UAS See-and-Avoid using two different approaches of Fuzzy Control

Miguel A. Olivares-Mendez[1] and Luis Mejias[2]
Pascual Campoy[1] and Ignacio Mellado-Bataller[1]
and Ivan Mondragon[1]

Abstract— **This work presents two UAS See and Avoid approaches using Fuzzy Control. We compare the performance of each controller when a Cross-Entropy method is applied to optimase the parameters for one of the controllers. Each controller receive information from an image processing front-end that detect and track targets in the environment. Visual information is then used under a visual servoing approach to perform autonomous avoidance. Experimental flight trials using a small quadrotor were performed to validate and compare the behaviour of both controllers.**

I. Introduction

Fuzzy control techniques are demostrasting to be succesful for controlling the dynamics of UVS. This paper explodes the utilization of Fuzzy techniques that weight the classical terms of PD and PID controllers. The optimization of those terms used by the Fuzzy controller is addressed by using the innovative Cross Entropy method.

Advances in electronics have allowed the miniaturization of sensors and avionics onboard aircraft, causing a radical increasing in manufacturing of all sort of aerial robots at an affordable price. Nowadays is possible to acquire a basic rotary-wing UAS in common electronic or toy shop [1]. As a consequence, aerial robotics is gaining considerable predominance among researchers nowadays. Industry, academia and general public are placing more attention in Unmanned Aerial Systems (UAS) to understand the potential benefits UAS could provide to society.

But before UASs are allowed to routinely fly in civil airpspace, several technological hurdles need to be addressed. For example, collision avoidance or safe termination systems are some of the technologies UAS are requires before the share the airspace and fly over populated areas [2]. The capability to gain onboard situational awareness is achieved by the use of multiple onboard sensors. For instance, Laser ranger finder [3], sonars [4] or cameras. The most common and computational efficient sensors are cameras. An example of the use of cameras for attitude estimation can be found in [5], [6]. Furthermore, advanced systems such as the ”kinect” [7] have been demonstrated with quadrotors [8].

By combining the information from these sensors with the onboard guidance and control is how an UAS can perform its ultimate mission. Onboard guidance and control can be implemented by a number of classical or modern control theories. However, techniques that are demonstrating to be simple yet efficient are Soft-Computing. They are becoming of common use in engineering application such as prediction [9], data mining [10] and control.

The way Soft-Computing can manage uncertainty and inaccuracies of sensors made them very suitable for automatize robotic systems. Some examples for control purposes can be found in [11], [12].

This paper explores the utilization of a class of Soft-computing techniques called ”Fuzzy” that weight the classical terms of PD and PID controllers. The optimization of those terms used by the Fuzzy controller is addressed by using Cross Entropy techniques.

In order to obtain a robust control system once the controller was developed an optimization process is required. One of the recent optimization method developed is the Cross-entropy [13] which has not been widely used for control tasks [14], [15].

This paper is structured as follows. In section II we describe the image processing front-end used in our approaches. In section III we explain both visual servoing approaches using fuzzy logic for heading control. The cross-entropy theory is introduced in section IV. Experimental results of both controllers are presented in section V. Finally, concluding remarks and future work are presented in section VI.

II. Image processing front-end

Visual awareness is achieved by using an onboard forward-looking camera. Images from the camera then sent for off-board processing in a laptop ground-station. The outcome of the visual processing (and servoing commands) are then send 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 of left (as seen from image centre). When the object is first detected is pushed to the edge of the image (far left of right side), and kept a fixed position that represent a constant relative bearing.

The target is detected by pre-defining a color and then designing an algorithm to highlight this color that then will be tracked along the image sequence. The tracking is performed by using the Continuously Adaptive Mean Shift [16] (CamShift). This algorithm is based on the mean shift originally introduced by Fukunaga and Hostetler [17]. This algorithm accounts for the dynamic nature of changes in

[1]Universidad Politécnica de Madrid, CAR - Centro de Automática y Robótica Email: miguelangel.olivares@upm.es, http://www.vision4uav.eu/?q=miguel/personal [2]Australian Research Centre for Aerospace Automation (ARCAA) Queensland University of Technology GPO Box 2434, Brisbane, Queensland 4001, Australia Email: luis.mejias@qut.edu.au

lighting conditions by dynamically adapting to changes in probability distributions of color.

Using 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 coloured object. Using the location of 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.

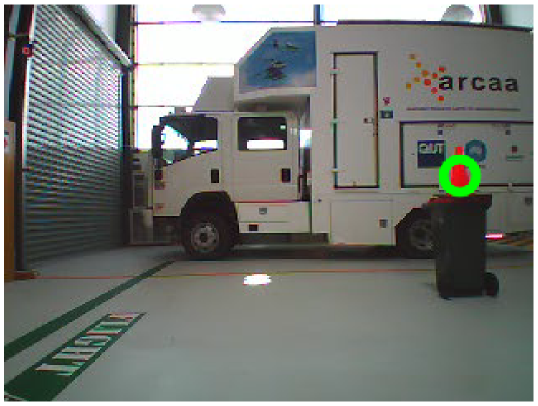


Fig. 1. Control loop with the optimization of the Cross-Entropy method.

III. Fuzzy Controller

The aim of both controllers is to generate desired yaw commands for the vehicle based on the location of the target in the image plane. The control task is based on Fuzzy Logic techniques. Both controllers have been implemented using our own library MOFS (Miguel Olivares' Fuzzy Software). This library has an hierchical 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 task of develop Fuzzy Logic Controllers. These routines have been used in a wide variety of control applications such as autonomous landing [18] and autonomous road following [**?**]. A deeply explanation of this software could be found at [19].

In both cases the inputs and the outputs have been defined using triangular membership functions. The product t-norm is used for the conjuntion of the rules and the Height Weight method has been selected for the defuzzification phase (Equation 1).

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

In the next subsections the two controllers are presented.

A. *First Approach: PD-Fuzzy Controller*

The first controller developed is a Fuzzy Logic system with two inputs and one output. The first input is the angle between the quadcopter, the object to avoid and the right or the left side of the image, as shown in Figure2. The second input is the measure of the evolution of this angle between the last two frames, as is shown in Figure3. The output of the controller is the desired yaw angle that the quadcopter need to turn to keep the object at the desired position, see Figure 4.

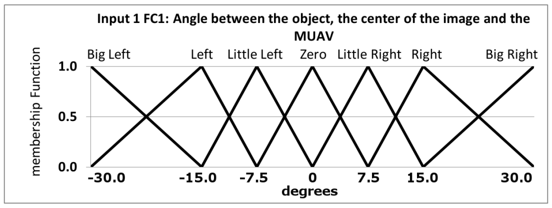


Fig. 2. PD-Fuzzy Controller: Membership function of the first input, the error.

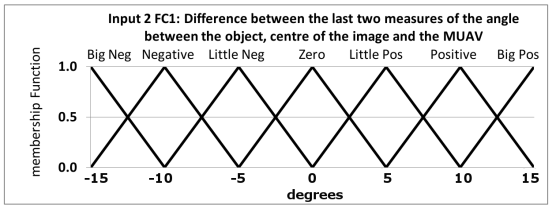


Fig. 3. PD-Fuzzy Controller: Membership function of the second input, the derivate of the error.

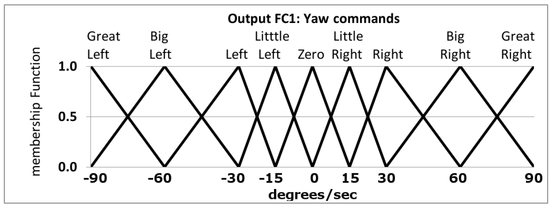


Fig. 4. PD-Fuzzy Controller: Membership function of the output, heading degrees to turn.

This controller has 45 rules that have been set using heuristics methods. In the Figure 5 is presented a 3D-surface reconstruction of the base of rules.

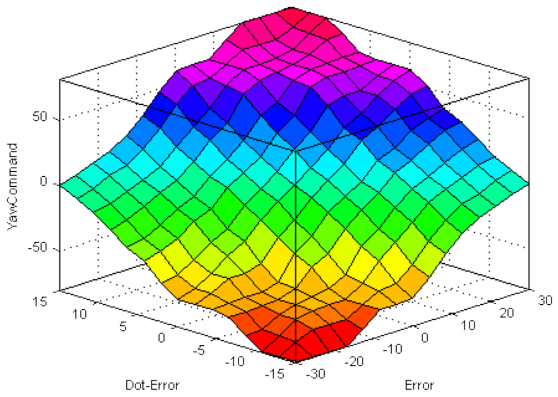


Fig. 5. 3D surface representation of the PD-Fuzzy controller rules base.

This fuzzy controller has been used for object following task with an excellent results [20].

B. Second Approach: PID-Fuzzy Controller optimized using Cross-Entropy

The second fuzzy controller has been defined using three inputs and one output. The first input measure the error in degrees between the quadrotor, the object to avoid minus the reference (Figure6). The second, is the derivate of the error, as is shown in Figure7, and third input, shown in the Figure 8 represents the integral of the error. The output is the commanded yaw that the vehicle needs to turn in order to keep the object at the desired relative bearing, see Figure 9. First and second outputs are equivalent to the inputs of the first approach.

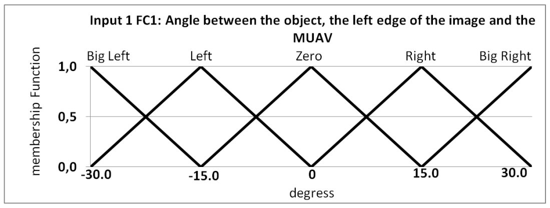


Fig. 6. PID-Fuzzy Controller: Membership function of the first input, the error.

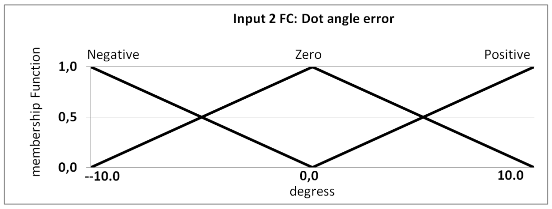


Fig. 7. PID-Fuzzy Controller: Membership function of the second input, the derivate of the error.

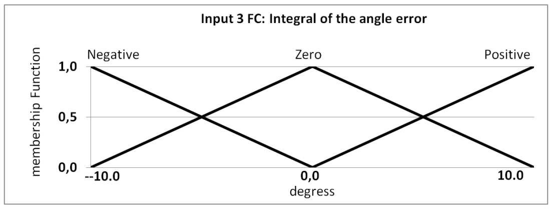


Fig. 8. PID-Fuzzy Controller: Membership function of the third input, the integral of the error.

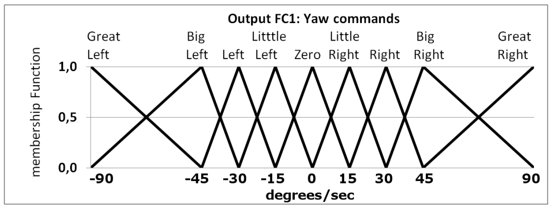
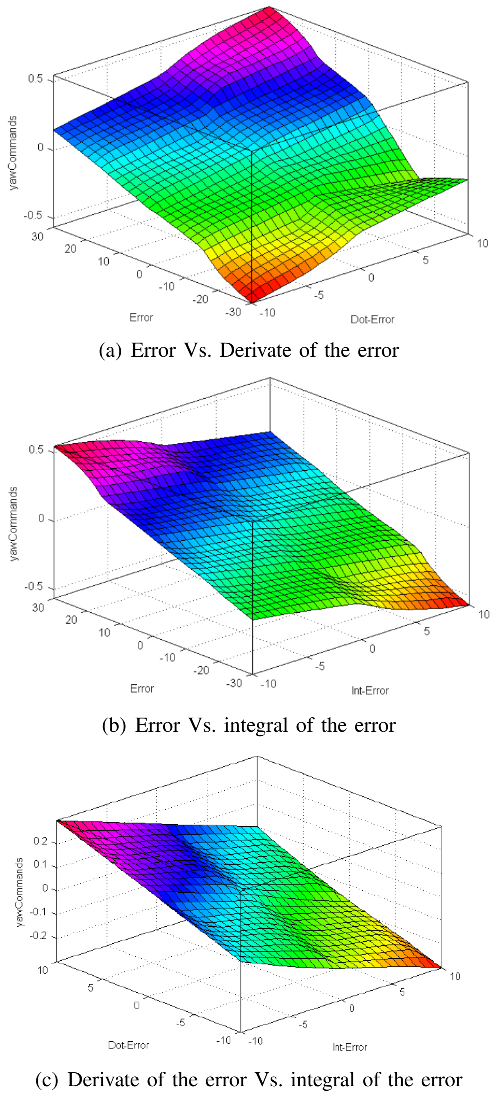


Fig. 9. PID-Fuzzy Controller: Membership function of the output, heading degrees to turn.

The definition of the fuzzy variables uses 45 rules. A 3D representation of this base of rules is shown in the Figure 10. This plot shows the one-to-one relationship of variables (horizontal plane), being the third dimension (vertical axis) the output. In where Figure 10(a) shows the relation between the first and the second inputs, Figure 10(b) shows the relation between the first and the third inputs, and Figure 10(c) presents the relation between the second and the third inputs.



(a) Error Vs. Derivate of the error

(b) Error Vs. integral of the error

(c) Derivate of the error Vs. integral of the error

Fig. 10. 3D surface representations of the PID-Fuzzy controller rules base.

Comparing with previous visual servoing works with aerial vehicles, in which no optimization have been applied, this approach shows a big reduction in the number of the membership function. A simplify base of rules has been obtained thanks to the cross-entropy tunning of the controller.

IV. 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.

In this paper 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 deeply explanation of the Cross Entropy method is presented on [13]

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

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

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

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

where E_v denotes the corresponding expectation operator. In this manner, Equation 3 transform 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 id the Kullback-Leibler, also known like 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 re-written 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 $(v_t \mid t \geq 1)$.

B. Fuzzy Control Optimization Approach

This approach is based on a population-and-simulation optimization [21]. 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 (Figures 6, 7 and 8). 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: $(x_j(t))_{1\leq j\leq N})$ from $g(x,v(t))$, being each $x_i = (x_{KEj}, x_{KDj}, x_{KIj})$
3. Compute $\phi(x_j(t))$ and order $\phi_1, \phi_2, ..., \phi_N$ from smallest to biggest.Get the $N^{elite}$ first controllers $\gamma(t) = \chi_{[N^{elite}]}$.
4. Update $v(t)$ with $v(t+1) = arg_v min \frac{1}{N} \sum_{j=1}^{N} I_{\{\chi(x_j(t))\geq\gamma(t)\}} \cdot ln\, g(x_j(t), v(t))$
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 $\tilde{\mu}_{th} = \sum_{j=1}^{N^{elite}} \frac{x_{jh}}{N^{elite}}$ and $\tilde{\sigma}_{th} = \sum_{j=1}^{N^{elite}} \frac{(x_{jh}-\mu_{jh})^2}{N^{elite}}$ where $4 \leq N^{elite} \leq 20$.

The mean vector $\bar{\tilde{\mu}}$ should converge yo 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 $0s$ and $1s$ in the parameter vectors, and β is a constant value which modify the value of η.

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

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 $0.4 \leq \alpha \leq 0.9$, $0.6 \leq \beta \leq 0.9$ and $2 \leq q \leq 7$. In order to get an optimized controller different cost functions 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).

V. Results

in order to validate the behaviour of both controllers we conducted real flights tests. We used a AR.Drone-Parrot [1] platform with our own software routines developed for this purpose [22]. 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 [23]. The VICON system was used for data logging, no data was used for the control of the quadrotor.

A. Quadcopter System

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 wi-fi connection. A extended explanation of this platform is presented at [1].

Fig. 11. Parrot-AR.Drone, the platform used for the real tests.

For both controllers the flight tests were performed with constant forward speed (constant pitch angle). No roll commands were sent during the experiments. The altitude was set to a constant value of $0.8m$ 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 $(0,0,0)$ meters. The obstacle to avoid was located in front of the initial position of the quadcopter at 6 meters of distance and at 1.1 meters from the floor $(6,0,1.1)$ meters.

B. First Approach: PD-Fuzzy Controller

We present next one test of the controller developed using the approach. The control loop diagram designed for this controller is shown in the Figure 12, where the two inputs of the controller are seen. In this case the constants K_e and K_d have not been optimized with the Cross-Entropy method and are equal to 1.

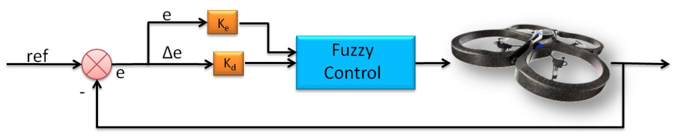


Fig. 12. Schematic diagram of the control Loop.

The avoiding obstacle task has been accomplish with successful results as is shown in the 3D flight reconstruction shown in Figure 13.

The Figure 14 shows a 2D version of this test, in which is marked the object to avoid as a black circle with a white cross.

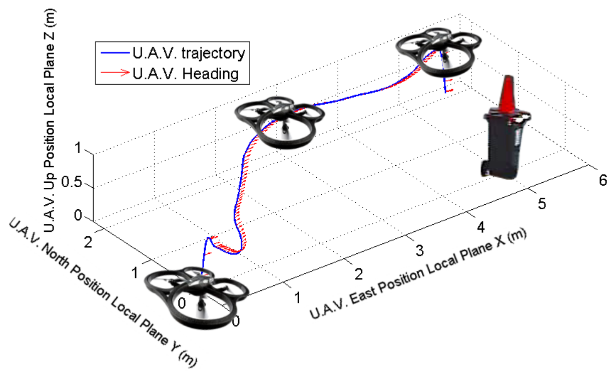


Fig. 13. 3D flight reconstruction of the flight path. The obstacle to avoid is a orange traffic cone located at the position (5,0,1.1).

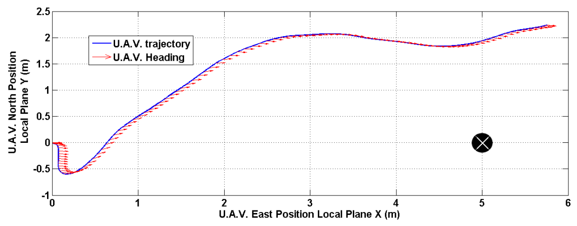


Fig. 14. 2D flight reconstruction with the VICON data. The black circle and the white cross at the position (6,0) represent the object to avoid.

Once the quadrotor takes-off, it flies for 0.2 meter towards the obstacle in open loop. Then the control process is activated, and then during the next following 5 seconds the controller sends commands to the aircraft. The image processing and control task are finished after the quadrotor reaches its minimum allowable yaw angle. After this point the aircraft will go forward without any yaw commands. The Figure 15 shows some images captured from the onboard camera during the execution of this test. The Figure 15(a) shows when the motors have not been ignited. The Figure 15(b) shows the beginning of the test during the first part in open loop. The Figures 15(c) and 15(d) shows two frames during the control process, and the Figure 15(e) shows when the quadrotor is overtaking the obstacle. A full video of the test can be found at [22] and [24] .

The behavior of the controller is represented in the Figure 16 which shows the evolution of the error during the test. The red line step represents the moment in which the image processing start. The measure of the step is 25 degrees, but at the moment when the step is applied the aircraft was looking at the opposite side increasing the step command to 35 degrees. To evaluate the behavior of the controller we use the error estimator of the root mean-square error (RMSE). The lower value this error estimator of $RMSE =$ 9.57 degrees. The quick response of the controller shown in this Figure corroborate the excellent behavior of the optimized-controller.

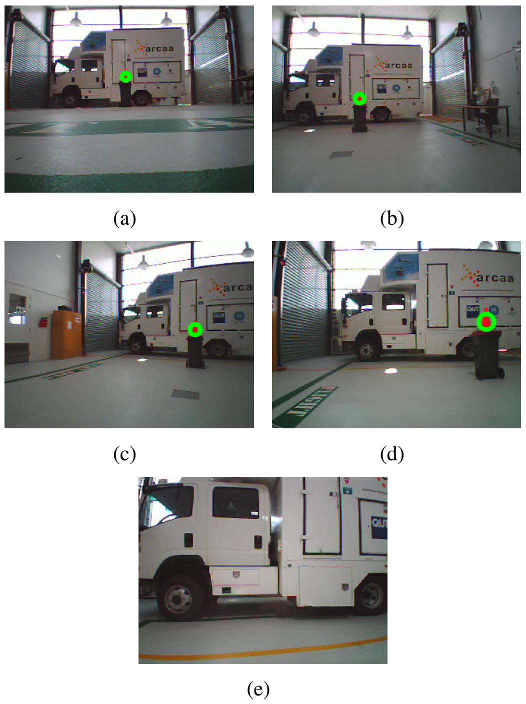


Fig. 15. Onboard images taken during the execution of the test. Figures 15(a) and 15(b) are previous to the control activation. Figures 15(c) and 15(d) belongs to the control process and Figure 15(e) is when the obstacle has been overtaken.

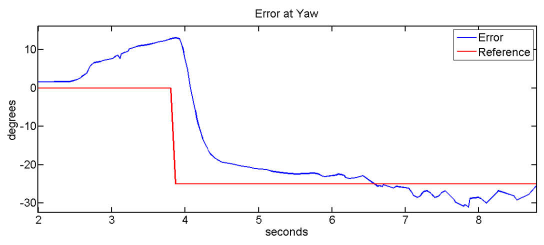


Fig. 16. Evolution of the error during a real test.

C. Second Approach: PID-Fuzzy controller optimized using Cross-Entropy

In this subsection are presented the simulation enviroment, the optimization of the Cross-Entropy method, the simulated tests and the real test.

1) Simulation Environment: The simulation tests were performed using the ROS (Robotics Operative System) and the 3D simulation Gazebo [25]. In the simulations, a quad-copter model of the starmack ros-pkg developed by Berkeley University [26] was used. The obstacle to avoid is defined by a virtual yellow balloon.

Two external software routines in $C++$ were developed for accomplish these tests. One using the cross-entropy method. This program is responsible for the optimization process. It generates a set of controllers, select the controller to test and when all the controllers are tested, update the pdf with the tests results to obtain the new set of controllers. The other one is the 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 stage for all the tests. The Figure 17 shows the tests flowchart.

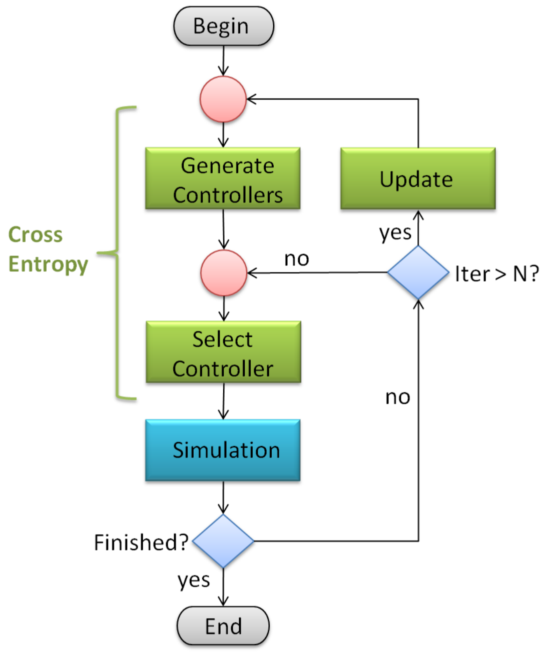


Fig. 17. Flowchart of the optimization process.

Besides this external software to execute in a loop the 3D simulator, two nodes have been added to the ROS-Gazebo. One is the visual algorithm which gets the visual image obtained by the simulated camera onboard the quadcopter. Each frame is converted in to an OpenCV image to be processed. Then the visual information 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.

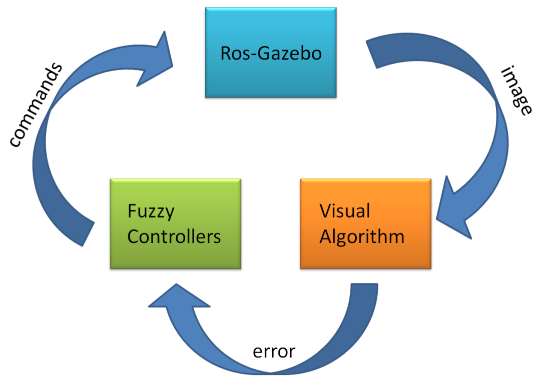


Fig. 18. Interaction between the ROS-Gazebo 3D simulator and the two other process developed for this work.

2) Optimization process using the simulation environment: In order to obtain the optimal controller, we define a fixed time for each simulation cycle. The quadcopter start positioned in front of the object to avoid. Each test is performed sending a constant pitch command to the aircraft.

To evaluate each test the Integral Time Absolute Error (ITAE) function cost was used. Some tests were made with the Root Mean Square Error (RMSE) cost function with similar results. We choice the ITAE error estimator is motivated by the error penalisation it imposes at the end of the test. Being more important estimator during a optimization process. The cross-entropy system generate $N = 30$ controllers per iteration based on the last update of the probabilistic density function for each gains. From this set of controllers the five with the lower ITAE value are selected ($Nelite = 5$) to update the next pdf parameters. xThe 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 $q = 2$, $\beta t_0 = 0$, $\beta_0 = 0.92$, $\alpha_0 = 0$. Those values are based on values reported in [14] and [21].

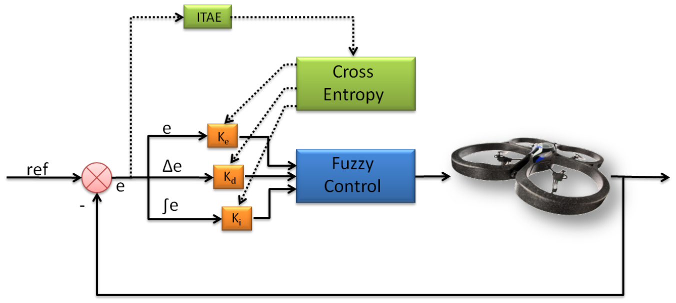


Fig. 19. Control loop with the optimization of the Cross-Entropy method.

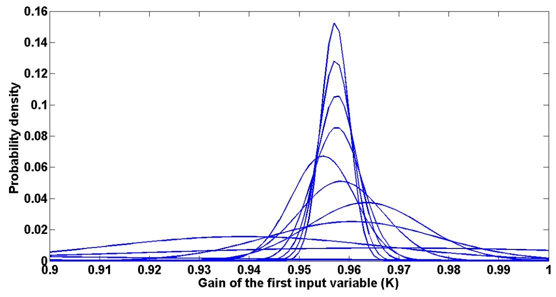


Fig. 20. Evolution of the probability density function for the first input gain. The standard variance converge in 12 iterations to a value of 0.0028 so that the obtained mean 0.9572 can be used in the real tests.

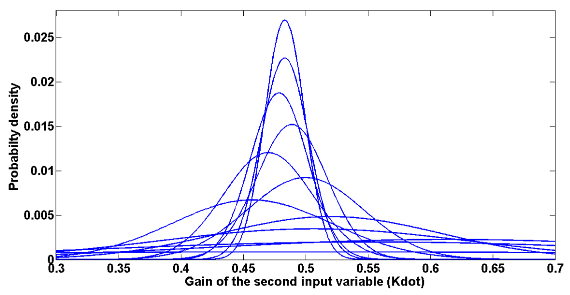


Fig. 21. Evolution of the probability density function for the second input gain. The standard variance converge in 12 iterations to a value of 0.0159 so that the obtained mean 0.4832 can be used in the real tests.

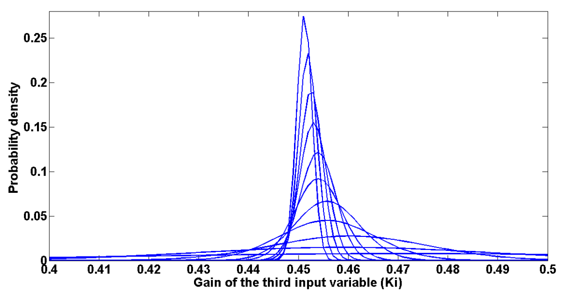


Fig. 22. Evolution of the probability density function for the third input gain. The standard variance converge in 12 iterations to a value of 0.0015 so that the obtained mean 0.4512 can be used in the real tests.

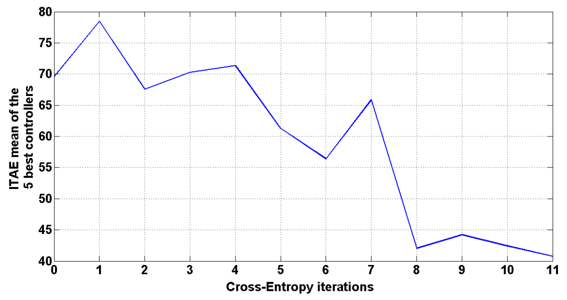


Fig. 23. Evolution of the itae error during the 12 Cross-Entropy iterations. The ITAE value of each iteration correspond to the mean of the first 5 of 30 controllers of each iteration.

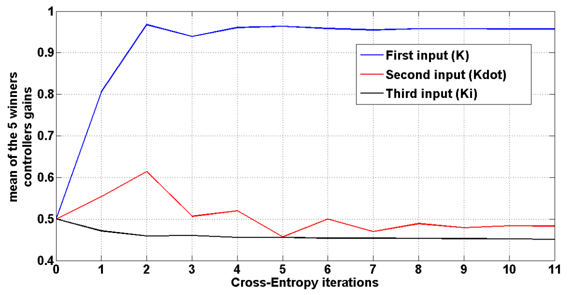


Fig. 24. Evolution of the gains of each input. The value of the gain correspond to the first 5 of 30 controllers of each iterations.

A number 330 tests were perfomed to obtain the optimal controller. This process corresponds to 11 updates of the pdf for the gains. Figure 20 shows the evolution of the probability density function of the first input of the controller. The final values of the pdf were $mean = 0.9572$ and $sigma = 0.0028$. Figure 21 shows the evolution for the second input with the final values of $mean = 0.4832$ and $sigma = 0.0159$. In the same way Figure 22 shows the evolution of the pdf for the third input, which finalize with $mean = 0.4512$ and $sigma = 0.0015$. Figure 23 presents the evolution of the mean of the ITAE value of the 5 winners from each set of 30 controllers. The Figure 24 shows the evolution of the different gains of the controller during the 330 tests.

3) Flight Test: The test was performed in similar fashion as the simulation and the previous tests done with the PD-Fuzzy controller. Figure 25 shows the control loop of the system without the Cross-Entropy optimization process.

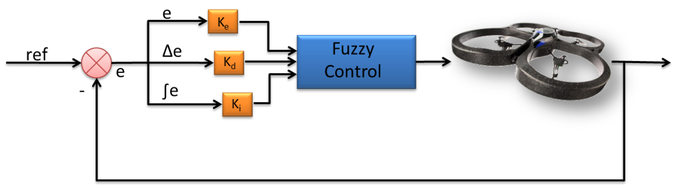


Fig. 25. Control loop with the optimization of the Cross-Entropy method.

Figure 26 shows the 2D reconstruction of one of the tests done and Figure 27 shows the 3D flight reconstruction over a captured frame from the camera used to record the test. This video can by found at [27].

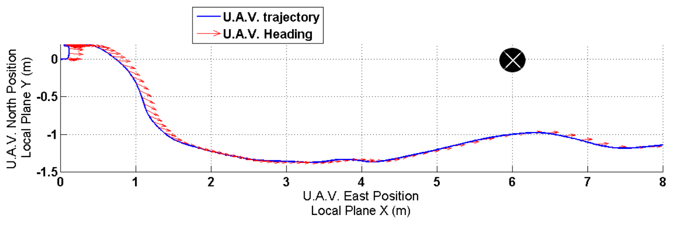


Fig. 26. 2D flight reconstruction with the VICON data. The black circle and the white cross at the position (6,0) represent the object to avoid.

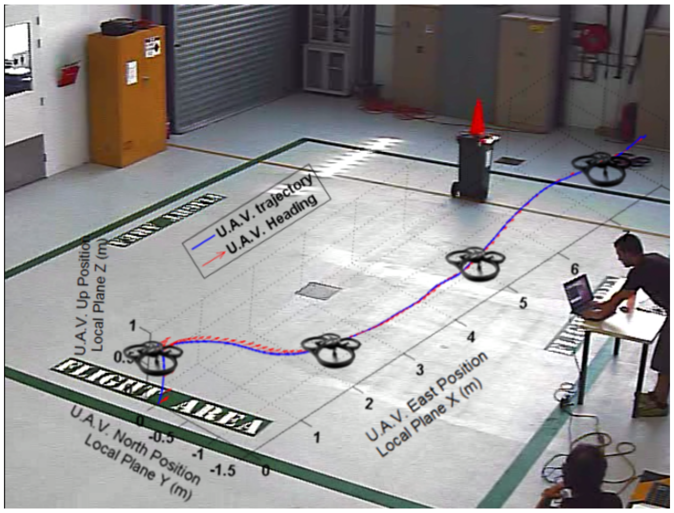

U.A.V. trajectory
U.A.V. Heading
U.A.V. Up Position Local Plane Z (m)
U.A.V. East Position Local Plane X (m)
U.A.V. North Position Local Plane Y (m)
FLIGHT AREA

Fig. 27. 3D flight reconstruction with the VICON data over a image capture with an external camera. The obstacle to avoid is a orange traffic cone and it is set at the position (6,0,1.1).

Once the quadrotor take-off, it flies one meter towards the obstacle in open loop. Then the visual control process is activated. During the next 5 seconds the controller sends commands to the aircraft. Once the aircraft has reached a maximum allowed turn, is commanded from this point with a constant yaw (last yaw commanded). The Figure 28 shows some images captured from the onboard camera during the execution of this test. The Figure 28(a) shows the beginning of the test during the first meter in open loop. The Figure 28(b) shows the capture image at the middle of the test and at the Figure 28(c) can be seen when the quadrotor is overtaking the obstacle.

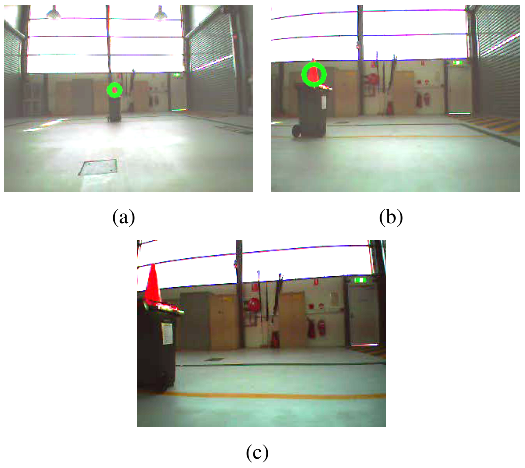


Fig. 28. Onboard images during the execution of the test.

The behavior of the controller is represented in the Figure 29 which shows the evolution of the error during the test. The red line step represent the moment in which the image processing start. To evaluate the behavior of the controller we use the metric RMSE and not the ITAE like in the optimization process. RMSE is better at evaluating the optimal controller. Low values of the RMSE corroborates the excellent behavior of the optimized-controller.

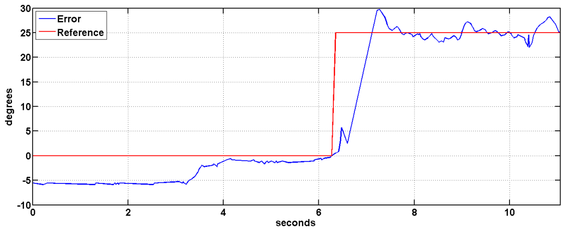


Fig. 29. Evolution of the error during a real test.

A video of this and other tests can be found at [22], [24].

VI. Conclusions

This work have presented two control approaches for UAS see and avoid task using micro unmanned aerial vehicles (MUAV). Both approaches were implemented using Fuzzy Logic techniques. The Cross-Entropy method was used to optimise the parameters for one of the controllers. The process of optimization was done testing 330 different controllers using the virtual environment ROS-Gazebo with the starmac aircraft model. Both controllers have been tested using an AR.Drone-Parrot in an indoor environment with good results. Table I depicts the comparison between the best tests for each approach. A RMSE value of infinite was achieved by the approach that was not optimised given that consistently lost the target because the controller was unable to respond accordingly, therefore not accomplishing the task. While the optimised version achieved RMSE values below 3 deg. The uses of the Cross-Entropy method reduce significantly the error during the test.

TABLE I
COMPARISON BETWEEN THE TWO APPROACHES

Type of controller	RMSE (degrees)
First Approach: PD-Fuzzy controller	9.576
First Approach: PD-Fuzzy controller	7.7752
Second Approach: CE-PID-Fuzzy controller	2.89
Second Approach: CE-PID-Fuzzy controller	3.044
Second Approach: Without Cross-Entropy optimization	∞

The quick response of the controller and the small error during the test indicates an excellent behavior of both controllers, besides the differences between the model with the simulator, the Starmac, the object folllowing test with the Ascending Technologies Pelican and the the aircraft AR.Drone-Parrot used in the tests presented in this paper.

The uses of the Cross-Entropy method made possible a significant size reduction of the base of rules given it allows to reduce the number of the membership functions and a notable reduction of the RMSE of the test. We are in process of designing new controllers for altitude control. These new controllers could include more inputs such as aircraft speed, range to target, etc. For future works a comparison between other optimization methods will be done in order to evaluate the accuracy of this novel method. The uses of this method will be extended to other parts of the Fuzzy controller in order to made the tunning of the Fuzzy controller totally autonomous, from the creation of the rules base until the definition of the fuzzy set of each variable.

ACKNOWLEDGMENT

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] "Ar.drone parrot," 2010. [Online]. Available: http://ardrone.parrot.com
[2] "United states department of defense," 2010. [Online]. Available: http://www.defense.gov
[3] O. Meister, N. Frietsch, C. Ascher, and G. Trommer, "Adaptive path planning for a vtol-uav," in *Position, Location and Navigation Symposium, 2008 IEEE/ION*, may 2008, pp. 1252 –1259.
[4] A. Moses, M. Rutherford, and K. Valavanis, "Radar-based detection and identification for miniature air vehicles," in *Control Applications (CCA), 2011 IEEE International Conference on*, sept. 2011, pp. 933 –940.
[5] A. E. R. Shabayek, C. Demonceaux, O. Morel, and D. Fofi, "Vision based uav attitude estimation: Progress and insights," vol. 65, no. 1-4, 2012, pp. 295–308.
[6] C. Martinez, I. Mondragon, M. Olivares-Mendez, and P. Campoy, "Onboard and ground visual pose estimation techniques for uav control," vol. 61. Springer Netherlands, 2011, pp. 301–320, 10.1007/s10846-010-9505-9.
[7] "Nintendo, kinect," 2012. [Online]. Available: http://en.wikipedia.org/wiki/Kinect
[8] "Hibrid system laboratory, berkeley university, quadrotor and kinect," 2012. [Online]. Available: http://hybrid.eecs.berkeley.edu/
[9] D. Wang, J. Liu, and R. Srinivasan, "Data-driven soft sensor approach for quality prediction in a refining process," *Industrial Informatics, IEEE Transactions on*, vol. 6, no. 1, pp. 11 –17, feb. 2010.
[10] M. Nikravesh, "Computational intelligent for reservoir management," in *Industrial Informatics, 2003. INDIN 2003. Proceedings. IEEE International Conference on*, aug. 2003, pp. 358 – 363.
[11] R. Caponetto, L. Fortuna, G. Nunnari, L. Occhipinti, and M. Xibilia, "Soft computing for greenhouse climate control," *Fuzzy Systems, IEEE Transactions on*, vol. 8, no. 6, pp. 753 –760, dec 2000.
[12] W. Tsui, M. Masmoudi, F. Karray, I. Song, and M. Masmoudi, "Soft-computing-based embedded design of an intelligent wall/lane-following vehicle," *Mechatronics, IEEE/ASME Transactions on*, vol. 13, no. 1, pp. 125 –135, feb. 2008.
[13] R. Y. Rubinstein and D. P. Kroese, *The Cross Entropy Method: A Unified Approach To Combinatorial Optimization, Monte-carlo Simulation (Information Science and Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2004.
[14] R. E. Haber, R. M. del Toro, and A. Gajate, "Optimal fuzzy control system using the cross-entropy method. a case study of a drilling process," *Inf. Sci.*, vol. 180, pp. 2777–2792, July 2010.
[15] M. Bodur, "An adaptive cross-entropytuning of the pid control for robot manipulators," in *Proceedings of the World Congress on Engineering, WCE 2007, 2007*, 2007, pp. 93–98.
[16] G. R. Bradski, "Computer vision face tracking for use in a perceptual user interface," no. Q2, 1998. [Online]. Available: http://citeseer.ist.psu.edu/585206.html
[17] K. Fukunaga and L. Hostetler, "The estimation of the gradient of a density function, with applications in pattern recognition," vol. 21, no. 1, Jan. 1975, pp. 32 – 40.
[18] M. Olivares-Mendez, I. Mondragon, P. Campoy, and C. Martinez, "Fuzzy controller for uav-landing task using 3d-position visual estimation," in *Fuzzy Systems (FUZZ), 2010 IEEE International Conference on*, july 2010, pp. 1 –8.
[19] I. Mondragón, M. Olivares-Méndez, P. Campoy, C. Martínez, and L. Mejias, "Unmanned aerial vehicles uavs attitude, height, motion estimation and control using visual systems," *Autonomous Robots*, vol. 29, pp. 17–34, 2010, 10.1007/s10514-010-9183-2. [Online]. Available: http://dx.doi.org/10.1007/s10514-010-9183-2
[20] M. Olivares-Mendez, P. Campoy, I. Mondragon, C. Martinez, and L. Mejias, "Aerial object following using visual fuzzy servoing," in *Research, Development and Education on Unmanned Aerial Systems (RED-UAS 2011)*, December 2011.
[21] Z. Botev and D. P. Kroese, "Global likelihood optimization via the cross-entropy method with an application to mixture models," in *Proceedings of the 36th conference on Winter simulation*, 2004, pp. 529–535.
[22] "Computer vision group-upm," 2012. [Online]. Available: http://www.vision4uav.eu
[23] "Motion capture system from vicon," 2012. [Online]. Available: http://www.vicon.com
[24] "Youtube channel computer vision group-upm," 2012. [Online]. Available: http://www.youtube.com/colibriprojectUAV
[25] "Robot operating system (ros)," 2012. [Online]. Available: http://ros.org/wiki/gazebo
[26] "Starmac-ros package. hybrid systems laboratory, uc berkeley," 2012. [Online]. Available: http://www.ros.org/wiki/starmac-ros-pkg
[27] "Computer vision group-upm. see and avoid fuzzy controller optimized using cross-entropy," 2012. [Online]. Available: http://vision4uav.eu/?q=researchline/SeeAndAvoidCE