Evolutionary Multi-Agent Systems as a Methodology for Developing Autonomous Controllers for MAV Teams

Fabio Ruini, Angelo Cangelosi

***Abstract* — This paper describes the approach we are following for using an evolutionary Multi-Agent System (MAS) for the control of teams of autonomous Micro-unmanned Aerial Vehicles (MAVs). Experiments previously carried out using a simple 2D model have demonstrated how the combination of MAS and Evolutionary Robotics (ER) methodologies can in principle be exploited for the development of a distributed control system for flying robots. The computer simulations have resulted in controllers evolved with the following capabilities: (1) navigation through unknown environments, (2) obstacle-avoidance, (3) tracking of a movable target, and (4) execution of cooperative and coordinated behaviours based on implicit communication strategies. In order to improve the robustness of results and their potential use in real MAV teams, a new 3D simulator was developed. This new simulation model mostly captures the essential flight dynamics in a 3D urban environment. The initial results obtained with the new 3D simulator demonstrate the feasibility of the approach. Simulation experiments demonstrate that evolutionary process successfully leads to the development of controllers for the search and hit tasks. Future work will focus on further extensions of this model to investigate the processes involved in communication and coordination between autonomous MAVs.**

***Index Terms* — Multi-Agent Systems, Evolutionary Robotics, Neural Networks, MAVs.**

I. Introduction

Evolutionary Robotics (ER) [1] uses genetic algorithms to evolve the neural controller of autonomous robots. Numerous studies based on ER have focused on ground-based navigation and exploration tasks [2]. More recently, the ER approach has been employed to domains other than ground-based robots, such as underwater autonomous vehicles [3].

The application of ER methodologies for air navigation and control is still at its infancy. Current approaches to the development of autonomous controllers for aircraft mainly rely on theoretical tools other than ER. Methodologies generally adopted are behavior-based robotics [4], genetic programming [5] evolution-based path planning [6], modeling field theory [7], classic graph search methods (such as A*[8]), and – rarely - neural networks [9][10]. Only recently, with the work carried out at the EPFL [11], a proper ER approach has been taken into account for the development of the whole evolutionary control system for a group of autonomous flying robots.

The work presented herein aims at the extension of ER techniques for unmanned aircrafts. In particular, our research focuses on teams of autonomous Micro-unmanned Aerial Vehicles (MAVs) [12] engaged on a "search and hit" task within an urban environment. The approach we have decided to follow relies on computer simulations developed through the mixture of Evolutionary Robotics (ER) and Multi-Agent Systems (MAS) methodologies (see [13] for an overview). Work at this stage will only be based on simulations.

As the adoption of MAS suggests, the behavior of MAV teams is governed by a distributed control system, rather than by a centralized one. Central controllers are generally considered faster and more efficient solutions when compared to distributed alternatives. But designing effective cooperative strategies for teams composed of many autonomous, unmanned vehicles could be a very hard task, as demonstrated for example by the works of Hussain [14] and Gaudiano [15]. Furthermore, distributed control provides a number of advantages. In particular, its non-critical reliance on any specific element can in turn guarantee increased reliability, safety, and – according to Wu [16] - even speed of response to the entire system. Richards and colleagues [5], in their systematic review of the numerous approaches to the development of centralized and distributed control systems for teams of unmanned vehicles, propose a useful classification of different methodologies. The approach we will follow is based on what they have identified as "reactive strategies". This implies that no central controllers are used and MAVs do not have to deal with any specific and pre-calculated flight path. The course of action they will undertake entirely relies on the mixture of low-level information gathered from the environment using their sensors, and information coming from different sources.

The work on MAV teams has some similarity with the swarm robotics approach such as those based on the concepts of flocking and swarming applied to autonomous aircraft (see for example [17]). In the simulations we describe here, instead, none of the essential traits of swarm behaviour are

Manuscript received May 1st, 2009. Effort sponsored by the Air Force Office of Scientific Research, Air Office Material Command, USAF, under grant number FA8655-07-1-3075.

All authors are with the Adaptive Behaviour and Cognition Research Group, School of Computing, Communications and Electronics, University of Plymouth, Drake Circus, Plymouth, PL4 8AA, UK (phone: +44 (0)1752 586288; e-mail: fabio.ruini@plymouth.ac.uk).

replicated. MAVs belonging to the same team are typically spread along the environment, not acting, even from a graphical point of view, as flocks of birds. At the same time they cannot rely on the fundamental rules of swarm/flock behaviour, which can be resumed as (1) separation, (2) alignment, and (3) cohesion.

II. Overview of previous work

The proposed approach based on the combination of ER and MAS methodologies was initially tested using a simple 2D scenario [18][19]. In the previous study, the simulated environment represented part of the Canary Wharf district in London, UK. An outline of the main buildings and skyscrapers present inside that area was included, and they were perceived as obstacles by the MAVs. The MAVs' task consisted in a classic "search and hit" scenario in the context of urban counter-terrorism. A target, which might corresponds to an enemy person or a vehicle, is deployed in a random position inside the reference area. A MAV team, composed by four unmanned aircraft starting each from different positions, has to navigate through the environment to reach the target and finally neutralize it performing a detonation (which also causes the loss of the individual MAV).

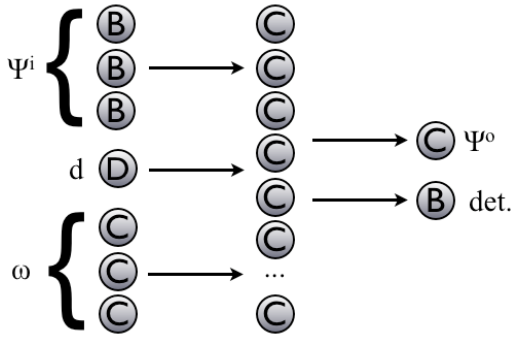

Fig. 1. Graphical representation of the basic neural network controller used for the 2D model. The input layer, on the left, contains three neurons encoding the horizontal angle separating the MAV and the target (Ψ^i), the distance (d) between the MAV and the target and the ultra-sonic perceptions (ω) of obstacles. Between the input and the output layer there is a layer made of 15 continuous neurons, activated by a tan-sigmoid function. The output layer, on the right, contains instead the neuron generating yaw (Ψ^o), plus the one dedicated to the detonation of the aircraft (det.) [B: Boolean, D: discrete, C: continuous]

The agents' behavior is governed by a three-layered fully connected feed-forward neural network (Figure 1). The controller receives sensorial inputs from the environment and in turn triggers the appropriate motor answer. Even though each individual MAV is endowed with its own neural controller, they all share the same connection weights and biases: they are, in fact, clones of each other.

Automatic Target Acquisition (ATA) is not provided to the MAVs. The assumption behind the MAV navigation system relies on the presence of a satellite-based target-tracking system able to monitor the target and broadcast real-time information about its position to all the MAVs engaged in the task. The goal of the MAV's neural controller is to exploit this information, combined with local low-level knowledge directly gathered by the embodied sensors, in order to find its way to the target.

The results obtained with 2D simulations demonstrated the validity of the chosen approach for different kinds of task. Particularly, four incremental experiments were carried out. In the basic scenario, the buildings are considered as a sort of "no-fly zones" for the MAVs. Agents can detect buildings, as well as the environment boundaries, the teammates and the target, via embodied ultra-sonic sensors. If a MAV attempts to fly over one of those areas its test ends at that point. In the second scenario, the target is a more "robust" one, as it requires two consecutive (i.e., taking place within a limited amount of time) hits to be neutralized. In the third and fourth scenarios, we introduced a moveable target, able to move away from the approaching MAVs. In order to be neutralized, the target has respectively to be attacked individually or cooperatively as in the two previous setups.

The neural architecture shown in Figure 1 has required slightly modifications to deal with the different input/output setups.

For all the experiments just outlined the evolutionary process applied on the MAVs has been able to identify optimal solutions. At the end of the evolution, the success rates obtained on the first three scenarios were particularly positive. MAV teams were able to reach a success rate (intended as the average of tests concluded successfully for the members of the last generation) of 93.46%, 87.18%, and 73% respectively. The fourth scenario only (target able to escape and requiring a cooperative attack to be eliminated) produced relatively limited results (41.28% tests succeeded).

III. Description of the new 3D model

The new work reported in this paper focuses on the extension of the original 2D model to a 3D simulator. This will also require an extension of the number of degrees of freedom (DOF) involved in the evolutionary and neural control process.

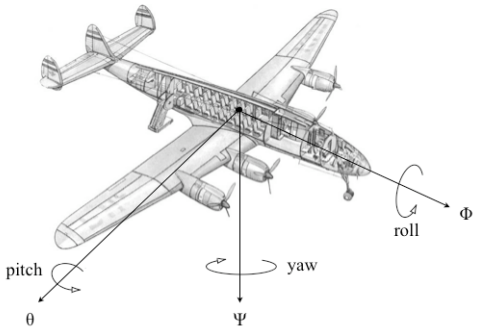

Fig. 2. The degrees of freedom available to a typical fixed-wing aircraft

The new 3D model introduces two new DOFs. In the previous simulator, MAVs were only able to rotate their body clockwise or counter-clockwise, while constrained to move along their heading direction at every time step. They were essentially relying on a single DOF. The 3D aircraft we are

now modeling in the 3D environment is instead able to rotate along three different axes. If we consider an orthogonal axis system fixed on the aircraft and constrained to move with it (what is sometimes defined as "body axis system"), the rotations that are possible to the MAVs can be respectively defined as yaw, pitch, and roll. In details: yaw corresponds the rotation of the aircraft around its vertical axis; pitch is the rotation around the side-to-side axis; roll refers to the rotation around the front-to-back axis. Figure 2 shows the rotation axis of the aircraft described above (for a more accurate description of the systems of axes and notations used in flight dynamics, see for example [20]).

The use of a 3D simulator involves an additional issue in the domain of autonomous controllers for aircraft. The issue is terrain avoidance (see for example [21]), and it represents a key capability that should be provided to MAVs along with obstacle avoidance. The ability of the MAVs to rotate around their horizontal axis (pitch) causes their altitude to change with time. For the sake of simplicity, the scenario used for this simulation is assumed as a flat ground plane located at the sea level. Though an explicit control strategy for terrain avoidance is not planned in the model, as we will see later the MAVs will acquire this ability as a side product of the evolutionary process. MAVs have to be able to avoid crashing into the ground, but also must not fly at an excessive height. The simulator, in fact, will consider "lost", without any distinctions, all the aircraft exiting from the bottom or top boundaries of the environment.

The environment simulated in the 3D model measures 1,000 (X) x 750 (Y) x 600 (Z) graphical units (GU). The target is represented by a sphere with 15 GU radius. All the MAVs have the identical shape and size with length of 5 GU and wingspan of 3.5 GU.

The task in this new simulator is the same as before. A certain number of MAVs – all members of the same team, but starting from different positions - have to navigate through the environment, reach a certain target deployed inside a central area (without exiting the environment boundaries) and then perform a detonation in order to neutralize it. The test is considered successfully when the detonation happens within a 15 GU distance from the target. If all MAVs "die" without having neutralized the target, the test is considered failed. In addition to losing an aircraft because it has self-detonated or has exited the environment boundaries, during a test a MAV can also die if it runs out of energy or if it collides against a teammate.

A. Neural network controller

Our research relies on neural network controllers developed through ER methodologies [1]. This implies that a set of sensors "embodied" on the MAVs have to continuously gather information from the surrounding environment and from the aircraft itself in order to autonomously trigger the most appropriate behaviour.

The controller we have designed is based on a feed-forward neural network, constituted by one input and one output layers (a graphical representation of this architecture is provided in Figure 3). The input layer receives five chunks of information: (1) horizontal angle between the MAV and the target, (2) current pitch angle, (3) current roll angle, (4) delta height compared to the target, and (5) distance to the target. The output layer provides instead the effectors of the MAV's behaviour: (1) yaw, (2) pitch, (3) roll, and (4) detonation.

The two layers are not fully connected. While the neural modules dedicated to modify the MAV flight trajectory (i.e., yaw, pitch and roll) receive their incoming connections from all the input neurons, the module dedicated to the detonation only receives information about the distance between the MAV and the target.

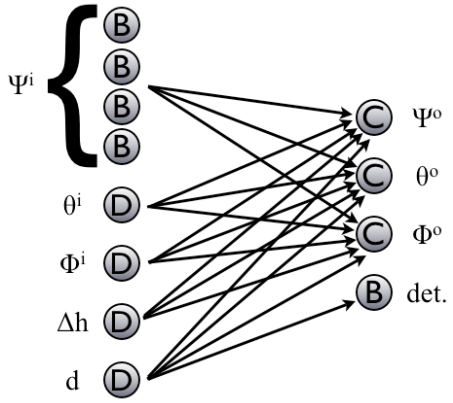

Fig. 3. Graphical representation of the neural network controller used for the 3D model. The input layer, on the left, contains four neurons encoding the horizontal angle separating the MAV and the target (Ψ^i), the MAV's pitch and roll angles (θ^i and Φ^i respectively), the delta height (Δh) and the distance (d) between the MAV and the target. The output layer, on the right, contains instead the neurons generating yaw, pitch and roll (Ψ^o, θ^o, and Φ^o), plus the one dedicated to the detonation of the aircraft (det.)

Each controller is characterised by 25 connection weights and 4 biases (each of them applied to one of the output layer's units). All the neurons belonging to the input layer do not have any bias and use an identity transfer function, which does not modify the values set in input. The information fed to the network is processed according to the connection weights and then passed to output layer. The output neurons that manage yaw/pitch/roll produce a continuous output value. They are activated according to a tan-sigmoid function (slope 1.0), which outputs are in the range [-1.0; +1.0]. The output unit dedicated to the detonation is instead a Boolean one activated by a step function with a 0 threshold. When it turns into 1 the MAV detonates, while nothing happens when the value of this neuron is 0.

For measuring pitch and roll angles (also referred to as "attitude") we have chosen a right handed axis system. This means that positive pitch is nose up and positive roll is right wing down. The correlation between the output values generated by the controller and the effects played on the aircraft is 1-to-1. During each time step a MAV can therefore perform a yaw between -1.0 (nose left) and +1.0 (nose right), a pitch between -1.0 (nose down) and +1.0 (nose up), and a roll ranging from -1.0 (left wing down) and +1.0 (right wing down) degrees. The 0 output does not generate any change to

the current aircraft's attitude. In order to make the modeled MAVs as flexible as possible in their movements, we have decided to implement roll as an independent component of flight motion. This approach differs for example by what Watson and colleagues [17] have proposed by associating roll with yaw, where when the aircraft performs a yaw, it performs a correspondent roll also, with an intensity calculated according to a certain formula.

B. Inputs calculation and discretisation algorithms

As already pointed out in previous work and confirmed by the preliminary analyses carried out using the new 3D model, in this particular experimental scenario the neural network controllers have demonstrated to perform better when the input units receive discrete rather than continuous values. The discretisation algorithms applied on the continuous variables collected by the MAVs' sensors assume therefore a fundamental role.

The most important information that the controllers have to process in order to adjust the flight trajectory of the MAVs are those received by the external target-tracking system: distance and horizontal angle from the target.

The distance between the MAV and the target is simply calculated as the Euclidean distance between their respective centre points. Then the discretisation process takes place reducing the original value to one out of nineteen discrete values, according to Table I. Consider that a distance compatible with the aircraft's detonation range is discretised as 1.0, while bigger distances correspond to values ranging from 0.95 to 0.10 with steps 76.19 long. The maximum distance between two points, inside this environment, is 1,386.54 GU.

TABLE I
DISCRETISATION TABLE FOR d

Original distance (d)	Discretised distance
0 <= d <= 15.0	1.0
15.0 < d <= 91.19	0.95
91.19 < d <= 167.38	0.90
167.38 < d <= 243.57	0.85
…	…
1,310.23 < d <= 1,386.54	0.10

The horizontal angle between the MAV's heading direction and the target is obtained projecting the two objects on the same two-dimensional surface. In other words, we assume that they are at the same height and that the MAV's pitch angle is 0. This measure is 0 when the MAV is heading the target. The value increases counter-clockwise, tending to 360. The neural network input consists of four Boolean neurons, encoding via a Gray Code system the angle calculated through the above procedure (see Table II).

Sensors embodied on the aircraft to date are limited to those encoding basic information such as current height, and pitch and roll angles.

Delta height is calculated as the difference between the current height of the target and the height of the MAV, both measured according to their centre points. The resulting value is therefore negative when the robot is flying over the center of the target (and needs to reduce its altitude through a negative pitch), positive vice-versa. In this way there is an immediate correspondence between the input received by the network and the output that it has to produce in response.

TABLE II
DISCRETISATION TABLE FOR ψ

Original horizontal angle (ψ)	Discretised horizontal angle
0 <= ψ <= 11.25 348.75 <= ψ <= 0	0 0 0 0
11.25 < ψ<= 33.75	0 0 0 1
33.75 < ψ <= 56.25	0 0 1 1
56.25 < ψ <= 78.75	0 0 1 0
78.75 < ψ <= 101.25	0 1 1 0
101.25 < ψ <= 123.75	0 1 1 1
123.75 < ψ <= 146.25	0 1 0 1
146.25 < ψ <= 168.75	0 1 0 0
168.75 < ψ <= 191.25	1 1 0 0
191.25 < ψ <= 213.75	1 1 0 1
213.75 < ψ <= 236.25	1 1 1 1
236.25 < ψ <= 258.75	1 1 1 0
258.75 < ψ <= 281.25	1 0 1 0
281.25 < ψ <= 303.75	1 0 1 1
303.75 < ψ <= 326.25	1 0 0 1
326.25 < ψ < 348.75	1 0 0 0

The normalization algorithm considers irrelevant a 15 GU distance. Values bigger than 15 and less than -15 are discretised according to Table III.

TABLE III
DISCRETISATION TABLE FOR Δh

Original delta height (Δh)	Discretised delta height
-15.0 <= Δh <= 15.0	0.0
15.0 < Δh < 73.5	0.1
73.5 < Δh < 132.0	0.2
…	…
541.5 < Δh <= 584.0	1.0
-73.5 < Δh < -15.0	-0.1
-132.0 < Δh < -73.5	-0.2
…	…
-584.0 <= Δh < -541.5	-1.0

TABLE IV
DISCRETISATION TABLE FOR θ AND Φ

Original pitch/roll angle (θ || Φ)	Discretised pitch/roll angle
-2.0 <= (θ || Φ) <= 2.0	0.0
2.0 < (θ || Φ) < 18.0	0.1
18.0 < (θ || Φ) < 36.0	0.2
…	…
162.0 < (θ || Φ) <= 180.0	1.0
-18.0 < (θ || Φ) < -2.0	-0.1
-36.0 < (θ || Φ) < -18.0	-0.2
…	…
-180.0 <= (θ || Φ) < -162.0	-1.0

Finally, current pitch and roll angles are discretised at the same way, according to Table IV. We have opted for a right

handed system, therefore pitch angle is positive when the MAV's nose is facing up, negative when it is facing down. The same applies for the roll angle. Its value is positive when the aircraft's right wing is lower than the left one, negative in the opposite case. Both discretised pitch and roll angles have value 0 when the variations from the "equilibrium state" (i.e., the MAV's front-to-back axis parallel to the ground and not rolled) are smaller than 2 degrees.

IV. Preliminary Results

The results described in this section refer to the 3D experimental setup based on the parameters listed below.

Each MAV's movement is 3 GU long and cost 1 energy unit. The flying speed is assumed as constant, it cannot be modified during flight and cannot therefore affect the energy consumption. The energy stored into each MAV amounts to 1,000 fuel units (FU). MAVs can therefore fly for 3,000 GU.

An initial population of 100 teams is created. When the simulation begins, the individuals' genotypes (each connection weight and bias in the neural network) are created with random values in the range [-5.0; +5.0]. All agents within a team are genetic clones, thus sharing the same genotype.

Each team is formed by 4 MAVs. A team is tested for 8 epochs. At the beginning of each epoch, the target is randomly placed in a different position, always 15 GU above the ground level. All the MAVs belonging to the same team are deployed inside this environment, in different starting positions close to the four environment corners and facing the centre of the cuboidal arena (with some random noise added to alter their initial position).

The fitness formula used for driving the evolutionary process has been kept as simple as possible:

$$f = -\alpha + \beta \qquad (1)$$

where: α is the average distance (measured in GU) between the target and the team member detonated closest to it, calculated basing on the various tests; β is the average amount of energy retained by the MAV detonated closest to the target during each test. If no MAVs detonate during the entire sets of test epochs (which is a condition that could frequently happen during the first stages of evolution), the values of α and β are manually set to 1,386.54 and 0 respectively.

At the end of each generation, the 10 teams that have scored the best performances according to the fitness formula are selected for reproduction. Every parent team generates 9 offspring teams, which inherit the same genotype. For each new team, a mutation process is applied. All the connection weights and biases are affected, with probability 0.2, by the addition of a random value included within the range [-1.0; +1.0]. Elitism is also applied, so the first offspring of the best individual within a given generation is not affected by any kind of modifications of its genome and is kept in the next population as it is.

The entire process is iterated for 350 generations. Five different replications (i.e. same simulation with different initial random genotypes at generation 0) are carried out. The results reported here are averaged across the five replications in order to identify a general evolutionary trend.

The charts below provide a summary of the results generated by the evolutionary process.

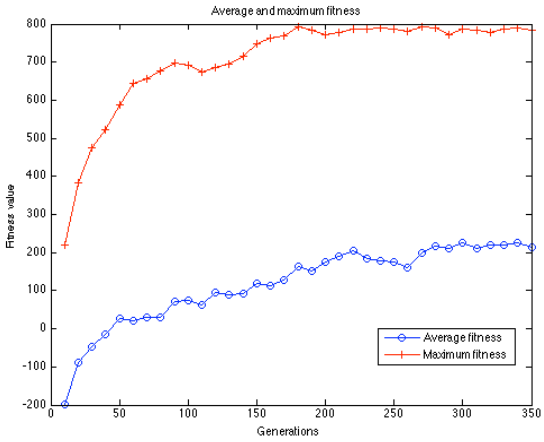

Fig. 4. Average and maximum fitness during the 350 generations

In Figure 4 we can see how average and maximum fitness increase generation by generation, until reaching what looks like a steady state after 300 generations. As expected, both lines start from low fitness levels (for the first 50 generations the average fitness is negative, indicating that the "average team" is not yet able to perform the desired task) and then quickly increase.

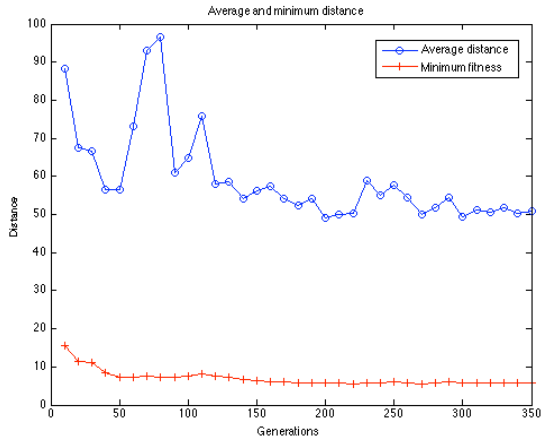

Fig. 5. Average and minimum distance between the target and the detonation point of the MAVs detonated closest to it

As mentioned above, the fitness formula used for evaluating the performance of a MAV team has two main components. The first is the average distance between the target and the MAV detonated closest to it, during the 8 tests carried out for each team. Figure 5 shows how the minimum average distance (i.e., the one scored by the best swarm averaging the 8 tests carried out) reaches a steady state after few generations. The line representing the average for the entire population is instead more jagged and decreases more slowly. The second component of the fitness formula is the amount of energy retained by the MAVs that have neutralized the target,

averaged for all the successful tests performed. This parameter has been included into the fitness formula with the purpose of creating a discriminatory effect whenever most of the MAV teams would have been able to correctly reach and neutralize the target. The idea is to favour those teams able to perform the task faster than other members of the population. We were therefore expecting a curve characterised by values increasing generation by generation (or at least starting to increase after that the average success rate for the entire population has exceeded a certain threshold). The results presented in Figure 6 show how this expected phenomena in fact does not take place. This is presumably due to the amount of generations evolved, which is too limited for this trend to appear.

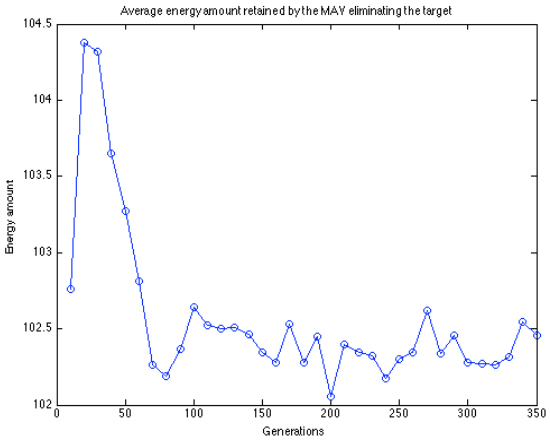

Fig. 6. Average amount of energy retained by the MAV eliminating the target

It is interesting to note that even if the fitness graph suggests that the evolution has reached a steady stable, in reality this is not the case. Looking at Figure 7 it becomes obvious that the evolutionary process is still going on. What is happening is that the entire population is converging to the optimum point individuated by the evolutionary algorithm. This phenomenon might results difficult to see in Figures 4 and 5, but it is clearly observed in Figure 7.

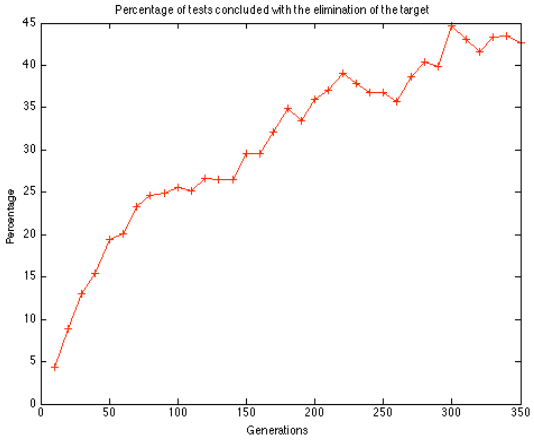

Fig. 7. Percentage of tests concluded successfully

Finally, Figure 8 shows a difference emerging from the new simulations compared to what was happening with the 2D simulator. In the previous 2D model, during the first generations MAVs used to detonate immediately at the beginning of each test. Then gradually, generation by generation, the percentage of aircraft detonating reduced, tending to the expected value of 1. Now the process takes place in the opposite way. During the first generations most of the MAVs end the various tests exiting by accident from the environment boundaries. This proportion slowly diminishes, while the average number of MAVs detonating within each team increases as expected.

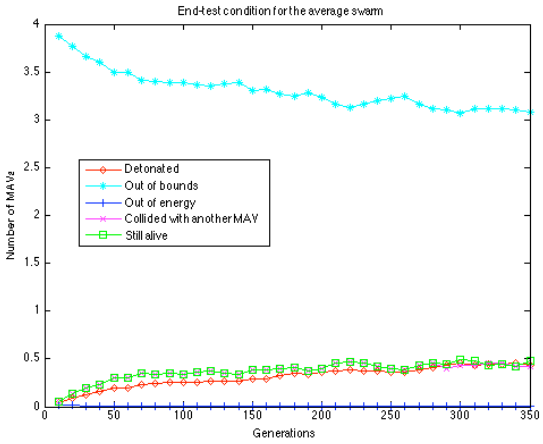

Fig. 8. End-test condition for the members of the "average" swarm. The plot shows, during a typical test, how many MAVs detonate, exit from the environment boundaries, run out of energy, collide against a team-member, or survive

The effect described above is probably due to the size of the environment, which - although adding a new dimension - is relatively smaller than the one used for carrying out the 2D simulations, thus resulting in MAVs that can exit more easily from the environment boundaries.

V. Discussion and Conclusion

When we consider the validity of this work for real MAVs control, it is fundamental to consider that this simulation model intentionally uses a high level perspective, i.e. focuses on generic flight navigation dynamics, and navigation and search strategy. At this preliminary stage, we are not interested in studying low-level aspects of the MAVs' physical interactions, such as considering air resistance on specific aircraft configurations. We aim to provide the agents with realistic, general flight dynamics that a real aircraft could effectively reproduce. It will be part of future work, in particular for aircraft engineering, to focus on the design of robotic MAV platforms capable of accurately translating controller's decisions into proper commands for the aircraft.

One of the main contributions of this model is to propose the use of combined ER and MAS systems for the distributed control and coordination of autonomous flying agents. One of the reasons behind the lack of progress in this area could be due to the absence of a cheap and reliable robotic platform to be used for MAV flying experiments. It is undisputable that the initial success of the ER field has been greatly due to the development and availability of easy reach robot platforms such as Khepera [22]. This has allowed researchers to test

computer-elaborated behaviours on a physical robot facing the full complexity of the real world[2]. However, a potentially important development in the area of ER and flying behavior is the contribution by Thomas and colleagues [23], who have recently presented a test-bed flying vehicle that could potentially fill this gap. The idea behind their work consists in installing inexpensive commercial off-the-shelf components on a basic radio controlled airplane in order to transform it to an autonomous vehicle. They have employed two devices manufactured by Crossbow®, namely the MNAV (a solid state micro-electro-mechanical system unit that incorporates three-axes measurement of acceleration and angular rates, along with magnetometers, barometric pressure sensors, three-axis temperature sensors, and an integrated GPS receiver) and the Stargate (a single board computer running a 400MHz processor, which can support a 802.11 Wi-Fi network card). In their case, software running on a laptop computer wirelessly connected to the aircraft makes possible to communicate with it, receiving telemetry data and uploading new flight paths. Even if this approach does not constitute a significant element of novelty, it demonstrates how current technologies are mature for the development of a new "flying Khepera". Once an affordable platform can be identified, carefully tested and adopted as a standard by researchers across the world, the investigations on the area of development of autonomous controllers for flying robots through ER methodologies could potentially register a significant boost.

When we compare ER for ground and air robotics, another issue to consider is safety and robustness. A flying robot is necessarily less resistant than a ground-based one and exposed to bigger risks during experiments. While a Khepera can harmlessly hit a wall if provided with a wrong instruction by the controller, a MAV could instead crash on the ground seriously compromising its functionality. And an additional issue related to this is the need of ample lab space required for carrying out flying experiments. Small ground-based robots typically used in ER do not require much room, since experiments involving this kind of platforms are frequently carried out inside small squared arenas with a few meters side. Aircraft generally need instead a much bigger area in order to work properly. These two points have in turn both a financial and a practical impact on the research and they have to be taken into account since they cannot be excluded.

VI. Future work

Plans for future work consider four main directions of research.

A. Introduction of obstacles

In order to replicate all the results obtained with the 2D simulations, the next step will consist in extending the new model and add 3D obstacles such as buildings. MAVs will have to be provided with embodied sensors capable of detecting the presence of those obstacles. These sensors need to be added to the three used in the old 2D simulator. A tentative solution consists in using nine sensors configured as shown in Figure 9.

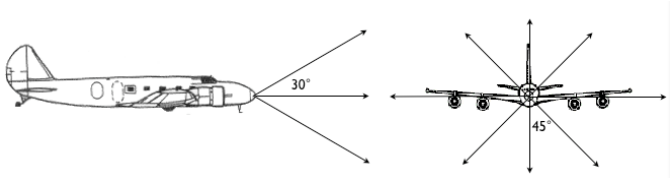

Fig. 9. Isometric side and front views that show how the ultra-sonic sensors could be deployed on the MAVs

B. Cooperation and language

Previous work has considered tasks requiring explicit cooperation between teammates. It has been demonstrated how the introduction of extremely simple forms of implicit communication could allow cooperative behaviours to evolve without requiring the pre-design of a cooperative strategy.

Future investigations along this direction will focus on the introduction of various forms of communication between teammates (from explicit, pre-defined communication protocols, to self-organizing signaling systems). The purpose of adding communication capabilities can clarify how direct and local agent-agent communication could lead to a better level of coordination between the MAVs, and in turn allowing the teams to increase their effectiveness in performing complex tasks. The approach toward the evolution of self-organising communication systems will be based upon symbol grounding theory as elaborated by Cangelosi et al. [24].

C. Provide the controller with short-term memory

During the last decades, many different methods to add short-term memory to a perceptron architecture have been proposed (see for example the widely used Elman networks [25]). This principle, applied to the development of neural autonomous controllers for robots, has proven to be particularly beneficial for navigation tasks within unknown environments [26]. Future work will consider the introduction of memory structures into the agents' controllers. Since the neural architecture we are using in the current 3D model does not include any hidden layer, future work will focus on the implementation of a Jordan network [27]. A set of context units will be then added to the neural network, reintroducing at time $t+1$ the values produced by the output units at time t.

D. Multi-threading

The amount of time requested by the computer simulator to perform the evolutionary process represents a serious issue that has to be solved. The C++ code on which the simulator is based on has already been optimized at the best of our knowledge, disabling graphics rendering during the evolutionary process, using simplified functions for collisions detection, etc. Nonetheless, each simulation replica elaborated and presented herein has required an average of about 60 hours of computation to evolve the relatively modest amount of 350 generations. Although the evolutionary algorithms slowness in converging toward an optimal solution represents

[2] At the same time, developing a computer simulator aimed to flying robots presents issues definitely more complex than the ones coming from the development of a simulator for ground-based robots. The Khepera's success has been strongly helped by the availability of good computer simulators.

a common concern in ER, in this case there should be an edge for improvement. On the basis of the actual trend in computer science, looks like the most promising approach toward optimization involves the introduction of multi-threading principles. Future work will therefore include the transformation of the developed simulator from a single-threading to a multi-threading application. In our lab, this improvement will make possible to fully benefit from the employment of P-ARTS (Plymouth Advanced Robot Training Suite), a powerful computer cluster recently awarded by our research group as part of the Apple® PARTS program.

Finally, it is interesting to consider another aspect related to simulation speed. In both the models described in this paper all the MAVs belonging to a certain team share the same controller, so it is essentially a single controller that evolve. If we do not take into account complex setups as those requiring communication and/or involving a "robust" target, it might look feasible to evolve teams made by only one MAV each in order to shorten the amount of time required by the evolutionary process. What immediately appears looking at Figure 10 is that the advantages coming from the use of a single MAV (which roughly reduces the computational efforts by one fourth if compared to the scenarios described herein) are more than mitigated by the massively larger number of generations required in order to evolve the desired behaviour.

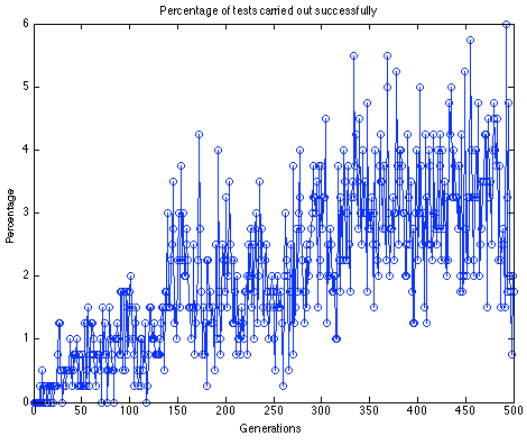

Fig. 10. Percentage of tests concluded successfully for a MAV team made of a single member. As it is possible to see into the plot, the success rate merely reaches a maximum of 6% in 500 generations

What happens is quite reasonable. During the first generations, in fact, MAVs approach and neutralize the target purely by chance. Then this evolutionarily advantageous behaviour improves and spread among the entire population. The more MAVs are employed, the more likely is that this process will take place since the very first generations. The likelihood of succeeding in the task is in fact positively correlated with the team size. The experiments carried out have demonstrated how the selection of a proper team-size is an important aspect to consider.

DISCLAIMER

The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies and endorsements, either expressed or implied, of the Air Force Office of Scientific Research or the U.S. Government.

The U.S. Government is authorized to reproduce and distribute reprints for government purpose notwithstanding any copyright notation thereon.

REFERENCES

[1] S. Nolfi, and D. Floreano, *Evolutionary Robotics*, MIT Press, 2000.
[2] D. Floreano, and C. Mattiussi, *Bio-Inspired Artificial Intelligence,* MIT Press, 2008.
[3] V. Kodogiannis, "Neuro-control of unmanned underwater vehicles," in *Int. Journal of Systems Science*, Vol. 37(3), pp. 149-162, 2006.
[4] M. Dong, and Z. Sun, "A behavior-based architecture for unmanned aerial vehicles," in *Proc. IEEE Int. Sym. on Intelligent Control*, pp. 149-155, 2004.
[5] M.D. Richards, D. Whitley, and J.R. Beveridge, "Evolving cooperative strategies for UAV teams," in *Proc. of GECCO 2005*, pp. 332-339**,** 2005.
[6] D. Rathbun, S. Kragelund, A. Pongpunwattana, and B. Capozzi, "An evolution based path planning algorithm for autonomous motion of a UAV through uncertain environments," in *Proc. of 21st Digital Avionics System Conf.*, pp. 608-619, 2002.
[7] R. Deming, L. Perlovsky, and R. Brockett, "Sensor fusion for swarms of unmanned aerial vehicles using modeling field theory," in *Proc. of KIMAS 2005*, pp. 122-127, 2005.
[8] Y. Qu, Q. Pan, and J. Yan, "Flight path planning of UAV based on heuristically search and genetic algorithms," in *Proc. of IECON 2005*, pp. 45-49, 2005.
[9] G. Buskey, J. Roberts, P. Corke, P. Ridley, and G. Wyeth, "Sensing and control for a small-size helicopter," in *Experimental Robotics VIII*, Springer Berlin, pp. 476-486, 2003.
[10] R. De Nardi, O. Holland, J. Woods, and A. Clark, "SwarMAV: a swarm of miniature aerial vehicles," in *Proc. of 21st Int. UAV Systems Conf.*, 2006.
[11] S. Hauert, J.C. Zufferey, and D. Floreano, "Evolved swarming without positioning information: an application in aerial communication relay," in *Autonomous Robots*, Vol. 26(1), pp. 21-32, 2009.
[12] J.M. Sullivan, "Revolution or Evolution? The rise of the UAVs," in *IEEE Technology and Society Magazine*, Vol. 25(3), pp. 43-49, 2006.
[13] Wooldridge, M., *An Introduction to MultiAgent Systems*, John Wiley & Sons, 2009.
[14] T. Hussain, D. Montana, and G. Vidaver, "Evolution-based deliberative planning for cooperative unmanned ground vehicles in a dynamic environment," in *Proc. of GECCO 2004*, pp. 1185-1196, 2004.
[15] P. Gaudiano, E. Bonabeau, and B. Shargel, "Evolving behaviors for a swarm of unmanned air vehicles," in *Proc. of SIS 2005*, pp. 317-324, 2005.
[16] A.S. Wu, A.C. Schultz, and A. Agah, "Evolving control for distributed micro air vehicles," in *Proc. of IEEE Conf. on Computational Intelligence in Robotics and Automation Engineers*, pp. 174-179, 1999.
[17] N.R. Watson, N.W. John, and W.J. Crowther, "Simulation of Unmanned Air Vehicle Flocking," in *Proc. of TPCG'03*, pp. 130-137, 2003.
[18] F. Ruini, and A. Cangelosi, "Distributed Control in Multi-Agent Systems: a Preliminary Model of Autonomous MAV Swarms," in *Proc. of FUSION 2008*, pp. 1043-1050, 2008.
[19] F. Ruini, A. Cangelosi, and F. Zetule, "Individual and Cooperative Tasks performed by MAV Teams driven by Embodied Neural Network Controllers," in *Proc. of IJCNN 2009*, in press.
[20] M. Cook, *Flight Dynamics Principles*, Elsevier, 2007.
[21] T. Netter, and N. Franceschini, "A Robotic Aircraft that follows Terrain Using a Neuromorphic Eye," in *Proc. of the 2002 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Vol. 1, pp. 129-134, 2002.
[22] F. Mondada, E. Franzi, and A. Guignard, "The Development of Khepera," in *Proc. of 1st Int. Khepera Workshop*, pp. 7-14, 1999.
[23] P. R. Thomas, and A. K. Cooke, "Simulation of an Automated Flight Test Safety System for Autonomous System Identification of Small UAVs," in *Proc. of 24th Int. Conf. on Unmanned Air Vehicle Systems*, pp. 35.1-35.16, 2009.
[24] A. Cangelosi, V. Tikhanoff, J.F. Fontanari, and E. Hourdakis, "Integrating language and cognition: A cognitive robotics approach," in *IEEE Computational Intelligence Mag.*, Vol. 2(3), pp. 65-70, 2007.
[25] J.L. Elman, "Finding Structure in Time," in *Cognitive Science*, Vol. 14, pp. 179-211, 1990.
[26] A.L. Nelson, E. Grant, J.M. Galeotti, and S. Rhody, "Maze exploration behaviors using an integrated evolutionary robotics environment," in *Robotics and Autonomous Systems*, Vol. 46, pp. 159-173, 2004.
[27] M.I. Jordan, "Attractor dynamics and parallelism in a connectionist sequential machine," in *Proc. of Eighth Annual Conf. of the Cognitive Science Society*, pp. 531–546, 1986.