Distributed Control in Multi-Agent Systems: A Preliminary Model of Autonomous MAV Swarms

Fabio Ruini
Adaptive Behaviour and Cognition Research Group
School of Computing, Communications and Electronics
University of Plymouth, U.K.
fabio.ruini@plymouth.ac.uk

Angelo Cangelosi
Adaptive Behaviour and Cognition Research Group
School of Computing, Communications and Electronics
University of Plymouth, U.K.
a.cangelosi@plymouth.ac.uk

Abstract - This article focuses on the use of Multi-Agent Systems for modelling of Micro-unmanned Aerial Vehicles (MAVs) in a distributed control task. The task regards a search and destroy scenario in the context of security and urban counter-terrorism. In the simulations developed, a swarm composed of four autonomous flying robots, driven by an embodied neural network controller, has to approach a target deployed somewhere within the given environment. When close enough to the target, one of the aircraft needs to carry out a detonation in order to neutralize it. The controllers used by the MAVs evolve through a genetic algorithm. The preliminary results presented here demonstrate how the adaptive evolutionary approach can be successfully employed to develop controllers of this kind. The MAV swarms evolved in this way are in fact able to reach and hit the target, navigating through an obstacle-full environment. Further works on this model will focus on the development of a 3D physical simulator, in order to move towards the usage of MAVs with neural network controllers in real applicative urban scenarios.

Keywords: MAVs, multi-agent systems, autonomous robotics, obstacle-avoidance, neural networks, genetic algorithms, embodied cognition.

1 Distributed control in Multi-Agent Systems (MAS)

Distributed control, particularly when it requires a certain level of coordination/cooperation [1][2][3], is a notably interesting problem from both a technological and scientific perspective. Compared to centralised control where a central controller (e.g. human operator or airplane's "leader" agent) is responsible for (pre)planning, task-assignment and supervision of the coordination task, in distributed control systems intelligent autonomous (or semi-autonomous) agents are capable of sensing, acting, cognition and communication and together contribute to the task solution. These network-centric systems only require partial interaction with other agents, and may necessitate simpler architectures and individual resource requirements as knowledge is distributed in the population.

The significant advantages of this approach are that the system is more robust, adaptive and fault tolerant since there is no critical reliance on any specific individual, and that decentralization results in increased reliability, safety and speed of response [3][4]. In addition, distributed approaches have the benefit of not requiring the full pre-planning of the cooperative strategy. Adaptive solutions can emerge at runtime through the interaction between autonomous individuals and from the task and environment requirements, which might not be fully accessible at the beginning of the problem.

Studies on distributed control greatly benefit from the utilization of Multi-Agent Systems (MAS), since they provide a platform for simulation and testing of various hypotheses based on the principle of distributed (artificial) intelligence [5]. Distributed control MAS approaches have been used in various domains, such as unmanned air, terrestrial/underwater vehicles, search and rescue scenarios, collective robotics, social cognition etc. For example, Sastry and colleagues [4] have focused on coordination and distributed control in unmanned underwater vehicles; Sykara and collaborators [6] have concentrated their attention on the study of hybrid rescue group systems based on humans, software agents, and autonomous robots. What they propose are coordination architectures capable of quickly finding optimal solutions to the combined problems of task allocation, scheduling, and path-planning subject to system constraints. In the SWARM-BOT project [3][7] groups of robots evolve a cooperative strategy for exploratory tasks. In such a study distributed coordination implies that the characteristics of the group's behaviour (e.g. individual sensorimotor strategies, the roles played by the different robots, the synchronization problems raised by their interactions) are not managed centrally by one or few "leaders" but are the result of self-organizing processes instead. Examples of these processes are "positive feedback" (if each individual of a group follows a rule of the type "do what the majority does", the individuals' behaviours will tend to become homogeneous) or "consumption of building blocks" (e.g. if the number of individuals forming a group is limited, the process of convergence towards the same behaviour caused by a positive feedback mechanism will necessarily slow down and then stop). Finally, various MAS models of social cognition have been proposed, such as those modelling animal collaborative tasks such as in ant

colonies (which have inspired the SWARM-BOT application) and predator group behaviour [8].

The goal of this paper is to introduce a new methodology, based on MAS, to develop autonomous controller systems for unmanned aerial vehicles (UAVs). We will focus on a particular category of UAVs characterised by their very small size, the so-called Micro-unmanned Aerial Vehicles (MAVs). The interest in such a model arise from the fact that, given their specifications, swarms composed by many MAVs could be successfully employed for counter-terrorism operations to be carried out within urban crowded environments.

2 Autonomous UAVs/MAVs path-planning

Typically, the UAVs[1] used nowadays in real applicative scenarios are dynamically remote controlled (think for instance about the famous Predator, which is currently widely employed in the main warfare environments) by a human crew staying in a remote position. Many UAVs, at the same time, also have their own guidance systems, via which they can fly autonomously. The limitation is that usually these guidance systems are slightly similar to automatic pilots used within the civilian aviation domain as they simply provide a means of keeping the UAVs following a given pre-planned route.

During the last few years we have noticed an increasing interest in developing intelligent autonomous UAVs' controller systems. The focusing toward autonomous guidance systems is not only an economical matter, although the use of autonomous aircraft instead of the usual combination "manned airplane plus human pilot" would allow to save the enormous amount of money normally required for pilot training, skills upgrading, and so on. Computer software can frequently outperform humans in carrying out many different tasks, both in terms of reliability (for example, consider the "dull" factor which Cambone and colleagues refer to [9]) and accuracy (computer software is more accurate than a human pilot to perform an already planned manoeuvre and, most important of all, it is able to perform with the shortest reaction time possible). For tasks where more than a single UAV has to be employed, for example because a certain level of cooperation is required, the magnitude of the problem increases accordingly.

According to Richards and colleagues [10] current approaches for autonomous cooperative UAV control can be separated into the following strategies[2]:

- *deliberative approach*: focused on developing a specific flight path for each UAV to follow. Such flight paths are rigid and they cannot be altered even in the event that new information is discovered. In other words, the entire scenario is assumed to be already known. This approach could be successfully employed for civilian flight planning, but it simply results in being unusable in a military perspective. An example of this approach is in the work by Ablavsky [11];
- *adaptive replanning approach*: in order to achieve some degrees of flexibility, few deliberative systems incorporate an element of adaptive replanning. Like in the deliberative approach, in the adaptive replanning the main role is played by a centralized controller that provides to generate a specific flight path for each UAV to follow based on the currently available information. The UAVs move according to the flight paths received, but they are also able to gather sensorial information from the environment and send it back to the controller as it becomes available. As the controller receives new information, it may generate new flight paths that are in turn broadcast back to the UAVs [12][13];
- *reactive strategies*: rather than generate a specific flight path that requires live updates, this approach aims to generate a so-called "reactive strategy" for every UAV. This kind of strategy can be imagined as a single decision tree that controls the aircraft for the life of the mission. The decision tree determines changes in the UAV's heading, based on immediate low-level information collected from its sensors [14][15].

Richards and colleagues use a team of UAVs that has to explore a given area in a cooperative way, relying on a decision tree that implements a reactive strategy and controls the various aircraft developed through genetic programming (GP) methodologies. A more convenient approach might consist in the usage of evolutionary neural networks (NNs) [16][17], mainly for two reasons. First, it is easier to use neural networks instead of GP for this kind of task since there is no need to provide the MAVs with a predefined set of possible manoeuvres. Robotic aircraft endowed with a neural network controller can achieve a greater flexibility level, which in turn could allow them to employ innovative solutions (i.e., not expected by the experimenter) for the task they are carrying out. Second, if properly trained, neural networks can guarantee a much greater generalisation capability than a decision tree evolved through genetic programming. This can allow the MAVs evolved within a certain experimental setup to perform well in a different scenario

Nonetheless, in both GP and NNs cases, a computer simulation is required for reasons of cost and time. The strategies developed have first to be evaluated within a simulated environment, where the (potentially) thousands of strategy evaluations required to converge on effective solutions do not translate into real economic costs.

[1] Within this section we will use the term UAV in a generic way, meaning any possible kind of unmanned aerial vehicle, including MAVs.

[2] In reality, Richards, Whitley and Beveridge classify these approaches in four different groups. For simplicity purposes, we limit our analysis to only three of these, excluding the "behaviour based controller systems".

In addition to being frequently used to control terrestrial robots, neural networks have recently started to be considered also in the field of underwater robotics [18]. Despite that, they have been only rarely used as controller systems for flying robots. The main exception encountered so far, reviewing the literature, consists in the work of Floreano and colleagues [19]. They employ fully autonomous MAV swarms, where each swarm's member acts as a signal repeater, in order to create a reliable communication infrastructure between human rescuers and base station working into areas hit by natural disasters. At the same time, Holland and collaborators [20][21] are studying how to employ neural networks as controllers for autonomous helicopters. Finally, in addition to evolutionary methods, other meaningful insights come from the work carried out within the Autonomous Flight System Laboratory at the University of Washington. Stressing the importance of using heterogeneous autonomous systems in place of traditional hierarchical structures, Rathbun and Capozzi [22] had developed an efficient path planning algorithm for situations where the UAVs need to modify their paths in order to avoid a number of other aircraft flying in their vicinity.

3 Simulation experiments

The simulation experiments that will be presented here concern the usage of MAV swarms in a distributed control perspective. These experiments focus on a particular task, specifically a "search and destroy" scenario in the context of security and urban counter-terrorism. The main idea is that, given the MAVs' small size and their high level of maneuverability, they could be successfully employed on counter-terrorism operations to be carried out within urban environments.

To clarify the scenario, let us imagine being in the presence of a potential "danger", such as a terrorist walking along the centre of a modern city[3] in order to reach the place of an already planned attack. We can roughly identify two main categories of possible countermoves to a menace of this kind:

- *direct approach*: blocking the attacker through the intervention of a security task-force. It might be extremely dangerous if the target is, for example, a kamikaze wearing an explosive belt, since he could instinctively react to a physical aggression by detonating himself;
- *indirect approach*: neutralizing the target by hitting him from a remote position. This is the typical action carried out by a team of snipers. The problem with this approach is the difficulty involved in passing unnoticed while deploying a large amount of snipers around a city.

Using MAV swarms it might be possible to avoid the main disadvantages related to both the direct and the indirect approaches. The fact that electrical-propelled flying robots are able to fly silently and out from the typical line of sight of a person allows them to remain unnoticed while reaching the target. Furthermore, unlike the employment of a sniper team, a MAVs swarm will also be able to eventually perform a non-lethal action. The outcome of neutralizing the target could in fact be pursued through a low-potential detonation, or using some chemical substances instead. Those chemical elements might be something able to block a device starter or to immobilize the target. Another possibility would be to drop a flashbang grenade against the target, in order to make it temporarily inoffensive and allowing in this way the intervention of a task force deployed in the vicinity.

The fundamental assumption made in the model described here is that the MAVs have to be always aware of the target's position. It should not be an unrealistic hypothesis, as we can easily imagine a satellite-based system that, while continuously monitoring the movements of the target, at the same time shares the gathered information related to its position with the various MAVs.

The specifications of the MAVs employed in these simulations (size, speed and autonomy) have been inspired by the WASP Block III, produced by the American manufacturer Aerovironment.

The experiments that will be showed in this paper constitute the first steps of an incremental set of simulations aimed to gradually move toward a realistic model of MAVs swarm cooperative behaviour[4].

3.1 The simulation model

The environment where the simulation takes place is a two-dimensional rectangular area - sized approximately 630x675 meters - representing a portion of London's Canary Wharf. A swarm is composed of four MAVs, with starting positions close to the rectangle's corners and facing the centre of the environment (with the addition of a certain amount of random noise from their starting orientation). A target is deployed somewhere into the area, occupying a random position that is always known to the swarm's members in terms of distance and relative angle.

The neural network controlling the MAV behaviour is characterized by a simple three-layered feed-forward architecture, detailed as follows:

- the input layer consists of four neurons. One of them is dedicated to receiving the sensorial input related to the distance that separates the MAV from the target; the other three are dedicated to

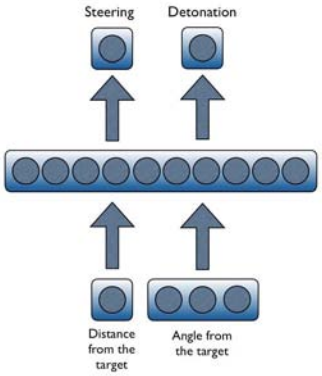
[3] The idea is to represent an urban scenario typical of a Western-like city, characterized by the presence of many high buildings, grouped in a semi-regular way, crossed by few huge roads.

[4] From a technical point of view, the simulator software has been written in C++, using the Qt libraries as graphical framework. The source code, the corresponding binary files (compiled both for Windows 32 bit and MacOS X 10.4/10.5) and some demos are available on line at the URL: http://www.tech.plym.ac.uk/research/SOC/abc/plymav/

encoding the relative angle between the two agents;

- the ten neurons belonging to the hidden layer are characterized by a tan-sigmoid activation function, with minimum value -1.0, maximum 1.0 and curve's slope 1.0;
- the output layer is composed of two neurons. One of them, continuous, is dedicated to the MAV steering. Its output value can vary between -1.0 and +1.0, according respectively to a 10° left and to a 10° right turn. The other neuron is a Boolean one: when it turns to 1 the MAV detonates (we suppose that, within the possible actions introduced in the previous paragraph, the MAVs of these simulations attack the target through a low-potential detonation).

Figure 1 - The neural network controller achitecture for Simulations A1, A2, A5 and A6



The MAVs' controller systems evolve using a genetic algorithm, through an evolutionary process lasting for 500 generations. An initial population of 100 different swarms is created with both connection weights and biases randomly assigned in the range -1.0/+1.0. Note that, even if each member is endowed with its own neural controller, the MAVs belonging to the same swarm share the same connection weights and the same biases as well: they are, in fact, clones of each other. Each swarm is tested four times within four different environments, which vary only for the target's position. Every test starts with the swarm's members deployed in their starting places, with the maximum amount of energy available (5,000 energy units[5]). Each MAV sequentially perceives its sensorial inputs, elaborates the appropriate behavioural response (steering amount and/or detonation) and actuates it at each time-step. The movement, which is in the new direction after steering has taken place, is 3.14 meters long and costs the aircraft 3.01 energy units. The test ends when the target has been destroyed by a MAV detonated close enough to it (2.2 meters or less) or when there are no more aircraft alive. Consider that a MAV – as well as detonating - can also die if it moves out of the environment boundaries, if it collides with a team-mate or if it finishes its autonomy.

The fitness formula through which the collective performance obtained by each swarm - after the conclusion of the four tests - is measured is:

$$fitness = -\alpha + (\frac{\beta}{50}) + (\sigma * 50) + (\phi * 5) \qquad (1)$$

where:

- α is the average distance between the target and the swarm's member exploded closest to it, calculated based on the four tests;
- β is the average amount of energy retained by the MAV detonated closest to the target, calculated based on the four tests;
- σ is the number of tests concluded by the given swarm with the elimination of the target;
- ϕ is the total number of swarm's members remained alive after the 4 tests (maximum 3 MAVs x 4 tests = 12 MAVs).

This formula tends to favour not only those swarms able to reach and destroy the target, but also the ones that are both quick in accomplishing the task and capable to performing it while losing the lowest number of MAVs possible.

The 20 swarms obtaining the best fitness score are selected for reproduction. Each of these swarms creates 5 copies of itself, which inherit its connection weights set, along with the biases related to the hidden and to the output layer (the input layer's neurons have not any bias). A certain amount of random mutation (ranging between -1.0 and +1.0) is added to each inherited weight and bias with probability .25. The elitism operator is also applied in order to preserve the unmodified reproduction of the swarm that - within a given generation – obtains the best performance. The best swarm creates five copies of itself, but just four of these are subject to random mutations

3.2 Simulations A: preliminary testing on an open environment

The first set of experiments aims to identify the most appropriate encoding for the sensorial input. Eight different simulations (A1-A8) have been carried out, testing various encodings and the related architectures.

A1: the angle that separates the MAV from the target (dependent on the current MAV's facing direction) is divided into eight different sub-fields. The first includes all the angles equal to or greater than 347.5° and lower than 22.5°; the second one is related to the angles between 22.5° (included) and 67.5° (excluded) and so on. These sub-fields are numbered progressively, according to a Boolean encoding, as shown in Figure 2A. The distance -

[5] Please consider that this value is much lower than the one that should correspond to the real WASP III's autonomy (it should be approximately 33,800). The decision to keep this value lower is justified by the long computational time required to carry out a simulation when the MAVs have the same autonomy as their real counterparts. Sometimes, in fact, and especially during the first generations, it might happen that some swarm's members move in the loop, without reacting to the variation in sensorial perception, until the autonomy finishes.

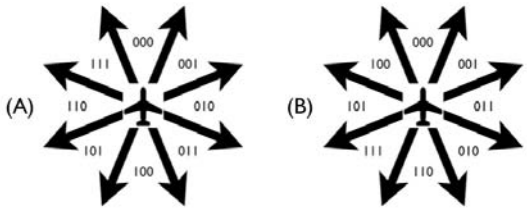
which in the simulated environment used can range from 0 to 1,165 meters - is discretized into 11 different values, according to Table 1.

Table 1 - Distance encoding for neural network architecture A1

Distance (meters)	Discretized value
1,165 <= distance < 1,008	0.0
1,008 <= distance < 896	0.1
896 <= distance < 784	0.2
784 <= distance < 672	0.3
672 <= distance < 560	0.4
560 <= distance < 448	0.5
448 <= distance < 336	0.6
336 <= distance < 224	0.7
224 <= distance < 112	0.8
112 <= distance < 2.2	0.9
2.2 <= distance <= 0	1.0

A2: the distance-related encoding is the same as A1, but the angle that separates the MAV from the target is encoded in a different way. It is still segmented in eight different parts but these are now numbered through a Gray Code encoding instead of a Boolean one (see Figure 2B).

Figure 2 - The angle encondings for neural network architecture A1 (A) and A2 (B) respectively



A3: the neural network architecture used differs from the two previously seen, since it exploits only two continuous neurons to encode the angle between the MAV and the target. Given an angle between 0° and 360°, the first neuron encodes its sin, while the second encode its cosine instead. The distance is encoded as in A1 and A2.

A4: this neural network architecture uses only a single continuous neuron to encode the angle that separates the MAV from the target. The angle value (0°→360°) is normalized in the range [0;1]. In order to avoid a very different encoding between a couple of angles both near the front of the MAV, before the normalization the angle is rotated to ensure the 0 stays behind the MAV. The distance is encoded as in A1, A2 and A3.

A5, A6, A7, A8: all of these architectures encode the distance between the MAV and the target in the same way. It is not discretized as it was in the first four simulations, but simply reduced into the range [0, 1], where 0 corresponds to the maximum distance (1,165 meters) and 1 to the minimum one (0 meters). The angle is encoded as follows:

- Simulation *A5*: encoded as in Simulation A1;
- Simulation *A6*: encoded as in Simulation A2;
- Simulation *A7*: encoded as in Simulation A3;
- Simulation *A8*: encoded as in Simulation A4.

3.2.1 Simulations A: results

The results from the first set of simulations, summarized in Table 2, clearly identify the A2 as the combination of neural network architecture and input encoding that generates the best performance.

Table 2 - Average fitness and percentage of tests concluded with the elimination of the target for Simulations A. The values are the average of the last 10 generations, based on 5 seeds

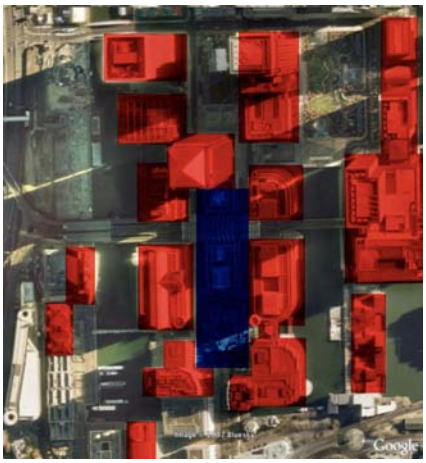
Simulation	Average fitness	Percentage of tests concluded successfully
A1	110.49	75.09
A2	315.18	93.46
A3	-152.46	11.68
A4	55.32	58.14
A5	111.66	75.33
A6	240.19	88.14
A7	-142.20	10.89
A8	-287.03	6.97

In general it is possible to see how the discretization of the sensorial inputs dramatically helps the controllers to evolve toward optimal solutions.

3.3 Simulations B: environment with obstacles

In the second set of simulations we have inserted some obstacles into the environment, with the intent of representing the biggest buildings present in the urban area we are using as a model. For simplicity, the new simulated environment is still two-dimensional. The buildings represent for the MAVs a kind of "no-fly zone": if they try to enter these areas, they will be immediately destroyed. In other words, we are assuming that these buildings are too high to allow the MAVs to fly over them, so their only chance to avoid these obstacles is through circumnavigation.

Figure 3. The environment where simulations B take place, with the main buildings mapped (in red) as no-fly zones. The highlighted zone, in the centre of the scenario, is what we define as the "enclosed area" [The background image has been taken from Google Earth©]



According to Figure 3, 19 building/obstacles have been mapped into the environment. Their shape is always rectangular and the differences between them are just related to their size. These obstacles correspond to the tallest buildings present in Canary Wharf.

In order to evolve an obstacle-avoidance capability, the MAVs have to be equipped with a sensor (or a set of sensors) able to detect the presence of any obstructions. We use an ultra-sonic sensor capable of detecting the presence of an object, no matter what kind of object it is (just a surface that can reflect the signal is enough) situated in front of the MAV - along a straight line - until 25 meters distance. In the context of our simulations, the obstacles that the aircraft are able to perceive through this sensor consist of: a building, the target and another MAV.

Starting from the A2 architecture, we have added one neuron to the input layer and two to the hidden layer. The behaviour of the new input neuron is straightforward, as it simply encodes the distance from the nearest obstacle perceived by the MAV, according to Table 3.

Table 3. Econding related to the ultra-sonic sensor perception

Distance (meters)	Discretized value
1165 <= distance < 33.6	0.0
33.6 <= distance < 30.24	0.1
30.24 <= distance < 26.88	0.2
26.88 <= distance < 23.52	0.3
23.52 <= distance < 20.16	0.4
20.16 <= distance < 16.8	0.5
16.8 <= distance < 13.44	0.6
13.44 <= distance < 10.08	0.7
10.08 <= distance < 6.72	0.8
6.72 <= distance < 3.36	0.9
3.36 <= distance <= 0	1.0

One of the main difficulties of this task arises from the fact that, using a sensor of this kind, the MAVs can simply detect a generic obstacle in front of them without the knowledge of what kind of obstacle it is. They are not aware, in fact, of the real nature of the obstacle they are facing, i.e. if it is, for example, a building or the target they are looking for instead. Therefore they need to make an assumption about what the perceived obstacle is, based on the others sensorial information gathered. In other words, they have to identify the target matching the information provided by the ultrasonic sensor with the distance from the target that they receive in turn from another sensor.

From a technical perspective the simulation has been subject to only few minor changes with respect to the A-series. First of all, the fitness formula (1) has been modified and now - in order to assign a strong importance to the obstacle-avoidance issue - it stresses the factor ϕ more than before:

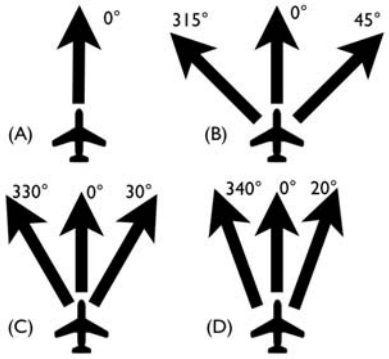
$$fitness = -\alpha + (\frac{\beta}{50}) + (\sigma * 50) + (\phi * 10) \quad (2)$$

To achieve a good generalization ability, the MAVs no longer start from the usual positions close to the environment's corners, but from a semi-random starting point instead. The distance from the boundaries is fixed (one MAV per side of the environment), while the exact starting position is randomly assigned before every test. Furthermore, during the four tests, the target is deployed two times in a random position into an "enclosed area" at the centre of the environment, and the other two times in a random position outside the enclosed area.

In order to make the navigation task easier, the MAVs' turning radius has been increased from +/-10° to +/-20°. This modification has been partially mitigated by a second change: now - during each step - the MAVs no longer make a movement 3.14 meters long, but a movement of 2.24 meters' length instead. The amount of energy spent for a single step has consequentially been reduced from 3.01 to 2.14.

Also, since the new behaviour requested to the MAVs is more complicated than the previous one, the evolution length has been increased from 500 to 2,000 generations.

Figure 4. Ultra-sonic sensors set up for Simulations B (A: architecture B1; B: B2; C:B3; D:B4)



The experimental setup described here corresponds to Simulation B1. In order to pinpoint the most efficient configuration able to evolve obstacle-avoidance capability, three other simulations have been carried out with different sensors set up. As detailed in Figure 4, simulations B2, B3 and B4 all use three ultra-sonic sensors (with the respective input neurons) instead of only one. The corresponding neural network architectures are slightly different with respect to the one used in B1, since a larger amount of hidden neurons (15 instead than 12) has been inserted on them.

3.3.1 Simulations B: results

The results coming out from the second set of simulations need to be analyzed thoroughly.

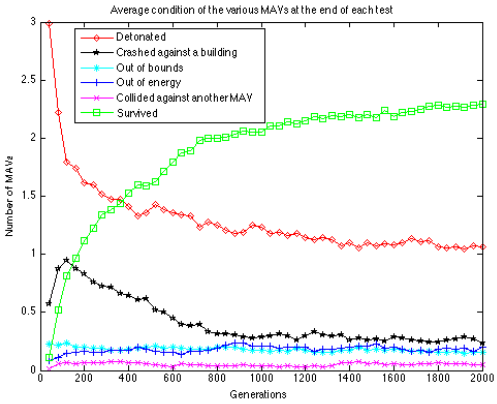
Table 4. Average fitness, percentage of tests concluded successfully and average minimum distance from the target for Simulations B. The values are the average of the last 10 generations, based on 5 seeds

Sim.	Average fitness	Percentage of tests succeeded	Minimum distance from the target
B1	51.09	62.13	0.67
B2	257.27	87.07	0.14
B3	238.09	84.5	0.44
B4	257.62	87.18	0.54

What is striking is that Simulation B1 is the one which performs worst in the overall group. This outcome was to some extent expected, since the presence of only one ultra-sonic sensor does not allow the MAVs to know in which direction to turn when facing an obstacle. Aircraft evolved in this experimental setup are able to reach and neutralize the target only 62% of the times, while, on average, one aircraft per test crashes against a building.

Simulations B2, B3 and B4 look much more promising. All the members of this subsection score a good result in terms of percentage of tests successfully concluded (respectively 87%, 84% and 87%). The relative bad performance of B3 is due to the fact that the obstacle-avoidance ability of the MAVs evolved in this setup is less effective with respect to Simulations B2 and B4. To clarify this point, consider that on the "average swarm" of the population evolved in Simulation B3, there is (again, on average) one MAV that crashes against a building every three tests, versus one each four tests for Simulation B3 and one each five tests for Simulation B4). Furthermore, the MAVs - particularly when enclosed within restricted areas surrounded by obstacles - can sometimes get stuck in a kind of loop. Basically they continue to turn on themselves, until their available energy goes out. This condition happens much more frequently in Simulation B3 than in Simulations B2 and B4.

Figure 5. The end-test condition for the average swarms of Simulation B4



Looking at Table 4, the results obtained by B2 and B4 appear very similar. The last column - "Minimum distance from the target" - could look like a bad score for Simulation B4, but we have to consider that, in order to destroy the target, a MAV needs to detonate within a two-pixel distance from it. It does not make a big difference having an average minimum distance from the target, for the swarm's member detonated closest to it, of 0.1394 or 0.5426 instead.

Observing the data contained in Table 5, related to the conditions of the "average swarm" at the end of the "average test", the advantages of Simulation B4 become evident. With respect to Simulation B2, in fact, there are a wider number of MAVs alive (2.3 vs. 2.17), due to fewer detonations (1.03 vs. 1.05) and to a better obstacle-avoidance capability (on average, 0.22 MAVs crash against a building vs. 0.32). The only pitfall is the average amount of MAVs that finish their energy during the test: 0.23 for B4 vs. 0.18 for B2.

Table 5. End-test conditions for the Simulations B's "average swarm"

Sim.	Alive	Detonated	Crashed against a building	Out of energy	Other death
B1	1.18	1.22	0.99	0.32	0.29
B2	2.17	1.05	0.32	0.18	0.28
B3	2.2	1.03	0.28	0.27	0.22
B4	2.3	1.03	0.22	0.23	0.22

Simulation B4 can be anyhow declared the best between all the "Family B" simulations carried out.

4 Conclusion and further developments

In the simulations described here we have identified the minimum set of sensors, with the respective encodings, needed to evolve neural network controllers for autonomous MAV swarms able to navigate along an unknown environment, with or without obstacles, and to perform a pre-defined action when a certain target has been reached. This work constitutes a baseline framework that will act as a solid starting point for the future studies regarding the employment of MAV swarms in different kinds of tasks. Particularly, the main research directions that will be followed during the next years are two.

Sociality and cooperative tasks. The task presented here, even if conducted by a swarm composed of many members, could not be fully classified as a cooperative task. The MAVs, in fact, act individually, interacting with the teammates only through their ultra-sonic perception when it happens that two or more of them are flying over the same area. The next step will be to make the task a social one, e.g. requiring a coordinated attack (two or more MAVs that detonate simultaneously) in order to neutralize the target. To successfully accomplish such a kind of task, the swarm's members need at least to know the position of their teammates.

Evolution of communication. From the previous point easily arise the awareness that the usage of explicit forms of communications between the MAVs could dramatically improve their performance in accomplishing a cooperative task. Most MAS models that have considered communication typically refer to implicit forms of communication, such as visual cues in predator models [8] and stigmergy communication in colonies [7], or to the technical aspects of agent communication protocols [5]. Instead, the use of explicit forms of communication (e.g. symbolic, language-like systems) can be crucial in tasks requiring higher levels of cognitive capabilities, such as planning and decision making, and for the integration of language and cognitive capabilities [23][24][25]. By explicit forms of communication we mean the use of symbolic lexicons in which it is possible

to identify a clear symbol/meaning relationship grounded on the agents' collaborative task properties and processes. New studies on the role of explicit communication in MAS have many theoretical and technological implications. In particular, agents that are allowed to communicate explicitly during the execution of collaborative tasks might benefit from the exchange of information regarding properties of the task being processed. Such explicit communication systems do not have to be defined a priory by the human designer, but can autonomously emerge from social interaction between agents [26][27].

At the same time, the simulator will gradually evolve toward a more realistic three-dimensional model, able to take into account the physical properties of the actors involved in the simulations. This, in fact, will constitute the last step required in order to think of the usage of MAVs, endowed with genetically evolved neural network controllers, within real applicative scenarios.

Acknowledgments

Effort sponsored by the Air Force Office of Scientific Research, Air Force Material Command, USAF, under grant number FA8655-07-1-3075. The U.S. Government is authorized to reproduce and distribute reprints for Government purpose notwithstanding any copyright notation thereon.
The authors would also thank euCognition.org for the support provided (Network Action NA097-3).

Disclaimer

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

References

[1] C.W. Reynolds, *Flocks, Herds, and Schools: A Distributed Behavioral Model*, Computer Graphics, Vol. 21-4 (SIGGRAPH '87 Conference Proceedings), pp. 25-34, 1987.

[2] G. Nitschke, *Emergence of Cooperation: State of the Art*, Artificial Life, Vol. 11-3, pp. 367-396, 2005.

[3] G. Baldassarre, D. Parisi and S. Nolfi, *Distributed Coordination of Simulated Robots Based on Self-Organization*, Artificial Life, Vol. 12-3, pp. 289-311, 2006.

[4] J.M. Eklund, J.S. Sprinkle and S. Sastry, *Template Based Planning and Distributed Control for Networks of Unmanned Underwater Vehicles*, 44th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC'05), 2005.

[5] R. Weiss (Ed.), *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, MIT Press, 2000.

[6] M. Koes, I. Nourbakhsh and K. Sycara (2006) *Constraint Optimization Coordination Architecture for Search and Rescue robotics*, Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) 2006, pp. 3977-3982, 2006;

[7] V. Trianni and M. Dorigo, *Self-organization and Communication in Groups of Simulated and Physical Robots*, Biological Cybernetics, Vol. 95-3, pp. 213-231, 2006.

[8] A.M. Barry and H. Dalrymple-Smith, *Visual Communication and Social Structure. The Group Predations of Lions*, Proceedings of International Workshop on the Modelling Natural Action Selection (MNAS05), 2005.

[9] S.A. Cambone, K.J. Krieg, P. Pace and L. Wells II, *Unmanned Aircraft Systems Roadmap 2005-2030*, http://uav.navair.navy.mil/roadmap05/USRoadmapAug%2005.pdf.

[10] M.D. Richards, D. Whitley and J.R. Beveridge, *Evolving Cooperative Strategies for UAV Teams*. Genetic and Evolutionary Computation Conference (GECCO 2005), ACM Press, 2005.

[11] V. Ablavsky, D. Stouch and M. Snorrason, *Search Path Optimization for UAVs using Stochastic Sampling with Abstract Pattern Descriptors*, Proceedings of the AIAA Guidance Navigation and Control Conference, 2003.

[12] S. Rathinam, M. Zennaro, T. Mak and R. Sengupta, *An Architecture for UAV Team Control*, IAV2004: Fifth IFAC symposium on intelligent autonomous vehicles, 2004.

[13] P. Vincent and I. Rubin, *A Framework and Analysis for Cooperative Search using UAV Swarms*, Proceedings of the 2004 ACM symposium on Applied computing, pp. 79-86, 2004.

[14] F.W. Moore, *A Methodology for Missile Countermeasures Optimization under Uncertainty*, Evolutionary Computation, Vol. 10-2, pp. 129-149, 2002.

[15] G.J. Barlow, C.K. Oh and E. Grant, *Incremental Evolution of Autonomous Controllers for Unmanned Aerial Vehicles using Multi-Objective Genetic Programming*, Proceedings of the 2004 IEEE Conference on Cybernetics and Intelligent Systems, 2004.

[16] S. Nolfi and D. Floreano, *Evolutionary Robotics. The Biology, Intelligence, and Technology of Self-Organizing Machines*, MIT Press, 2004.

[17] M. Mitchell, *An Introduction to Genetic Algorithms*, MIT Press, 1998.

[18] V.S. Kodogiannis, *Neuro-Control of Unmanned Underwater Vehicles*, International Journal of Systems Science, Vol. 37-3, pp. 149-162, 2006.

[19] D. Floreano, S. Hauert, S. Leven and J.-C. Zufferey, *Evolutionary Swarms of Flying Robots*, International Symposium on Flying Insects and Robots, pp. 35-36, 2007.

[20] O. Holland, J. Woods, R. De Nardi and A. Clark, *Beyond Swarm Intelligence: the Ultraswarm*, IEEE Swarm Intelligence Symposium (SIS2005), 2005.

[21] R. De Nardi, O. Holland, J. Woods and A. Clark, *SwarMAV: A Swarm of Miniature Aerial Vehicles*, 21st Bristol UAV Systems Conference, 2006.

[22] D. Rathbun and B. Capozzi, *An Evolution Based Path Planning Algorithm for Autonomous Motion of a UAV through Uncertain Environments*, Proceedings of the AIAA Digital Avionics Systems Conference, 2002.

[23] L.I. Perlovsky, *Integrating Language and Cognition*, IEEE Connections, Vol. 2-2, pp. 8-13, 2004.

[24] L.I. Perlovsky, *Neural Networks, Fuzzy Models and Dynamic Logic*, Chapter in R. Kohler and A. Mehler (Ed.), Aspect of Automatic Text Analysis (Festschrift in Honor of Burghard Rieger), Springer, Germany, pp. 363-383, 2006.

[25] V. Tikhanoff, J.F. Fontanari, A. Cangelosi and L.I. Perlovsky, *Language and Cognition Integration through Modeling Field Theory: Category Formation for Symbol Grounding*, Proceedings of ICANN06 International Conference on Artificial Neural Networks, 2006.

[26] A. Cangelosi and D. Parisi, *Simulating the Evolution of Language*, Springer-Verlag, 2001.

[27] D. Marocco and S. Nolfi, *Emergence of Communication in Embodied Agents Evolved for the Ability to Solve a Collective Navigation Problem*, Connection Sciences, (in press).