf Parrot AR.Drone. This is an aircraft with two cameras onboard, one forward-looking which has been used in this work, and one downward-looking. The aircraft is connected to a ground station via WiFi connection. A extended explanation of this platform is presented at [37].

Figure 15 shows the control loop of the system once the Cross-Entropy process was remove.

For both controllers the flight tests were performed with predefined constant forward speed (constant pitch angle) during the test. No roll

Fig. 14 Parrot-AR.Drone, the platform used for the real tests

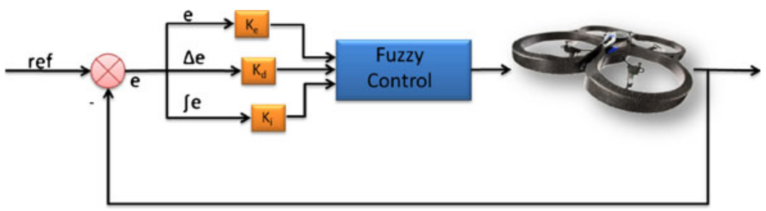

Fig. 15 Control loop with the optimization of the Cross-Entropy method

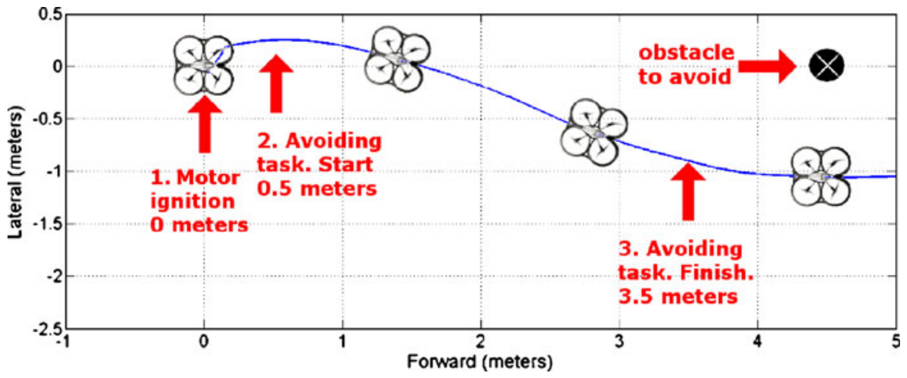

Fig. 16 Explanation of the avoiding task approach. The quadcopter starts at point 0.0 (1. Motor ignition) and flies 0.5 m keeping the obstacle to avoid in the center of the image (2. Avoiding task. Start). Then the reference to one of the edge of the image is added to the position of the obstacle in the image plane until 3.5 m (3. Avoiding task. Finish). The quadrotor continues. The last yaw command is send after the avoiding task is finished. The obstacle to avoid is at point (0, 4.5)

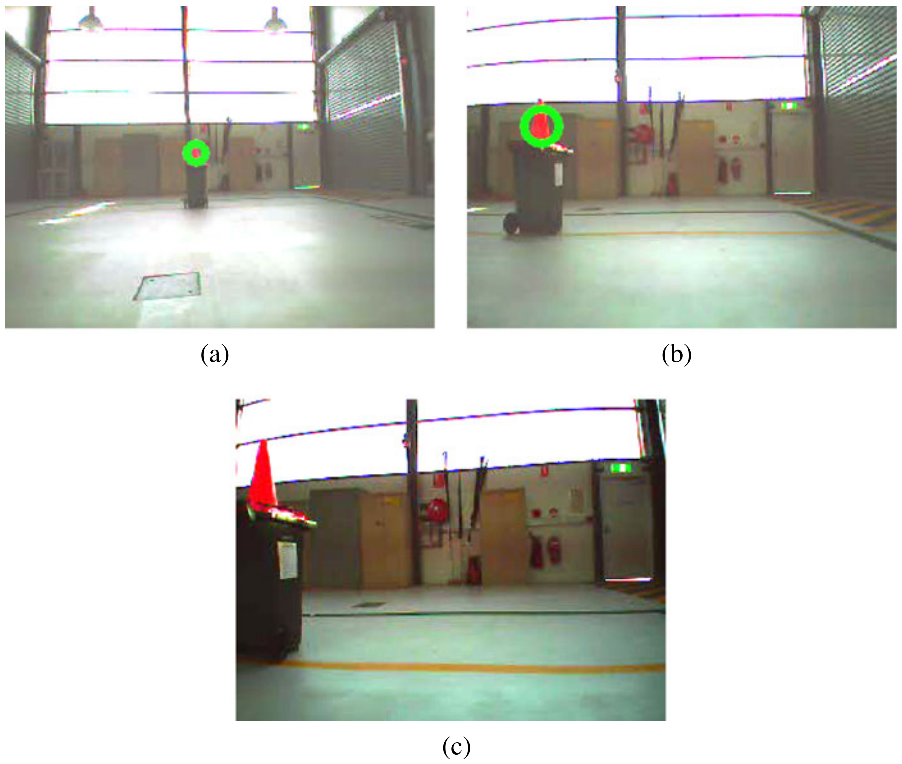

Fig. 17 Onboard images during the execution of the test

commands were sent during the experiments. The altitude was set to a constant value of 0.8 m and is was controlled by the internal altitude controller of the aircraft. The position of the quadcopter is calibrated at the beginning of the test, being the initial position the point (0, 0, 0) m in the VICON system. The obstacle to avoid was located in front of the initial position of the quadcopter at 4.5 m of distance and at 1.1 m from the floor (4.5, 1.1) m.

Once the quadrotor take-off, it flies 0.5 m towards the obstacle using the same controller with a reference value equal to 0 to keep the obstacle in the centre of the image. Once the aircraft flies the first half meter the reference for the control system it change to keep the obstacle in one of the edge of the image to try to avoid the obstacle. Until the aircraft does not reaches 3.5 in the forward direction it will continue trying to avoid the obstacle. Once this distance was reached by the aerial vehicle, a constant yaw (last yaw commanded) will be send. In that way we can compare how the optimization improve the behavior of the controller. Keeping the obstacle in one of the edge of the image tracking it with yaw commands imply lateral deviation of the trajectory of less than 2 m but keeping the same direction when the test finishes successfully. It must be take into account that the aim of this work is the optimization of the controller and not the way to starts and ends the avoiding obstacle task, as is shown in Fig. 16. The Fig. 17 shows some images captured from the onboard camera during the execution of one of these tests. The Fig. 17a shows the beginning of the test during the first 0.5 m keeping the obstacle in the center of the image. The Fig. 17b shows the capture image at the middle of the test and at the Fig. 17c can be seen when the quadrotor is overtaking the obstacle.

To compare the improvements of the optimization process we test both controllers at different speeds. Table 4 shown all the tests done. We test from 0.02 m/s speed until 1.4 m/s and avoiding the obstacle keeping it on the right side and on the left side. In this table is shown, also, the Root Mean Squared Error (RMSE) of each test. When no number is shown on this box it represents that the aircraft could not keep the obstacle to avoid on one edge of the image, losing it before the aircraft has covered the distance of 3 m. This kind of situations imply that the quadrotor change too much the trajectory or goes very close to the obstacle to avoid. Definitively the optimized controller has obtained better results. More tests have been finished successfully by this controller and

Table 4 Comparison between the non optimized and the Cross-Entropy optimized fuzzy controllers at different speeds

Type of controller	RMSE (degrees)	Speed (m/s)	Obstacle position
Non-optimized fuzzy controller	7.848	0.02	Left
CE-fuzzy controller	6.4048	0.02	Left
Non-optimized fuzzy controller	9.0081	0.04	Left
CE-fuzzy controller	5.2714	0.04	Left
Non-optimized fuzzy controller	–	0.06	Left
CE-fuzzy controller	7.4886	0.06	Left
Non-optimized fuzzy controller	–	0.08	Left
CE-fuzzy controller	9.8207	0.08	Left
Non-optimized fuzzy controller	–	0.1	Left
CE-fuzzy controller	11.3606	0.1	Left
Non-optimized fuzzy controller	–	0.12	Left
CE-fuzzy controller	9.4459	0.12	Left
Non-optimized fuzzy controller	–	0.14	Left
CE-fuzzy controller	–	0.14	Left
Non-optimized fuzzy controller	–	0.14	Right
CE-fuzzy controller	–	0.14	Right
Non-optimized fuzzy controller	–	0.12	Right
CE-fuzzy controller	10.3514	0.12	Right
Non-optimized fuzzy controller	–	0.1	Right
CE-fuzzy controller	11.4794	0.1	Right
Non-optimized fuzzy controller	–	0.08	Right
CE-fuzzy controller	10.5684	0.08	Right
Non-optimized fuzzy controller	–	0.06	Right
CE-fuzzy controller	8.1564	0.06	Right
Non-optimized fuzzy controller	12.7498	0.04	Right
CE-fuzzy controller	8.6037	0.04	Right
Non-optimized fuzzy controller	7.1514	0.02	Right
CE-fuzzy controller	6.3117	0.02	Right

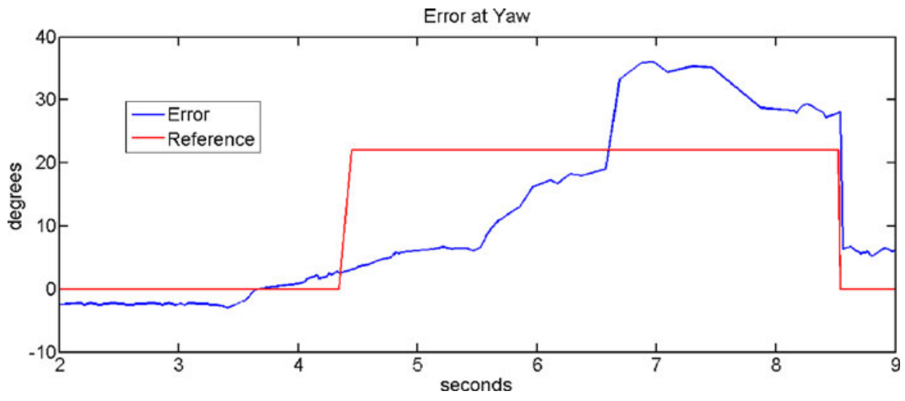

Fig. 18 Evolution of the error during a real test at 0.04 m/s forward speed using the non optimized fuzzy controller. A RMSE of 9.0081 has been obtained

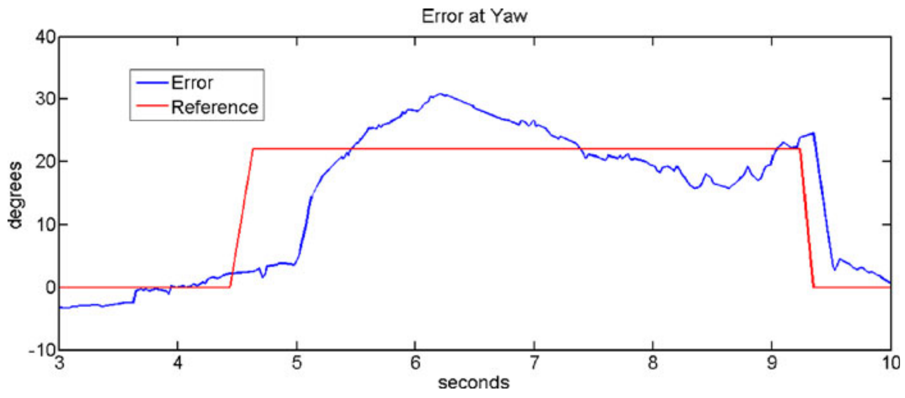

Fig. 19 Evolution of the error during a real test at 0.04 m/s forward speed using the fuzzy controller optimized using the Cross-Entropy method. A RMSE of 5.271 has been obtained, more than 40 % reduction

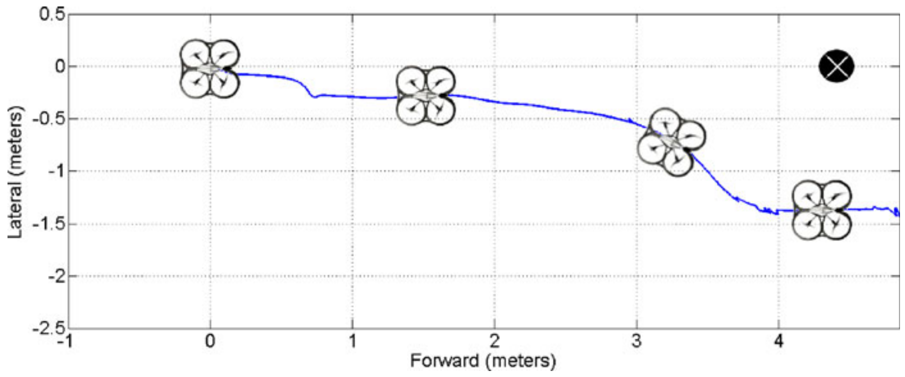

Fig. 20 2D reconstruction of the trajectory defined during a real test at 0.04 m/s forward speed using the non optimized fuzzy controller

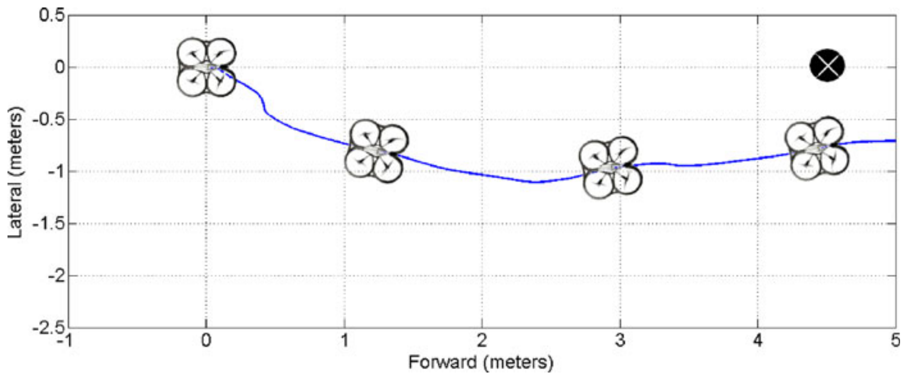

Fig. 21 2D reconstruction of the trajectory defined during a real test at 0.04 m/s forward speed using the fuzzy controller optimized using the Cross-Entropy method

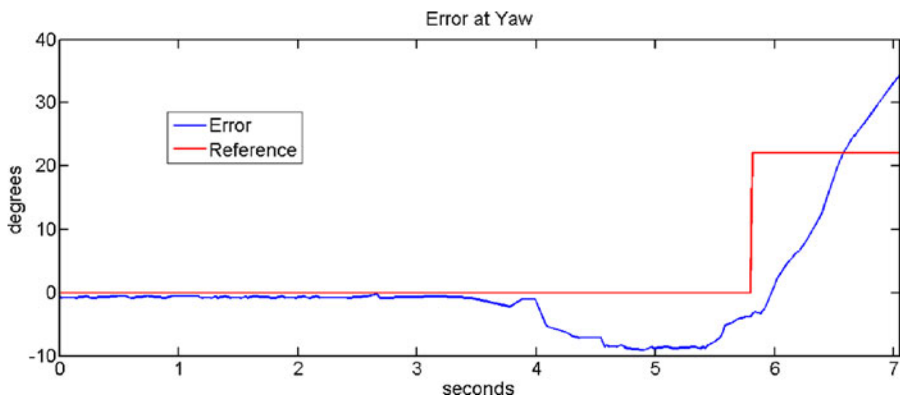

Fig. 22 Evolution of the error during a real test at 0.08 m/s forward speed using the non optimized fuzzy controller. A RMSE of 9.0081 has been obtained

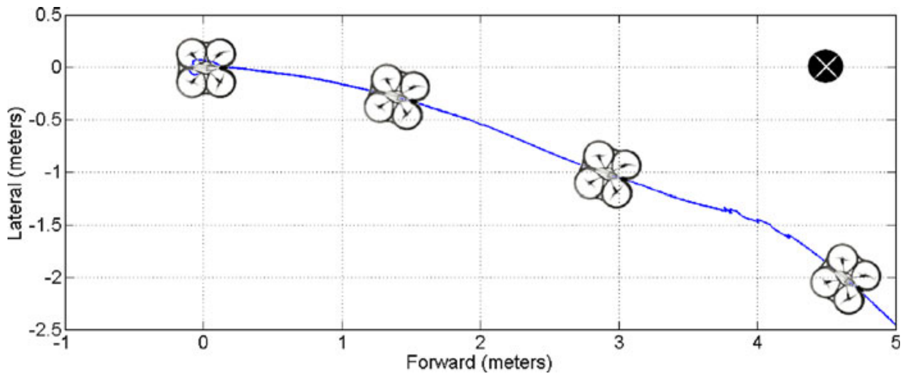

Fig. 23 2D reconstruction of the trajectory defined during a real test at 0.08 m/s forward speed using the non optimized fuzzy controller

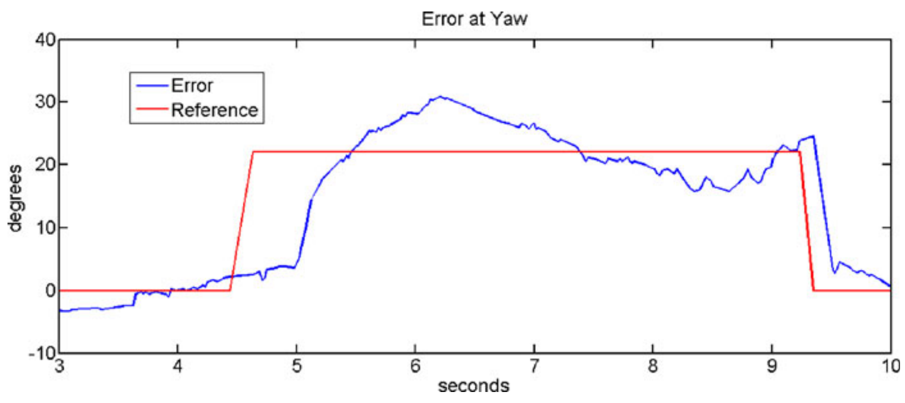

Fig. 24 Evolution of the error during a real test at 0.08 m/s forward speed using the fuzzy controller optimized using the Cross-Entropy method. A RMSE of 5.271 has been obtained, more than 40 % reduction

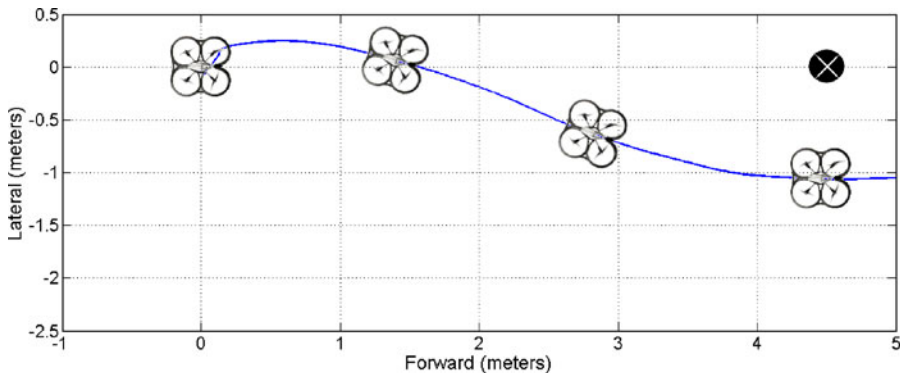

Fig. 25 2D reconstruction of the trajectory defined during a real test at 0.08 m/s forward speed using the fuzzy controller optimized using the Cross-Entropy method

the RMSE is lower when both controller finished the same test. In order to do a more reliable comparison no tests have been done at the same speed of the optimization process (0.03 m/s).

Following the most significant tests are presented. The first test shown is keeping the obstacle on the left edge of the image at 0.04 m/s. In this test both controllers past it successfully with a RMSE of 9.0081 for the non-optimized controller versus a RMSE of 5.271 of the optimized controller. These RMSE values represent a reduction of 41.5 %. Figure 18 shows the evolution of the error during this test for the non optimized controller. The Fig. 19 shows the same test for the optimized controller using the Cross-Entropy method. In both figures the red step represents the moment in which the avoiding task is done. Once the obstacle is out of the image no more error information has been obtained. The black circle with the white cross represents the position of the obstacle to avoid. In the first case the aircraft has to modify

A 2D reconstruction of the flight that the aircraft has done is presented in the next figures. For these figures we use the information obtained with the VICON, which has not used to control the vehicle for the avoiding obstacle task. Figure 20 shows the trajectory defined by the non optimized controller, and Fig. 21 shows the trajectory defined by the aircraft using the Cross-Entropy optimized controller. Comparing both figures is possible to appreciate that the non-optimized controller is slower than the optimized one, as is shown, also, in the error evolution figures.

Because of its slowness can happens to different situations. The aircraft will hit the obstacle to avoid or the trajectory change too much to the initial one. This last situation is the one what happens in the next test. In this case the speed doubles the one of the previous test. Figure 22 shows how the non optimized controller can not keep the obstacle to avoid on the edge of the image and loose it. This is appreciate, also, in Fig. 23, in which the trajectory defined by the aircraft is totally different to the previous test. The evolution of the error finish when the detected obstacle to avoid is out of the image. In less than 1.5 seconds the controller loose the obstacle. However the optimized controller could finish this test successfully. Figure 24 shows the evolution of the error, and Fig. 25 shows the movements of the aircraft to avoid the obstacle.

A video and more information of these and other tests can be found at [40, 41].

8 Conclusions

This work presents a macroscopic optimization of a PID Fuzzy controller. The optimization of the scaling factors of the controller were done using the novel optimization method named Cross-Entropy. The optimization is focused on improving a controller that commands an aerial vehicle to avoid possible obstacles in its trajectory. This optimization method has few uses in the literature for control, but never has been used in such a dynamic environment like the one presented in this work. The optimization process was done with the simulator environment of ROS-Gazebo at fixed vehicle speed. A software implementation of the Cross-Entropy method and different programs to inter-actuate with the simulator have been done. The optimization method just needs 11 iterations to obtained good enough results to use them in a real environment. The low number of controllers tested (just 330) remarks the power of this optimization method. A big amount of tests at different speed were done to determine the improvement of the controller. The optimized controller could successfully finish the avoiding tasks in more situations than the non optimized controller. The use of the Cross-Entropy method to optimize the controller allows tripling the speed of the non optimized controller. The faster test of the optimized controller was 4 times faster than the speed used to train it.

After the successful results obtained a comparison with other optimization method have be done. An extension to the two other scale magnitude of optimization of Fuzzy control will be done, in order to modify the membership functions and the base of rules.

Acknowledgements The work reported in this paper is the consecution of several research stages at the Computer Vision Group - Universidad Politécnica de Madrid and at the Australian Reasearch Centre for Aerospace Automation - Queensland University of Technology (ARCAA-

QUT). The authors would like to thank to Caja Madrid for the mobility research grant of one of the authors. This work has been sponsored by the Spanish Science and Technology Ministry under the grant CICYT DPI2010-20751-C02-01 and by the IRSES Program Marie Curie FP7 - PIRSES-GA-2009-230797 - ICPUAS (International Cooperation Program for Unmanned Aerial Systems Research and Development).

References

1. Zadeh, L.A.: Fuzzy logic and soft computing: issues, contentions and perspectives. In: Proc. IIZUKA94: 3rd Int. Conf. Fuzzy Logic, Neural Nets and Soft Computing, pp. 1–2 (1994)
2. United States Department of Defense: [Online]. Available: http://www.defense.gov (2010). Accessed 1 May 2010
3. Meister, O., Frietsch, N., Ascher, C., Trommer, G.: Adaptive path planning for a VTOL-UAV. In: Position, Location and Navigation Symposium, 2008 IEEE/ION, pp. 1252–1259 (2008)
4. Moses, A., Rutherford, M., Valavanis, K.: Radar-based detection and identification for miniature air vehicles. In: 2011 IEEE International Conference on Control Applications (CCA), pp. 933–940 (2011)
5. Hibrid System Laboratory, Berkeley University, Quadrotor and Kinect: [Online]. Available: http://hybrid.eecs.berkeley.edu/ (2012). Accessed 22 January 2012
6. Lai, J.S., Mejias, L., Ford, J.J.: Airborne vision-based collision-detection system. J. Field Robot. **28**(2), 137–157 (2011). [Online]. Available: http://eprints.qut.edu.au/32801/. Accessed 4 August 2010
7. Mejias, L., Mondragon, I., Campoy, P.: Omnidirectional bearing-only see-and-avoid for small aerial robots. In: The 5th International Conference on Automation, Robotics and Applications. IEEE, Wellington (2011). [Online]. Available: http://eprints.qut.edu.au/46597/. Accessed 17 October 2011
8. Mejias, L., McNamara, S., Lai, J.S., Ford, J.J.: Vision-based detection and tracking of aerial targets for UAV collision avoidance. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, Taipei (2010). [Online]. Available: http://eprints.qut.edu.au/32793/. Accessed 11 December 2010
9. Shabayek, A.E.R., Demonceaux, C., Morel, O., Fofi, D.: Vision based UAV attitude estimation: progress and insights. J. Intell. Robot. Syst. **65**(1–4), 295–308 (2012)
10. Martinez, C., Mondragon, I., Olivares-Mendez, M., Campoy, P.: On-board and ground visual pose estimation techniques for UAV control. J. Intell. Robot. Syst. **61**, 301–320 (2011). doi:10.1007/s10846-010-9505-9
11. Dusha, D., Mejias, L.: Error analysis and attitude observability of a monocular GPS/visual odometry integrated navigation filter. Int. J. Rob. Res. **31**(6), 714–737 (2012). [Online]. Available: http://eprints.qut.edu.au/46549/
12. Hunt, K., Sbarbaro, D., Zbikowski, R., Gawthrop, P.: Neural networks for control systems: a survey. Automatica **28**(6), 1083–1112 (1992). [Online]. Available: http://www.sciencedirect.com/science/article/pii/000510989290053I
13. Precup, R.-E., Hellendoorn, H.: A survey on industrial applications of fuzzy control. Comput. Ind. **62**(3), 213–226 (2011). [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0166361510001363
14. Zheng, L.: A practical guide to tune of proportional and integral (PI) like fuzzy controllers. In: IEEE International Conference on Fuzzy Systems, pp. 633–640 (1992)
15. Malhotra, R., Singh, N., Singh, Y.: Soft computing techniques for process control applications. Int. J. Soft Comput. **2**, 32–44 (2011)
16. Bonissone, P., Khedkar, P., Chen, Y.: Genetic algorithms for automated tuning of fuzzy controllers: a transportation application. In: Proceedings of the Fifth IEEE International Conference on Fuzzy Systems, vol. 1, pp. 674–680 (1996)
17. Li, W.: Optimization of a fuzzy controller using neural network. In: Proceedings of the Third IEEE Conference on Fuzzy Systems. IEEE World Congress on Computational Intelligence, vol. 1, pp. 223–227 (1994)
18. Jang, J.-S.R.: ANFIS: adaptive-network-based fuzzy inference system. IEEE Trans. Syst. Man Cybern. **23**, 665–685 (1993)
19. Werbos, P.: Beyond regression: new tools for prediction and analysis in the behavioral sciences. Ph.D. dissertation, Harvard University, Cambridge, MA (1974)
20. Bonissone, P.P., Chen, Y.-T., Goebel, K., Khedkar, P.S.: Hybrid soft computing systems: industrial and commercial applications. In Proceedings of the IEEE (1999)
21. Rubinstein, R.Y.: Optimization of computer simulation models with rare events. Eur. J. Oper. Res. **99**, 89–112 (1996)
22. Belmudes, F., Ernst, D., Wehenkel, L.: Cross-entropy based rare-event simulation for the identification of dangerous events in power systems. In: Proceedings of the 10th International Conference on Probabilistic Methods Applied to Power Systems. PMAPS ’08, pp. 1–7 (2008)
23. Celeste, F., Dambreville, F., Le Cadre, J.-P.: Optimal path planning using cross-entropy method. In: 2006 9th International Conference on Information Fusion, pp. 1–8 (2006)
24. Zhang, Y., Ji, C., Malik, W., O’Brien, D., Edwards, D.: Cross-entropy optimisation of multiple-input multiple-output capacity by transmit antenna selection. IET Microwaves Antennas Propag. **1**(6), 1131–1136 (2007)
25. Kobilarov, M.: Cross-entropy randomized motion planning. In: Proceedings of Robotics: Science and Systems, Los Angeles, CA, USA (2011)
26. Bodur, M.: An adaptive cross-entropytuning of the PID control for robot manipulators. In: Proceedings of the World Congress on Engineering, WCE 2007, pp. 93–98 (2007)

27. Haber, R.E., del Toro, R.M., Gajate, A.: Optimal fuzzy control system using the cross-entropy method. A case study of a drilling process. Inf. Sci. **180**, 2777–2792 (2010)
28. Bradski, G.: Computer vision face tracking for use in a perceptual user interface. In: IEEE Workshop on Applications of Computer Vision, Princeton, NJ, pp. 214–219 (1998)
29. Fukunaga, K., Hostetler, L.: The estimation of the gradient of a density function, with applications in pattern recognition. IEEE Trans. Inf. Theory **21**(1), 32–40 (1975)
30. Mondragón, I., Olivares-Méndez, M., Campoy, P. Martínez, C., Mejias, L.: Unmanned aerial vehicles uavs attitude, height, motion estimation and control using visual systems. Auton. Robots **29**, 17–34 (2010). doi:10.1007/s10514-010-9183-2
31. Olivares-Mendez, M., Mondragon, I., Campoy, P., Martinez, C.: Fuzzy controller for UAV-landing task using 3d-position visual estimation. In: 2010 IEEE International Conference on Fuzzy Systems (FUZZ), pp. 1–8 (2010)
32. Olivares-Mendez, M., Mellado, I., Campoy, P., Mondragon, I., Martinez, C.: A visual AGV-urban car using fuzzy control. In: 2011 IEEE International Conference on Automation, Robotics and Applicatios (ICARA) (2011)
33. Rubinstein, R.Y., Kroese, D.P.: The Cross-Entropy Method: A Unified Approach To Combinatorial Optimization, Monte-Carlo Simulation (Information Science and Statistics). Springer, Secaucus (2004)
34. Botev, Z., Kroese, D.P.: Global likelihood optimization via the cross-entropy method with an application to mixture models. In: Proceedings of the 36th Conference on Winter Simulation, pp. 529–535 (2004)
35. Robot Operating System (ROS): [Online]. Available: http://ros.org/wiki/gazebo (2012)
36. Starmac-ROS package. Hybrid Systems Laboratory, UC Berkeley (2012). [Online]. Available: http://www.ros.org/wiki/starmac-ros-pkg
37. Ar.Drone Parrot (2010). [Online]. Available: http://ardrone.parrot.com
38. Computer Vision Group-UPM (2012). [Online]. Available: http://www.vision4uav.eu
39. Motion Capture System from VICON (2012). [Online]. Available: http://www.vicon.com
40. Computer Vision Group-UPM. See and avoid fuzzy controller scaling factors optimized using cross-entropy (2012). [Online]. Available: http://vision4uav.eu/?q=researchline/SeeAndAvoidCE_SF
41. Youtube Channel Computer Vision Group-UPM (2012). [Online]. Available: http://www.youtube.com/colibriprojectUAV