How Complexity Pervades Specialization in Canonical Embodied Evolution

Pedro Trueba[1], Abraham Prieto[1], Francisco Bellas[1] and Richard J. Duro[1]

[1]Integrated Group for Engineering Research
Universidade da Coruña, Spain
francisco.bellas@udc.es

Abstract

Embodied Evolution (EE) is an evolutionary strategy based on natural evolution in which the individuals that make up the population are embodied and situated in an environment where they interact in a local, decentralized and asynchronous fashion. It has been successfully applied in collective problems showing its validity to perform on-line evolution both in simulated and real agents. A key feature of EE is that of emergent specialization, that is, this strategy is able to autonomously generate a distribution of individuals into species if that is advantageous in the scenario. This paper goes in the line of studying such feature in more depth, analyzing how the complexity of the task (fitness landscape) and the complexity of the individuals (control system) affect the emergence of specialization. The analysis is carried out using a canonical EE algorithm in a real problem consisting in a collective surveillance task with simulated Micro Aerial Vehicles.

Introduction

As research has advanced in the field of evolutionary optimization, new approaches and techniques have allowed researchers to address even more complex problems. In this sense, a remarkable challenge is that of solving real-world dynamic problems in the absence of centralized and updated information. This type of problem appears in tasks like routing, surveillance, resource assignment, etc.

Some of the most successful approaches in this line are based on multi-agent systems that exploit the coordination between the agents to provide a collective solution to the problem (Hanna, 2009) (Rinde, 2012). This way, each agent is assumed to have decentralized and out-of-date information and, with it, it must handle its small part of the problem in real-time. The effort of the evolutionary algorithm is on finding a global coordinated solution to the problem as an aggregation of the partial ones, which can be really complex in dynamic environments.

With the aim of dealing with a more flexible and general search process, some authors have included the behavioral specialization of the agents as a new dimension of the collective problem. That is, the agents can be heterogeneous in operation, so specialists can emerge from evolution if they are beneficial for the task. It could seem that this new dimension increases the complexity of the search space, but it has been shown that allowing such flexibility simpler agents can emerge, which leads to a general simplification of the problem. It must be pointed out here that in this type of heterogeneous collective evolution, the solution to the problem is provided by the concurrent execution of the whole population, and not by a replication of the best individual.

The most remarkable evolutionary strategies one can find in heterogeneous collective optimization are Cooperative Coevolution Evolutionary Algorithms (CCEA) (Wiegand, 2003) (Panait, 2010). In this type of algorithms, the control system of each agent, typically an Artificial Neural Network, is evolved in an independent population, although the fitness of each individual is obtained by their joint execution with their team. Thus, if the solution is made up of n components, a CCEA evolves n populations, each one containing the genotype that will define the response of each component. Specific types of CCEAs like SANE (Gomez, 1997), Multi-agent ESP (Yong, 1999), CONE (Nitschke, 2012) or Hyb-CCEA (Gomes, 2015) have been widely applied with success in collective optimization of problems. The main problem of CCEAs for solving real-world dynamic problems is that they must run off-line due to their high computational requirements. Although the evolutionary processes can be executed in parallel, they must be serialized for evaluation, which must be performed several times in order to provide enough combinations of genotypes to achieve a reliable evaluation.

An online version of CCEA is the evolutionary strategy known as Embodied Evolution (EE). It was created by Ficici and Watson in 1999, inspired by Artificial Life experiments, with the aim of speeding up online evolution (Ficici, 1999). The main difference with traditional CCEAs is that evolution is at the population level in EE, that is, each individual only carries its own genotype. Consequently, EE follows a natural evolution scheme in which the individuals that make up the population are embodied and situated in an environment where they interact in a local, decentralized and asynchronous fashion. This interaction is not driven by a preset synchronization mechanism as in traditional evolutionary algorithms, but by the result of the particular behaviors of the active individuals in the environment and their interactions with other active and passive elements within it (Schut, 2009). Evolution in EE is open-ended, leading to a paradigm that is intrinsically adaptive and highly suitable for real time learning in distributed dynamic problems.

In the last decade, EE interest has grown mainly in the field of multi-robot systems (Bredeche, 2012) (Elfwing, 2011) (Eiben, 2010). As a consequence, different algorithms have arisen, which share the same operational principles but differ in how they implement specific operators. During subsequent

years, some of those algorithms were successfully applied to different collective problems (Bredeche, 2012) (Elfwing, 2011) (Prieto, 2010) (Duro, 2011), which allowed the validation of the paradigm in practical terms, but also the lack of a formal characterization to be considered by researchers in the evolutionary computation field became clear. As a first step towards this standardization of EE, a canonical EE algorithm that isolates its operational principles from those of the particular implementations was presented and thoroughly studied in (Prieto, 2015).

The current paper follows the line of EE formal characterization. In this case we are interested in analyzing in depth one of the main features of EE: the emergence of specialists (Nitchske, 2008). As shown in (Prieto, 2015) and (Trueba, 2013), in EE the optimal number of species emerge as required by the problem. But, what features of the problem determine the emergence of specialists? Can we anticipate the species that will arise or understand the reason why a given organization has emerged? These two questions are very relevant when developing an optimization algorithm. It is obvious that the spatial or temporal separation between individuals promotes specialization because it avoids mating (Trueba, 2013), but there are several more features. As it is well-known in fields like Complex Systems (Mitchell, 2008) and Ecology (Epstein, 1996), one of the most relevant factors is the complexity of the environment (Bonabeau, 1997) (Burbeck, 2007) and the complexity of the individuals (Anderson, 2001) (Detrain, 2002) that make up the population.

Thus, the question we aim to address here is how complexity affects the emergence of specialists in EE. We have analyzed this problem from two perspectives: varying the complexity of the environment where the agents are situated and varying the complexity of the agent itself, that is, its optimization capability. To carry out this analysis, we have applied the canonical EE algorithm in a collective surveillance task with simulated Micro Aerial Vehicles (MAVs), which is a prototypic example of dynamic and decentralized problem that must be solved in real time.

Canonical Embodied Evolution Algorithm

The canonical EE algorithm is explained with detail in (Prieto, 2015), and here we will provide just a brief summary of its main parameters and operation. During its development we decided to simplify and extract the barebones specification of a generalized EE algorithm in terms of as few parameters as possible. This generalization has been achieved by substituting the activation of particular operators triggered by events produced in the real problem with probability distributions, which are not task dependent. A second objective in the definition of a canonical EE algorithm was to do away with the bonds the environment imposes on the structure and operation of this type of situated algorithm. The adaptation to the environment determines the type of tasks that are considered tractable. It also makes the algorithm and its behavior even more task dependent, and as a result, it becomes more complicated to extract general conclusions from experiments. Consequently, the circumstantial/spatial interactions have been replaced in the canonical algorithm by stochastic variables, which follow probability functions. The canonical EE pseudo-code is the following:

```
1   While simulation active do
2   Random creation of population
3   For each interaction
4     For each individual
5       Assign fitness – (Scenario interaction)
6       If random < P_mating
7         Look for fertile partners (P_selection)
8         Select a partner for recombination (P_elegibility)
9         Generate offspring and store it (P_ls)
10      If random < P_replacement
11        New individual ← current offspring genotype
11        Reset individual
13  End
```

The canonical EE performs three basic processes: evaluation (line 5 in the pseudo-code), mating (lines 6 to 9) and replacement (lines 10 to 12). The processes and parameters that define the algorithm response are:

- *Mating selection*: it has been modeled as an event that is triggered by a uniform probability function that depends on a single parameter, the probability of mating for every time step (P_{mating}). This probability can be calculated based on the maximum number of mating one individual can perform, which can be assimilated to a "maximum tournament window size" (S_{max}), and on the maximum lifetime of an individual (T_{max}), which is the same for every individual:

$$P_{mating} = \frac{S_{max}}{T_{max}}$$

- *Selection policy:* the probability of being eligible as a candidate for mating ($P_{elegibility}$) is defined through a function (ψ_e) that is based on three different criteria: the genotypic distance (φ_{cand}), a distance measure between certain status parameters (P_{cand}) which vary during evaluation time due to both the phenotype of each individual and environmental circumstances (for example, position in a geometric space) and the fitness value (γ_{cand}):

$$P_{eligibility} = \psi_e(\varphi_{cand}, \gamma_{cand}, [P_{cand}])$$

- *Genotypic recombination*: as in natural evolution, two main recombination operators can be distinguished: mutation and crossover. In order to characterize this genotypic recombination in a general and simple way, a new intrinsic parameter is defined: the probability of using a local search strategy (P_{ls}), that is, a mutation operator. It is a measure of the exploration and exploitation balance through the ratio between crossover and mutation frequency.
- *Replacement*: the replacement process is modeled here as triggered by a replacement probability ($P_{replacement}$) and it is defined based on a more intuitive and manageable parameter, which is the life expectancy (T_{exp}):

$$P_{replacement} = \frac{1}{T_{exp}}$$

The life expectancy is defined for each individual in each time step based on its current fitness (Q_i), which depends on its genotype and the genotypes of the others. Specifically, a piecewise function has been defined to assign the life expectancy (T_{exp}) for any fitness value using a linear model. This function has two pieces, one for individuals with low fitness ($Q_i < 2/3Q_{max}$) and another for individuals with high fitness ($Q_i > 2/3Q_{max}$):

$$T_{exp} = T_{mat} + \frac{Q_i C_m (T_{max} - T_{mat})}{2/3 Q_{max}} \text{ if } Q_i \le \frac{2Q_{max}}{3}$$

$$T_{exp} = T_{mat} + C_m (T_{max} - T_{mat}) + \frac{(1 - C_m (T_{max} - T_{mat}))}{1/3 Q_{max}} (Q_i - 2/3 Q_{max}) \text{ if } Q_i > \frac{2Q_{max}}{3}$$

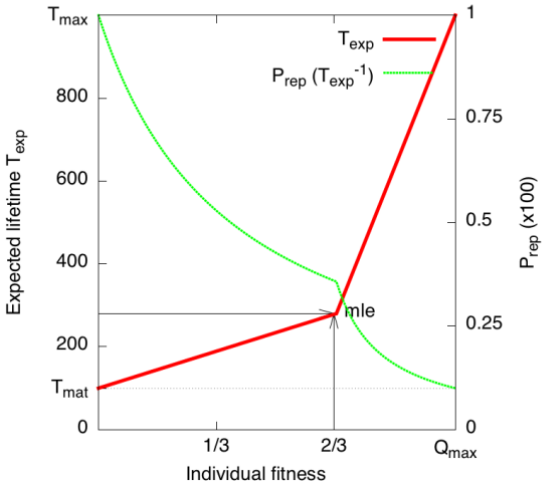

Fig. 1. Graphical representation of the function that parameterizes replacement

The use of this function to model replacement allows covering a broad range of different replacement policies, ranging from those with synchronous replacement ($T_{mat} = T_{max}$ and $C_m = 1$) to those with a strongly exploitative operation through elitism ($T_{mat} = 1$ and $C_m = 0$) by way of any intermediate combination. As it can be observed in figure 1, the relation between fitness and life expectancy has been modeled by means of four fixed parameters:

1. *Maturation time (T_{mat}):* it is defined here as a percentage of T_{max} and it sets the minimum T_{exp} for an individual and, as a consequence it captures the sum of two time periods, the minimum time required to reliably evaluate an individual (1 in our case) and the life-time for those with zero fitness. During the maturation time the individual cannot be replaced.
2. *Maximum current fitness (Q_{max}):* since the maximum fitness one individual can achieve for a specific scenario is not known beforehand, this value represents the maximum known fitness so far, and it is used to calculate a relative fitness $Qr = Q_i/Q_{max}$.
3. *Mediocrity coefficient (C_m):* it establishes the expected lifetime for the individuals with a relative fitness value equal to 2/3 ($Q_r = Q_i/Q_{max} = 2/3$) of the maximum current fitness (it can be seen as the expected lifetime of the mediocre individuals). This parameter ranges from 0 (meaning their life expectancy will equate T_{mat}) to 1 (meaning it will equate T_{max}).
4. *The maximum lifetime (T_{max}):* it is the maximum number of time steps a chromosome can participate in the evolution. It is assigned to individuals with their fitness equal to the maximum current fitness.

As a summary, six intrinsic parameters and an eligibility function have been defined to encompass and generalize the operation of a general EE algorithm: T_{max}, S_{max}, P_{ls}, Q_{max}, T_{mat}, C_m and ψ_e. See (Prieto, 2015) for a sensitivity analysis of the canonical EE algorithm.

To apply the canonical EE algorithm to real problems, two operators have to be adapted to the constraints the scenario imposes, and therefore, their dependence on the task is unavoidable: the mating operator, which is constrained by the communication limitations of the scenario and individuals to exchange their genetic codes, and the evaluation operator, which relies on the actual behavior and on the state of the scenario.

Collective Surveillance Task

To analyze how complexity affects the emergence of species using the canonical EE, we have designed a simulated environment in which a fleet of Micro Aerial Vehicles (MAVs) has to collectively survey an indoor scenario where there is not centralized information available. To do it properly, the MAVs need to locate themselves to keep track of their trajectories and to share this information with other robots. The determination of their positions will be performed using their IMU, artificial landmarks that can be sensed using the onboard camera, and the position of other MAVs in sight. The control of each of the MAVs is provided by an Artificial Neural Network (ANN), and the parameters of this ANN will be adjusted using the canonical EE algorithm. Thus, EE is in charge of organizing the MAVs in the scenario in order to increase the accuracy of the fleet location, and consequently, the speed at which a new point of interest is reached.

Experiment description

The experimental setup has been defined in simulation, based on a real indoor gathering task performed by MAVs as a previous step to translate the algorithm or directly the controllers to a fleet of real MAVs. The specific MAV that has been modeled is the Parrot ARDrone 2.0, a very popular general-purpose commercial quadcopter. Indoor navigation is performed then by simulated ARDrones that, as in the case of the real ones, are provided with an IMU and a camera to perform autonomous positioning. The most important aspect of the simulation is the model of the response of the location sensors when a certain maneuver is carried out. Firstly, the IMU will provide some velocity signals for each degree of freedom, which has to be integrated to produce the estimated motion. The estimation of the velocity is modeled as subject

to a normal distribution centered on the real input velocity with its corresponding variance matrix as is frequently assumed on real navigation.

The model implemented for the artificial markers represents the use of the AprilTags created by The APRIL Robotics Laboratory at the University of Michigan (Olson, 2011) since it is also what we have used during our tests with the real ARDrones. Therefore, the location estimation provided by the markers is based on a real accuracy model which was produced in our laboratory using an ARDrone 2.0 and 40 cm long AprilTags, and that can be formulated as a function L_e which relates the variance of the estimation $Var(\vec{p}_{drone})$ with the relative distance ($\|\vec{p}_{drone} - \vec{p}_{tag}\|$) and orientation ($yaw_{drone:tag}$) between camera and tag:

$$Var(\vec{p}_{drone}) = \boldsymbol{L_e}(\|\vec{p}_{drone} - \vec{p}_{tag}\|, yaw_{drone:tag})$$

These tags (*permanent tags*) provide an absolute and potentially accurate position estimation, which does not degrade with time. In order to improve the performance of the navigation by improving the accuracy of the MAVs the same type of tags are attached to the body of the quadcopters (*mobile tags*), which will make up a hybrid location sensor. Therefore, the detection provided by a mobile tag still constitutes a direct location estimation but, unlike in the case of permanent tags, their accuracy is variable and it is modelled as proportional to the velocity of the MAV's. As a consequence, those MAVs which will serve as mobile tags will have to stay still to be able to provide location accuracy. The use of this type of mobile tags leads to accuracy becoming a resource that MAVs can get and share to be able to slow down its degradation, and therefore, to accomplish their main task more efficiently. It could also allow some of the MAVs not to require visits to static tags, which are frequently non optimally located, in order to improve their location accuracy, being 'nourished' by mobile tags.

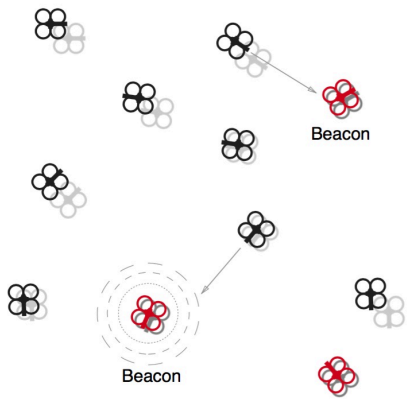

Fig. 2. Graphical representation of a portion of the scenario. The distance to each own shadow represents the current estimation error for the MAV. The red MAV represents a MAV acting as a mobile tag. The circles around the tags represent the different levels of accuracy provided by the tag.

Side of the arena (L)	768
Total area	L^2
Fixed tags detection range	L/4
Mobile tags detection range	L/16
Max velocity (V_{max})	L/50
Standard deviation for the velocity	V_{max}/50
Max accuracy provided by an April Tag	L/10

Table 1. Design parameters of the simulated scenario

The scenario was discretized to reduce computational effort as a 768 x 768 (square length units) non-toroidal square arena, which is provided with 4 fixed tags placed randomly. Figure 2 shows a schematic representation of a portion of the arena. Each MAV is associated to both a real and an estimated position. The former is shown with a solid color in the simulation and the later with a softened shadow of the MAV. The MAV has no idea of the real position, this is just an externally obtained value for display purposes. The further the distance between those positions, the higher the location estimation error and the lower the exploration level. Table 1 contains the specific parameters that define the scenario.

The final objective of the surveillance task is for the fleet of MAVs is to continuously cover the maximum possible area. In order to perform the search of unexplored areas, the MAVs can keep record of the areas they have already explored. This is stored in an *'exploration map'* carried by each MAV and that can also be shared with others when they meet. The exchange of this information allows the task to be solved cooperatively since it enables the distribution of the search among the group of MAVs. However, since the estimation of one's position has a varying accuracy, the updating of the exploration map has to take that into account. The exploration of a cell is modeled as an *exploration probability* (P_{ex}), which indicates the probability that a certain cell has of having been explored. This probability (P_{ex}) can be directly calculated as the ratio between the size of a cell (L_{cell}) and the location error range (E_{loc}) and is stored in the exploration map:

$$P_{ex} = \frac{{L_{cell}}^2}{\pi \cdot {E_{loc}}^2}$$

Subsequent MAVs must decide whether or not to *re-explore* that cell based on the guaranteed exploration probability P_{ex}. Therefore, the collaborative exploration map is the only information that a MAV gets from the scenario about the surveillance process. The individual fitness for each agent increases each time it covers and unexplored cell. The global fitness for the multi-agent system, which must be optimized by the canonical EE algorithm, is the sum of exploration probability for all the cells j in the scenario:

$$G = \sum_j P_{exj}$$

It must be highlighted that the MAVs are not transparent to each other and can collide with others and with the walls. When a MAV collides, it loses part of its location accuracy. To avoid collisions, they are provided with an obstacle sensor which mimics an infrared ring that detects near obstacles.

Individual encoding

The control architecture for each agent is a multilayer perceptron ANN with three layers: one input layer with three neurons, one hidden layer with a configurable number of neurons and one output layer with one neuron. The first input is the *current exploration capability* of the agent, which measures the exploration capability and it is based on its location error. The second input provides the *maximum attainable exploration* in the surrounding areas and the direction towards it. The third input provides the *distance to the closest obstacle* (up to the sensing range of the sensor) and the direction towards it. The size of the intermediate layer will be varied for different experimental configurations to analyze the complexity of the behavior of the agent, from 1 neuron in the simplest case to 10 neurons in the one with highest complexity. The output of the neural network modulates the behavior of the MAV between four pre-learned basic behaviors, namely: *move towards the most unexplored area detected*, *move towards the nearest tag*, *move against the closest obstacle* and *stay still to serve as a dynamic tag*. These behaviors are individually selected for each time step, the frequency with which each agent will display each behavior will determine the agent species.

Iterations	**20.000**
Population size	**40**
Maximum lifetime (T_{max})	**1000**
Maturity time (T_{mat})	**25**
Selection criteria (ψ_e)	**Higher fitness (γ_{cand})**
Tournament selection size (S_{max})	**40**
Local search probability (P_{ls})	**1.0**
Mediocrity coefficient (C_m)	**1.0**
Maximum current fitness (Q_{max})	**Automatic**
Chromosome length	**{4,8,40} x [0,1]**

Table 2. Parameters of the canonical EE algorithm

The implementation of the canonical EE algorithm used here follows the pseudocode presented in section 2, while the specific values for the parameters are those shown on Table 2. These values were selected according to the conclusions extracted from (Prieto, 2015).

Results

This section will show the results obtained for different configurations of the scenario in order test how different variations both in the complexity of the individuals and in the complexity of the environment affect the outcome of the evolution. The complexity of the individuals has been modified by changing the size of the ANN which controls the MAV. The complexity of the environment has been changed in two ways. First, by including more or less permanent tags, which will make mobile tags more necessary (or indispensable if there isn't any permanent tag) as the number of permanent tags decreases. Second, the environment is also adjusted by modifying the impact of the collisions on the degradation of the accuracy of the MAVs, which will make their navigation more or less complex (from neglecting the obstacle sensor to rigorously avoiding obstacles).

To clarify the canonical EE response in this type of collective optimization problem, we will describe first the emergence of specialization in a representative run performed using the following experimental configuration: no permanent tags, an ANN with 4 weights (3x1x1) and no penalty for collisions. The top plot of figure 3 shows the evolution of the global fitness of the population during 20000 iterations while the bottom one displays the average number of species per iteration (N_s). This number of species was calculated using the metric defined by the Davies-Bouldin Index (DBI) (Davies, 1979) applied to a k-means clustering algorithm. Since that metric does not consider the existence of a single cluster, an auxiliary species of ten individuals is always included in the data to be clustered. The number of species (N_s) is obtained using the average of each possible number of clusters (from 1 to 10) weighted by the inverse of the DBI associated to them (all possible numbers of clusters), so that the strong discretization of the low number of species is avoided and the measurement is more accurate. To account for the inclusion of an auxiliary species the final result is obtained by subtracting one from the weighted average, that is:

$$N_s = \left(\sum_{i=1}^{10} i * DBI(i)^{-\gamma} \right) - 1 \qquad with\ \gamma = 2$$

The parameter γ provides a smoothing coefficient, the lower the coefficient the more discrete the result.

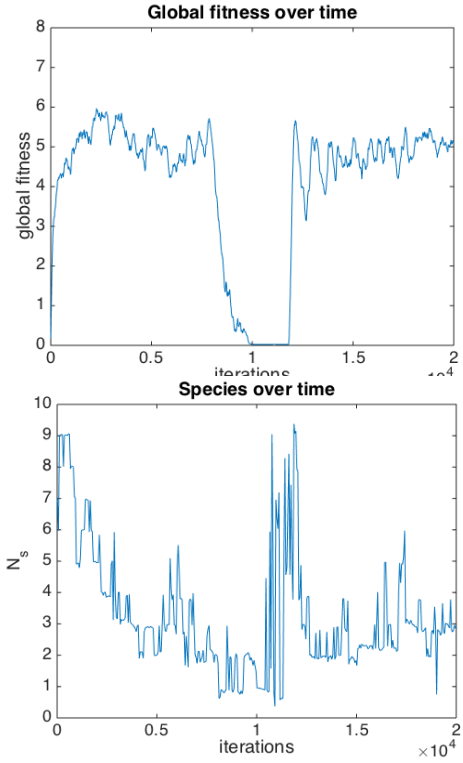

Fig. 3. Evolution of the global performance of the population (top) and average number of species (bottom) during 20000 evolution steps

As it can be observed in the top plot of figure 3, in this run, the population starts with low performance but it quickly achieves a successful global fitness level (less than 300 iterations). As demonstrated in several previous works (Prieto, 2010) (Duro, 2011), a remarkable strength of EE is the achievement of satisfactory results after few time steps. The number of species also converges to a value around 2, which can be observed in the bottom plot of figure 3.

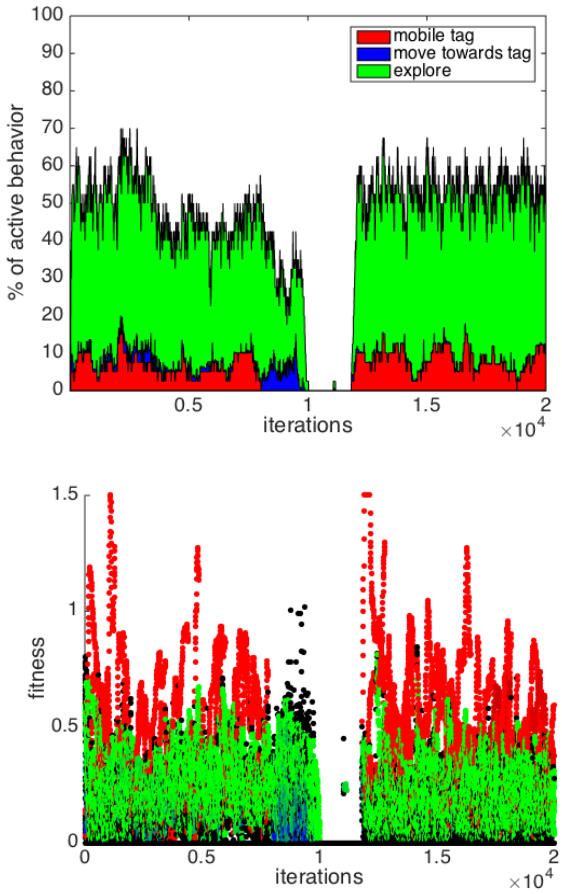

Fig. 4. Frequency of activation of each pre-learned behavior for the whole population (top) and individual fitness of each of the 40 individuals of the population (bottom)

The top plot of figure 4 shows the frequency of activation of each pre-learned behavior for the whole population, and the bottom one shows the individual fitness of each of the 40 individuals of the population. The coloring code is the same for all the plots in figure 4 and figure 5, which depicts individual parameters: red for those individuals with predominant *become mobile tag* behavior, green for *exploring the surrounding area*, blue for *moving towards the closer tag* and yellow for *avoiding obstacles* (although not present in this run since the penalty is set to zero). It can be seen in the top plot of figure 4 that up to iteration 8000, two main species have emerged: become *mobile tag* (red) and *exploring* the surrounding area (green). As shown in the bottom plot, being a mobile tag provides a higher individual fitness to the agents.

After around 8000 iterations, a period starts in which the population gets destabilized, which is something that happens often in a co-evolutionary process, and the global performance drops quickly (see figure 3 top). This run was indeed selected to illustrate both the dynamism of this type of evolution and the capabilities of the algorithm to recover from this period to achieve again a successful performance. As shown in the top plot of figure 4, in this unstable period, the number of species decreases to almost one since the *mobile tag* species (red) gets extinguished. Moreover, there is an increment in the *go to tag* species seeking accuracy (blue), but it is useless since there are no available tags any more. A few iterations before 10000 only *explorers* survived trying to use the little remaining accuracy to explore an environment which is becoming more and more uncovered. It takes the algorithm around 2000 more time steps of low quality individuals (all of them are condensed in a black line shown in the bottom part of the bottom plot of figure 4) to finally produce the two required species to successfully explore the environment and to rise the global fitness again to a successful level (become *mobile tag* and *exploring* the surrounding area).

Figure 5 contains a representation of the genes of the population along this evolution. To do so, each genotype is projected into two independent dimensions. Those individuals that live less than 100 time steps are depicted in black meaning that there was no time to consistently assign a behavior to them. As it can be observed in figure 5, during the instability gap (iterations 8000 to 12000), the genomes of the population expand along the genetic state space, producing a higher number of ephemeral species that, just before the recovery (iteration 12000), decrease quickly to converge again to a configuration of two main species.

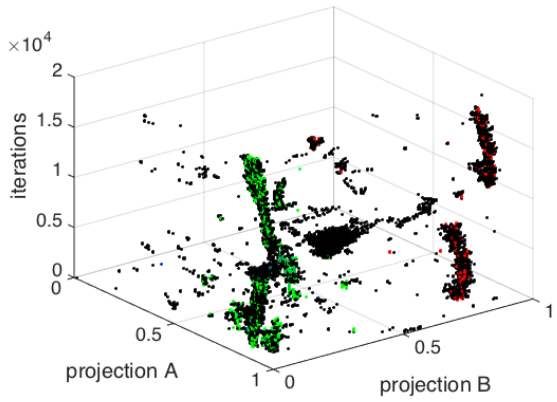

Fig. 5. Representation of the genes of the population along the evolution where each genotype has been projected into two independent dimensions.

Figure 6 displays 6 plots where the correlation between performance and number of species has been studied for a set of configurations that vary the environment and individual complexities. The size of the neural network was set to 3x1x1 (4 weights), 3x2x1 (8 weights) and 3x10x1 (40 weights) and it is displayed in the top, middle and bottom rows in figure 6. Moreover, the environment was set with and without permanent tags (left and right columns in figure 6), and with

and without obstacle penalties (red and blue lines in all the plots). The data displayed in this figure were obtained after 5 runs of 20000 iterations each.

The comparison shows several aspects of this interaction between complexity and the outcome of the co-evolutionary process. In the case of a scenario without permanent tags and without penalties for collisions (left column, blue lines), as expected, the simplest controller (4 weights) tends to produce two or three species to optimize its behavior (peak value of the blue curve shown in top-left plot). If the number of weights of the ANN is doubled (middle-left plot, blue line) then we can see that efficient solutions can be found with several configurations of species, with a slight tendency towards one species configurations, which indicates a greater robustness and versatility. However, if we set an ANN with up to 40 weights (bottom-left top, blue line), we observe a decrease in the overall performance and now the system tends to avoid only one species configurations. After studying different runs with this configuration (40 weights), it can be seen that it is harder for the algorithm to find a solution, and also to converge to only one species in such a high dimensional search space, and that some of the runs provide poor performance.

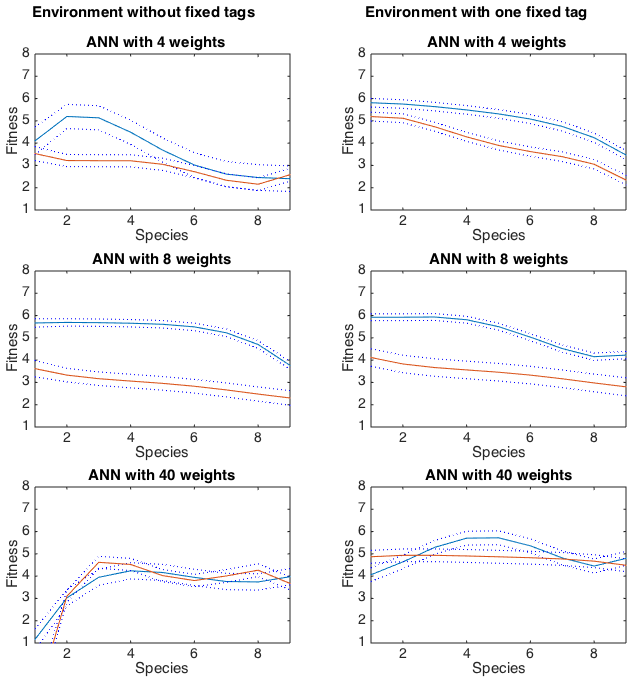

Fig. 6. Correlation between performance and number of species for a set of configurations that vary the environment and individual complexities. Top plots correspond to 3x1x1 (4 weights) ANN, middle ones to 3x2x1 (8 weights) and bottom ones to 3x10x1 (40 weights). Left plots do not have permanent tags while right ones do. Finally, the red lines correspond to executions with obstacle penalties while the blue lines correspond to executions without them.

When we activate the penalties for collisions (left plots, red lines), there is a higher complexity for navigation, and consequently the performance is much lower for 4 and 8 weights (top and middle plots). However, in the case of 40 weights it does not seem to affect greatly, which indicates that, although this configuration is harder to adjust, it is less affected by the complexity of the environment. On the other hand, when the environment becomes simpler because the permanent tags are included (right column plots), the evolution, in almost all the configurations, tends to create only one species which is able to perform successfully. Again, if the penalties are included (red lines), the performance decreases but the tendency regarding the species remains the same. Finally, the controller with 40 weights becomes, again, much less affected by the penalties and exhibits also again a tendency to avoid one species configurations.

Summarizing, for this experimental setup an intermediate complexity of the ANN (8 weights) has shown to be beneficial both in terms of performance and in stability regarding number of possible species. Using complex individuals or very simple ones led to a more unstable response of the algorithm. In the case of the more complex controller it is more complicated to evolve, but it is less affected by the complexity of the scenario. Finally, simpler scenarios tend to create one species configurations for most of the controllers.

In terms of the response of the algorithm, its capability to adapt the behavior and structure of the population to different individual and scenario configurations regarding their level of complexity has been shown. The algorithm is able to take advantage from the heterogeneity of the population when it is required to achieve a good global performance by making use of the specialization, as it has been constantly observed in natural evolution in real ecosystems (colonies of ants, termites or bees). It also tends to simplify the structure of the population when the scenario is simple in relation to the complexity of the individuals due to the evolutionary cost of adjusting a large genome. Interestingly, if the controller shows a high complexity the algorithm fails to produce an efficient population in some runs but when it doesn't it is more robust to changes in the complexity of the task.

Conclusions

This paper has studied the impact of complexity in the emergence of collective solutions to distributed problems by means of an implementation of Embodied Evolution for a collective surveillance task with simulated Micro Aerial Vehicles. The algorithm used, canonical Embodied Evolution, has already been tested for several applications and has shown its capability to decompose a task into several subtasks and to improve performance by generating different species among the individuals of the population. This work has presented a first attempt to deepen in the mechanisms that affect the creation of those species when the whole population pursues a common goal. The study has shown a strong impact on the outcome of the algorithm of variations in the complexity of the definition of the problem at different levels, namely, the complexity of the scenario and the complexity of the individuals. In terms of the response of the algorithm, the algorithm has found efficient groups of controllers for different individual and scenario setups regarding their level of complexity. As in the case of natural evolution, which

embodied evolution approaches mimic, the algorithm works with, and is able to take advantage of, heterogeneous populations when they are required to achieve a good global performance by making use of specialization. Our ongoing work, based on the results presented in this paper, is focused on including the number of parameters of the controllers as an individual evolvable parameter. The main goal will not be to obtain an optimal value for the size of the neural network of the whole population, since this can already be analyzed from the current results, but to study the dynamics produced when populations can be heterogeneous in both individual behavior and complexity.

References

Anderson C., McShea D.W. (2001), Individual versus social complexity, with particular reference to ant colonies, *Biological reviews of the Cambridge Philosophical Society. 76(2),* pages 211-37.

Bonabeau, E., Theraulaz, G., Denebourg, J.L., Aron, S. Camazine, S. (1997) Self-organization in social insects *Trends in Ecology and Evolution*, Vol. 12, No. 5. pages 188-193

Bredeche, N., Montanier, J.M., Liu, W., Winfield, A. (2012) Environment-driven Distributed Evolutionary Adaptation in a Population of Autonomous Robotic Agents, *Mathematical and Computational Modelling of Dynamical Systems 18, 1*, pages 101-129

Burbeck, S. (2007), Complexity and the Evolution of Computing: Biological Principles for Managing Evolving Systems**,** *Technical Report* (http://www.evolutionofcomputing.org)

Corchado, J., Bajo, J., Kozlak, J., Pawlewski, P., Molina, J., Gaudou, B., Julian, V., Unland, R., Lopes, F., Hallenborg, K., Garcia, P. (2014) Highlights of Practical Applications of Heterogeneous Multi-Agent Systems, *the PAAMS Collection: PAAMS 2014 International Workshops,* Springer Publishing

Davies, D., Bouldin, D. (1979) A Cluster Separation Measure, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. PAMI-1 (2), pages 224–227.

Detrain, C., Deneubourg, J.L. (2002), Complexity of environment and parsimony of decision rules in insect societies, *Biological Bulletin 202(3),* pages 268-74.

Duro, R.J., Bellas, F., Prieto, A., Paz-López, A. (2011) Social Learning for Collaboration through ASiCo based Neuroevolution, *Journal of Intelligent and Fuzzy Systems 22*, pages 125-139.

Elfwing, S., Uchibe, E., Doya, K., Christensen, H. (2011) Darwinian embodied evolution of the learning ability for survival, *Adaptive Behavior 19, 2*, pages 101-120

Epstein, J., Axtell, R. (1996) Growing Artificial Societies, *The MIT Press*

Eiben, A.E., Haasdijk, E., Bredeche, N. (2010) Embodied, On-line, On-board Evolution for Autonomous Robotics, *Symbiotic Multi-Robot Organisms: Reliability, Adaptability, Evolution*, pages 361-382 Springer

Ficici, S., Watson, R, Pollack, J. (1999) Embodied Evolution: A Response to Challenges in Evolutionary Robotics. *Eighth European Workshop on Learning Robots*, pages 14-22

Gomes, J., Mariano, P., Christensen, A.L. (2015) Cooperative Coevolution of Partially Heterogeneous Multiagent Systems. *In Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems (AAMAS '15)*, pages 297-305.

Gomez, F. and Miikkulainen, R. (1997) Incremental evolution of complex general behavior, *Adaptive Behavior, 5*, pages 317–342.

Hanna, L., and Cagan, J. (2009) Evolutionary Multi-Agent Systems: An Adaptive Approach To Optimization In Dynamic Environments, *ASME Journal of Mechanical Design*, Vol. 131, No. 1, pages 1-11

Nitschke, G.S., Schut, M.C., and Eiben, A.E. (2008) Emergent Specialization in Biologically Inspired Collective Behavior Systems. *Chapter in A. Yang and Y. Shan (eds.), Intelligent Complex Adaptive Systems,* pages 215-253, IGI publishing.

Nitschke, G.S., Schut, M.C., and Eiben, A.E. (2012) Evolving behavioral specialization in robot teams to solve a collective construction task, *Swarm and Evolutionary Computation, Volume 2*, pages 25-38

Mitchell, M. (2011) Complex Systems: a guided tour, *Oxford University Press*

Olson, E. (2011) AprilTag: A robust and flexible visual fiducial system. *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA),* pages. 3400-3407

Panait, L. (2010) Theoretical convergence guarantees for cooperative coevolutionary algorithms. *Evolutionary Computation, 18(4)*, pages 581–615, MIT Press

Prieto, A., Becerra, J.A., Bellas, F., Duro, R.J. (2010) Open-ended Evolution as a means to Self-Organize Heterogeneous Multi-Robot Systems in Real Time, *Robotics and Autonomous Systems, vol. 58*, pages 1282-1291

Prieto, A., Trueba, P., Bellas, F., Duro, R.J. (2015) Towards the standardization of distributed Embodied Evolution, *Information Sciences, vol 312*, pages 55-77

Rinde R.S. van Lon, Tom Holvoet, Greet Vanden Berghe, Tom Wenseleers, and Juergen Branke. (2012) Evolutionary synthesis of multi-agent systems for dynamic dial-a-ride problems. In *Proceedings of the 14th annual conference companion on Genetic and evolutionary computation* (GECCO '12), pages 331-336, Terence Soule (Ed.)

Schut, M.C, Haasdijk, E., Prieto, A. (2009) Is situated evolution an alternative for classical evolution?, *Proceedings of the Eleventh conference on Congress on Evolutionary Computation*, pages 2971 – 2976

Trueba, P., Prieto, A., Bellas, F., Caamaño, P., Duro, R.J. (2013) Specialization analysis of embodied evolution for robotic collective tasks, *Robotics and Autonomous Systems, Volume 61, Issue 7*, pages 682-693

Wiegand, R. P. (2003) An Analysis of Cooperative Coevolutionary Algorithms. PhD thesis, George Mason University

Yong, C. H. and Miikkulainen, R. (2009) Coevolution of role-based cooperation in multiagent systems. *IEEE Transactions on Autonomous Mental Development, 1(3)*, pages 170–186