# Emerging Cooperation With Minimal Effort: Rewarding Over Mimicking

Georgios N. Yannakakis, *Member, IEEE*, John Levine, *Member, IEEE*, and John Hallam

***Abstract*—This paper compares supervised and unsupervised learning mechanisms for the emergence of cooperative multiagent spatial coordination using a top-down approach. By observing the global performance of a group of homogeneous agents—supported by a nonglobal knowledge of their environment—we attempt to extract information about the minimum size of the agent neurocontroller and the type of learning mechanism that collectively generate high-performing and robust behaviors with minimal computational effort. Consequently, a methodology for obtaining controllers of minimal size is introduced and a comparative study between supervised and unsupervised learning mechanisms for the generation of successful collective behaviors is presented.**

**We have developed a prototype simulated world for our studies. This case study is primarily a computer games inspired world but its main features are also biologically plausible. The two specific tasks that the agents are tested in are the competing strategies of obstacle-avoidance and target-achievement. We demonstrate that cooperative behavior among agents, which is supported only by limited communication, appears to be necessary for the problem's efficient solution and that learning by rewarding the behavior of agent groups constitutes a more efficient and computationally preferred generic approach than supervised learning approaches in such complex multiagent worlds.**

***Index Terms*—Artificial world, genetic algorithms (GAs), machine learning, multiagent, spatial coordination.**

## I. Introduction

COOPERATIVE behavior within artificial simulated worlds of multiple agents, such as artificial life environments and computer game virtual worlds, is a prominent area of research. Designing agents for such systems can be a repetitive and tedious procedure. This task becomes even more difficult when the investigated multiagent environment is highly dynamic, nondeterministic, and the agent's motion is continuous. Additional complication is present when agents' sensory information is limited. Thus, when designing controllers for autonomous simulated agents for such environments, there is little guidance on how complex the controller must be for the agents to achieve good performance in particular tasks. Furthermore, when high performances have to emerge via a learning mechanism, there is little knowledge about the mechanism's design, complexity, and computational effort.

Manuscript received June 29, 2005; revised April 26, 2006.

G. N. Yannakakis and J. Hallam are with the Mærsk Mc-Kinney Møller Institute for Production Technology, University of Southern Denmark, DK-5230 Odense, Denmark (e-mail: georgios@mip.sdu.dk; john@mip.sdu.dk).

J. Levine is with the Department of Computer and Information Sciences, University of Strathclyde, Glasgow G1 1XH, U.K. (e-mail: john.levine@cis.strath.ac.uk).

Digital Object Identifier 10.1109/TEVC.2006.882429

Motivated by the aforementioned challenges in such dynamic multiagent worlds, we automatically built successful artificial neural network (ANN) controllers that yield cooperative spatial coordination, using minimal effort to construct them. We follow a top-down approach by observing the global performance of agents, which own individual tasks, and then investigating the local agent interactions that lead to cooperative behaviors. Machine learning is used to serve this purpose. Hence, we introduce an efficient methodology for designing a group of high-performing homogeneous agents and compare the performance, robustness and required computational effort between supervised (gradient-search) learning mechanisms—"learning by samples" of good behavior—and evolutionary learning mechanisms that produce a strategy by reinforcing rewards and penalties dependent on the solution's behavior (i.e., "learning by rewards").

The coordinated motion task investigated is based on limited communication. Communication is limited in that it is:

- implicit—agents do not explicitly communicate, rather they sense each other's positions;
- partial—agents can only sense local neighborhoods, not the whole environment;
- passive—communication among agents does not alter the environment (unlike stigmergy, for example).

We make these choices by following principles of the *animat* approach [1] which provides the ground for more robust agent behaviors and generality over the complexity of the environment. Furthermore, we aim for a minimal agent motion controller because of the several important advantages that it offers. The smaller the controller, the better (and easier) the understanding of its functioning by direct inspection. Additionally, the controller becomes computationally more efficient and less expensive. Finally, the size and the structure of the minimized controller may provide an estimate of the task's complexity [2].

The "*Flatland*" [3], [4] case study is used to assess and compare the performance, robustness and effort cost of the applied machine learning mechanisms. *Flatland* is a prototype two-dimensional (2-D) multiagent world inspired primarily by the virtual worlds met in computer games and by biological aspects like animal foraging and mate-seeking. The two tasks that the agents are tested in are the antagonistic strategies of obstacle-avoidance and target-achievement (individual task). *Flatland* is a complex environment in that its properties depend on the *interactions* between agents—moreover, the dynamics of agent behavior depend on discontinuously changing relationships between the agents as they prove their double task. In that sense, complexity in *Flatland* is a feature determined by the number of agents present.

Overall, the results in this paper show that cooperative behavior among agents emerges—built on implicit, partial, and passive communication—and that unsupervised learning mechanisms are able to build more robust controllers than supervised mechanisms, and with less computational effort.

This paper is organized as follows. A literature review on related work is presented is Section II. In Section III, we present a detailed description of *Flatland*, and the agents' controllers. In Section IV, we discuss the difficulties and points of importance of this simulated world. The machine learning mechanisms used are analytically described in Section V. Section VI presents the learning procedure used for the minimization of the agent controller. Results obtained from varying agent population sizes in the *Flatland* world are presented and analyzed in Section VII. This section concludes with the presentation of an effective methodology for estimating the generalization error produced by supervised learning mechanisms, which provides evidence against their appropriateness in *Flatland*. Finally, the most important conclusions of this paper are summarized in Section VIII.

## II. Related Work

Areas related to our general approach mainly include work on emerging or learning cooperative spatial coordination in simulated environments of multiple agents. Work related to the *Flatland* world is discussed in Section III-A.

### A. Emerging Cooperation

Considerable research has been conducted towards the emergence of global cooperative spatial coordination from local communication in multiagent simulated environments. Artificial life approaches include Reynolds' work on *Boids* [5] based on the use of local partial communication towards the generation of global successful flocking strategies. Spatial coordination (flocking) in artificial multiagent worlds grounded in behavior-based [6] and artificial potential field [7] approaches are reported in [8] and [9], respectively. On that basis, *The Game* [10], designed by Bonabeau, is a simple agent-based simulation of collective behaviors which emerge from local interactions. Agents in this game are assigned an aggressor and a defender agent and they move to keep their defender between them and their aggressor. In [11], the proper levels of balance between global and local knowledge of a multiagent simulated world are investigated. Cooperative flocking ("keep formation") behaviors emerge through local rules.

Instead of designing the local agent interactions and then observing the emergent global behavior (as in all aforementioned studies on emergent cooperation—see [5] and [10]), we study the inverse procedure; by focusing on and investigating the global behavior (performance), we attempt to derive the features that generate cooperative behaviors through local interactions. This is achieved using machine learning procedures that either reward or mimic efficient global cooperative behaviors. This paper compares the two strategies of rewarding and mimicking, assessing their performance and the computational effort each requires.

### B. Automatic Design of Motion Controllers

Generating cooperative motion controllers through learning procedures in artificial worlds is a closely related field of research. This field includes primarily artificial life, multiagent systems, and artificial intelligence in computer games research. A popular example of such work is the neuroevolution procedure used in Werner's *BioLand* [12] to generate both herding and prey behaviors. Likewise, Miller's and Cliff's work [13] in coevolution techniques of pursuit-evasion tactics, Luke and Spector's [14] work on evolving hunting behaviors in the *Serengeti* world and Koza's [15] genetic programming applications in a wide array of simulated pursuit-evasion scenarios also constitute characteristic pieces of work in the field.

Other examples of building neural controllers for cooperative spatial coordination include the work of Togelius and Lucas [16] on the emergence of foraging-related behaviors in the simulated world (computer game) named "Cellz" and the work of Salustowicz *et al.* on learning evaluation functions for a simulated soccer game [17]. Moreover, Baldassarre's work on multiple neural-controlled agents that learn to gather good food and avoid bad food shares principles with the collective behaviors studied here [18]. However, Baldassarre investigates the impact of social transmission of behaviors and skills on the global performance.

Reynolds [19] used genetic programming techniques for the emergence of herding behaviors of *Critters* against predators. He evolved a motion controller gathering information about its neighbors and predators in a homogeneous environment. On the other hand, there are tasks, such as schooling behaviors in simulated fish, for which solutions could not be found this way [20].

Evolving neural controlled agents for computer games with multiple characters is a closely adjacent research area. Yannakakis *et al.* [21]–[23] present a robust adaptive offline evolutionary learning mechanism for shaping predator motion controllers in predator/prey games. Likewise, Stanley *et al.* [24] are applying neuroevolution techniques for the emergence of adaptive behaviors (e.g., capture-the-flag and wall-avoidance) in a training game. Furthermore, there have been numerous studies on shaping neurocontrollers for complex collective behavior in the swarm robotics literature ([25] and [26] among others); these constitute yet more examples of the rewarding (unsupervised learning) strategy presented in this paper.

In addition to unsupervised learning techniques, used in most of the work mentioned here, this paper presents a handcrafted preprogrammed near-optimal solution which is used as a good example for agents to mimic. Therefore, instead of just designing a unique learning approach, we examine the impact of the two machine learning types (by mimicking, by rewards) on the agents' performance and introduce a methodology for obtaining cooperative behaviors with minimal effort. Furthermore, additional focus is given to the minimization of the motion controller's size and the factors (elements of the learning mechanisms) that facilitate the emergence of cooperative behaviors.

## III. The *Flatland* Simulated World

The name "*Flatland*" is inspired by the title of E. Abbott's book [27] and its fundamental concept is based on previous research by Yannakakis [3], [4], [28]. The main purpose of this simulated world is to be used as a testbed environment for investigating evolutionary and gradient-based (to a lesser degree)

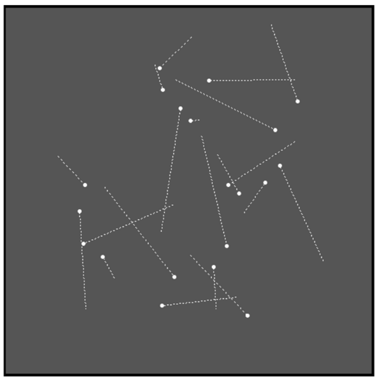


Fig. 1. *Flatland* world interface (the plane's dimensions are 80 cm × 80 cm in the experiments presented here).

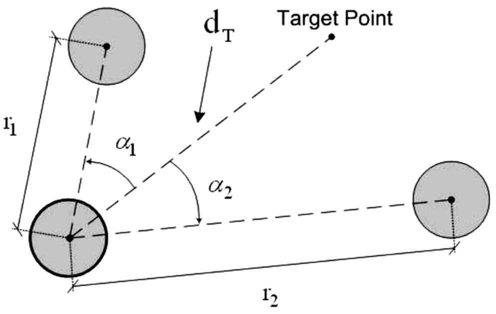


Fig. 2. Human's input data ($z = 2$).

learning techniques and furthermore, their ability to generate cooperative obstacle-avoidance and target-achievement behaviors.

*Flatland* is primarily a computer game inspired testbed and the design of the simulated agents used in this environment is deliberately kept abstract. Computer games provide an ideal environment for research on emerging cooperation because they are based on simulation of highly complex and dynamic multiagent worlds [29]. In contrast with the real world (e.g., realistic robotic environments), experiments can be easily observed and access to the world state is fully controllable.

*Flatland* is a square (80 cm × 80 cm), 2-D, multiagent environment consisting of a number of 2-D white circular (radius of 5 mm) agents (as illustrated in Fig. 1). Agent motion is not affected when they collide as they pass through each other. However, "collisions" are penalized when assessing fitness. Each agent is assigned a target point on the environment's surface—dashed straight lines connecting the agent's current position to its target point on the surface are illustrated in Fig. 1. As soon as an agent achieves its current target (i.e., manages to reach a circle of 5 mm around the target point), then a new target point is selected. The new target point is picked from a uniform random distribution at a specified distance of 30 cm from the agent's center.

## A. Related Problems to the Flatland Paradigm

The *Flatland* agents' tasks share features with the self-reconfiguration task approached via cellular automata and gradient information algorithms [30]. Moreover, our agent tasks (target-achievement and obstacle avoidance) are analogous to the tasks of the SimWorld simulation environment, which is used for investigating potential evolutionary trajectories between preprogrammed agents that gather resources and avoid dynamic obstacles [31].

The basic features of the *Flatland* paradigm are also biologically plausible. Foraging and clustering are two closely related tasks to the investigated world where target points represent food. The AntFarm [32] world used for investigation of cooperative ANN controlled artificial ants through evolution and Arkin's [33] work on foraging behaviors based on stigmergy techniques constitute representative examples in the field. Such approaches, however, imply communication by means of modifying the environment (i.e., stigmergic communication). Collision-avoidance behavior is either handcrafted or ignored in the aforementioned literature review; *Flatland* agents attempt to learn it. In addition to foraging, mate attraction may also be considered as a related biologically-inspired task, where each agent (e.g., frog—see [34]) is attracted by its own mate (target point).

Knowledge gained from *Flatland* has been successfully applied in the multiagent predator/prey computer games domain. Examples of these applications include the well known Pac-Man game [21], [23], as well as a predator/prey related game called "Dead End" [22]. In both games the impact of cooperative online (during play) learning opponent behaviors on the player's satisfaction (interest) is studied.

## B. Humans

ANNs are suitable means for embedding emergent adaptive behaviors in complex multiagent environments [35], [36], [37]. A feedforward neural controller is thus employed to manage the agents' motion. This "species" of agents is called "Humans."

*1) Input:* Using its sensors, each Human inspects the environment from its own point of view and decides its next action. Sensors implemented are omnidirectional with infinite range. Each Human receives information expressed in the ANN input array of dimension $2z + 1$ consisting of: a) polar coordinates $(\alpha_i, r_i, i = 1, \ldots, z)$—based on the axis determined by the current position of the Human and its target point (see Fig. 2)—of the $z$ closest Humans and b) the distance between the Human's current position and its target point $(d_T)$ (see 2).

All input values are linearly normalized into [0, 1] before they enter the neural controller. The input format in polar coordinates is based on Reynolds' work on artificial critters [38]. For the experiments presented in this paper $z = 2$, which was found to be the minimal amount of information for a Human to successfully achieve the desired behavior (for $z = 1$ neural controllers are not able to generate satisfactory obstacle-avoidance strategies).

*2) Architecture:* As previously noted, one of the challenges we encounter in our approach is to design the simplest motion controller capable of generating the desired behavior. Since it is a challenging task to define and quantify simplicity of an ANN, we aim for the minimization of successful fully connected architectures (i.e., number of neurons and hidden layers). To this end, moderate size (i.e., fewer than two hidden layers and fewer than

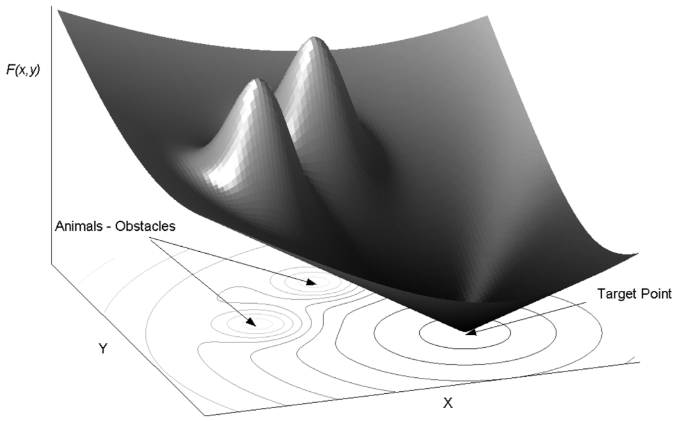


Fig. 3. APF—Situation of two obstacles—closest Animals ($z = 2$).

30 hidden neurons in each layer) multilayered fully connected feedforward ANNs have been used for the experiments presented here. The sigmoid function is employed at each neuron. In the attempt to minimize the controller's size, it is found (see Section VII-B1) that single hidden-layered ANN architectures, containing fewer than 13 hidden neurons, are capable of generating efficient and robust solutions.

The connection weights take values from $-5$ to 5, while the ANN output is a 2-D vector $[o_1, o_2]$ with each value in [0, 1]. This vector represents the Human's simulation step motion and is converted into polar coordinates $r_{\mathrm{NN}} = o_1 M$ and $\alpha_{\mathrm{NN}} = (2o_2 - 1)\pi$ where $M$ is agents' maximum speed; $r_{\mathrm{NN}}$ is the Human's step motion (in cm/simulation step) and $\alpha_{\mathrm{NN}}$ is the Human's turn angle (in degrees) with respect to its current axis. In experiments presented here, $M = 1$ cm/(simulation step). The agent's rotation and step motion take place simultaneously at each simulation step.

## C. Animals

Using the same environment, we explored an additional "species" of agents. These agents are called "Animals" and differ from Humans only in the control of their locomotion. Instead of an ANN, an artificial potential field (APF), specially designed for this environment, controls the Animals' motion. The essence of the APF is that points along the Animal's path to its target point are considered to be attractive, while obstacles (other Animals) in the environment are repulsive [7] (see also [33] for an application of APF in multiple robots). The overall APF causes a net force to act on the Animal, which guides it along a collision-free, target-achievement path. For illustration, consider the Animal as a small sphere (of radius $R = 5$ mm) that slides down the surface plotted in Fig. 3. This surface is plotted by each Animal at every simulation step and represents the potential

$$F(x,y) = \frac{M}{2}\sqrt{(x-x_T)^2 + (y-y_T)^2} \quad \text{(1a)}$$

$$+ \delta \sum_{i=1}^{z} e^{-[(\frac{x-x_i}{4R})^2 + (\frac{y-y_i}{4R})^2]} \quad \text{(1b)}$$

where $[x_T, y_T]$ are the coordinates of Animal's target point; $[x_i, y_i]$ are the coordinates of the center of the closest obstacles (other Animals); $\delta$ is a parameter that controls obstacle repulsive forces via (1b).

The surface plotted by each Animal changes at every simulation step as a result of *Flatland*'s dynamics (moving obstacles—other Animals—and changing neighbors). Hence, the Animals' motion consists of a fixed (nonevolving) nonlinear strategy and is determined by the 2-D discontinuously time-varying potential field represented by (1). While, in theory, the APF solution may be prone to getting stuck in local minima, in practice, in the dynamic *Flatland* world, the probability $p_C$ for such instances to occur is statistically insignificant ($p_C \ll 0.1$) and, therefore, can be ignored.

Any motion strategy that guides an agent to quickly achieve its target, avoiding any possible collision and keeping the straightest and fastest possible trajectory is definitely a "good" strategy in terms of *Flatland* world. Hence, Animals present a "good" (near-optimum) behavior in our simulated world, and thus are a reference case to compare with any Humans' behavior. Moreover, data from the Animals' motion strategy can be used to train the Humans' ANN controller (see Section V-B).

## D. Target Achievers (TAs)

TAs are agents that ignore collisions and move directly towards their target points with constant speed; $\alpha_{\mathrm{NN}} = 0^o$, $r_{\mathrm{NN}} = 0.5$ cm/simulation step.

## IV. Challenges

In this section, we provide evidence of the problem's complexity and learning difficulty, as well as its importance in the multiagent systems research area. In fact, *Flatland* is a hard environment for an agent to learn to perform in because of its distinct features. These are as follows.

- **Fully dynamical multiagent**. Agents move continuously. Each agent faces a number of moving obstacles (i.e., potentially 19 other agents) in a specific squared environment and it has no *a priori* knowledge about their motion.
- **Partial information**. Each agent is able to capture the position of only $z$—in all experiments presented here $z = 2$—other agents.
- **Implicit information**. Agents communicate just by "seeing" each other (see Fig. 2). This kind of communication regarding the specific tasks (i.e., obstacle-avoidance and target-achievement) is very common in the animal world (e.g., predator/prey behaviors), as well as in humans (e.g., crowded streets).
- **Discontinuous time-varying information**. The agent's input information suffers from discontinuity because of frequent alterations of the $z$ closest neighbors that it takes into account via its sensors. Hence, the values of the polar coordinates $\alpha_i, r_i$ $(i = 1, \ldots, z)$ alter in a discontinuous fashion.
- **Supervised training**. If we attempt to train Humans under the near-optimal APF (Animals) strategy, we face problems of missing information for many instances. These are situations that Animals would never get into (e.g., the instance of $r_i \leq 2R$) but trained Humans do. Overall, the task of choosing the most appropriate training set is something of an art in itself that requires a lot of trial and error

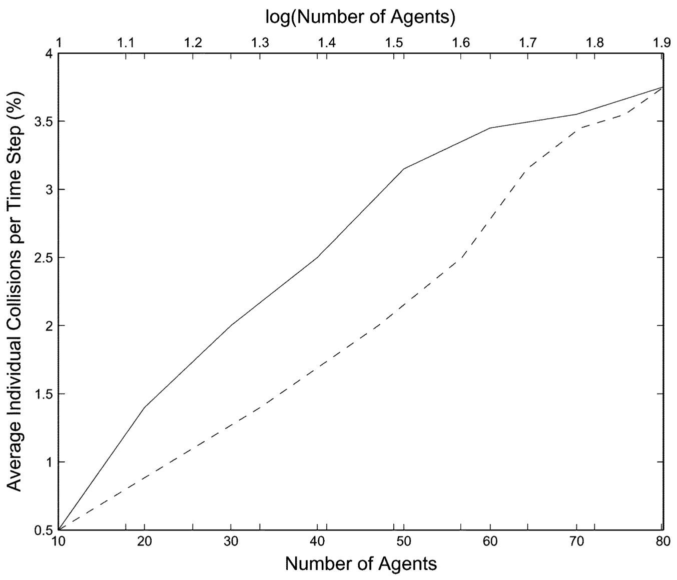


Fig. 4. Worst collision-avoidance behaviors: average individual collisions per simulation step over the number of simulated agents (solid line) and the logarithm of the number of simulated agents (dashed line).

experiments. Unfortunately, the exact features for an efficient set of data cannot be explicitly defined. Such features include the tradeoff between size and computational effort, the right proportion of antagonistic behavior examples (i.e., target-achieving contra collision-avoidance examples) and the specific required examples for each trained behavior.

- **Unsupervised learning**. One of the challenges of unsupervised learning in *Flatland* is the small number of collisions per simulation step in relation to the environments' complexity. In the worst obstacle-avoidance behaviors we experienced (we consider TAs as agents that generate such a behavior in this paper), in the most complex environment of 80 agents, each agent collides 375 times in $10^4$ simulation steps on average (i.e., 3.75% of its lifetime on average). For the simplest environment used (i.e., ten-agent) this percentage is approximately 0.5%. Therefore, it is both hard and computationally expensive for an obstacle-avoidance strategy to emerge from rewarding good examples of this strategy. Furthermore, when increasing the population of simulated agents, the average individual collisions per simulation step appear to increase logarithmically (see Fig. 4).

*Flatland*'s basic concept and features make the proposed testbed interesting for the multiagent artificial life research area. The generality of this world extends into the area of computer games as successful applications have already shown [21], [22].

- **Emerging cooperation**. *Flatland* is a simulated world in which we expect cooperative behaviors to emerge without any information exchange apart from spatial coordination (see above). Hence, emergent cooperation derives from: 1) the way Humans move and 2) the way they interact with their environment (see Section VII).
- **Strong agent-environment interaction**. There is a strong interaction and relation between the simulated agents and their environment. Agents in *Flatland* are part of their own environment. Furthermore, *Flatland*'s main feature, as an environment, is its own agents.

Both aforementioned features of *Flatland* define important points in the research of 2-D multiagent dynamic simulated worlds. Computer games and artificial life offer a great arena of such worlds and a plethora of applications such as *The Game* simulated world [10], predator/prey computer games like *Pac-Man* [21], [23] and *Dead End* [22], and the training *NERO* game [24].

## V. Learning Mechanisms

In this section, we present the three learning mechanisms used in the *Flatland* case study. The main approach (presented in Section V-A) is evaluating Humans via their own actions in the *Flatland* environment, whereas the alternative approaches presented in Section V-B consist of supervised learning techniques attempting to train Humans on Animals' behavior. To summarize, we discuss the mechanisms' key elements that determine the basic research axes that this paper covers (see Section V-C).

### A. Learning by Rewards

A generational genetic algorithm (GGA) [39] is implemented, which uses an "endogenous" evaluation function that derives from the Humans' actions in the environment and promotes good collision-avoidance and target-achievement behaviors. The ANNs that determine the behavior of the Humans are themselves evolved. In the algorithm presented here, the evolving process is limited to the connection weights of the ANN. Evolving both connection weights and architectures simultaneously is the algorithm used for minimizing the controller's size (see Section VI).

The evolutionary procedure used can be described as follows. Each agent has a genome that encodes the connection weights of its ANN. A population of $N_p$ ANNs is initialized randomly. Initial real values that lie within $[-5, 5]$ for their connection weights are picked randomly from a uniform distribution. Then, at each generation:

Step 1) Every Human in the population is cloned $N$ times ($N$ being the number of agents in *Flatland*). These $N$ clones are placed in the *Flatland* environment and tested for an evaluation period $e_p$ (e.g., 300 simulation steps). The outcome of this test is to ascertain the total number of collisions $C$ and target achievements $T$.

Step 2) Each Human is evaluated via the following function:

$$f_i = \frac{\max\left\{1 - \frac{C_i}{C_u}, 0\right\} + \min\left\{\frac{T_i}{T_u}, 1\right\}}{2} \quad (2)$$

where $f_i$ is the evaluation function of Human $i$; $C_i$ is the total number of collisions of Human $i$'s $N$ clones; $C_u$ is the total number of collisions' upper bound which is determined by the total number of collisions of $N$ TAs in $e_p$ simulation steps (for $N = 20$ and $e_p = 300, C_u = 60$—see also Table V); $T_i$ is the total number of target achievements of the Human $i$'s N clones; $T_u$ is the total number of target achievements' upper bound which is determined by the total number of target achievements of N Animals in $e_p$ simulation steps (for $N = 20$ and $e_p = 300$, $T_u = 96$—see also Table V).

Step 3) A pure elitism selection method is used where only a small percentage $N_s$ of the fittest solutions is able to breed and, therefore, determine the members of the intermediate population.

Step 4) Each of the parents clones an equal number of offspring so that the total population reaches $N_p$ members. Alternatively, uniform [40] and Montana and Davis [41] crossover operators have been used at this step but proved unsuccessful. The explanation is the disruptive feature of crossover operators when dealing with distributed knowledge representation (i.e., ANN). That is, crossover among parts of different successful ANNs is very likely to lead into unsuccessful offspring [42]. Results obtained from experiments with crossover operators are not presented in this paper.

Step 5) Mutation occurs in each gene (connection weight) of each offspring's genome with a small probability $p_m$. A uniform random distribution is used again to define the mutated value of the connection weight.

The algorithm is terminated when either a best fit Human (i.e., $f_i = 1.0$) is found or a large number of generations $\mathbf{T}$ is completed.

The use of (2) rewards Humans (their $N$ clones) that do not crash and achieve a determined number of targets ($T_u$) during an evaluation period. By this evaluation, we mainly promote clones capable of cooperating in order to successfully achieve the aforementioned desired behavior. Due to this, very interesting cooperative behaviors emerge within a homogeneous environment (see Section VII).

We mainly use small simulation periods $e_p$ to evaluate Humans via (2) to limit the computational effort. Thus, this evaluation function constitutes an approximation of the overall performance of the examined Humans in large simulation periods (see Section VII-A). Keeping an appropriate balance between computational effort and performance approximation is one of the key features of this unsupervised learning approach. In the experiments presented here, we use minimal evaluation periods capable of producing good estimation of the Humans' overall performance.

## *B. Learning by Samples of Good Behavior*

In this subsection, we present two alternative supervised learning mechanisms. Both attempt to generate desired behaviors by mimicking the near-optimal Animal strategy.

*1) Back-Propagation (BP):* The use of this supervised learning approach is based on an evaluation which promotes any behavior that mimics the Animals' strategy. The data set used for the supervised BP training of the neural controllers consists of inputs (perceptions) and actions of an Animal. This data collectively draws a near-optimal Animal path which is picked from a simulation of 80 Animals in the *Flatland* environment. That is because the more crowded the *Flatland* world, the more obstacle-avoidance examples to learn from. Furthermore, since obstacle-avoidance is harder than target-achievement behavior to learn, such data sets define good samples for training. Note that for the 20 and the 80-Animal environment, collision-avoidance samples cover approximately the 66% and the 95% of the Animal's simulation time, respectively.

Many different Animal paths have been used for determining a data set that produces the smallest generalization error and furthermore, the best agent performance. In this paper, results only from this data set are presented. We believe that a learning mechanism's efficiency and reliability are based on the overall effort made for achieving desired solutions. As far as the "learning by mimicking" mechanisms are concerned, this effort includes the experimental selection of the most appropriate data set.

The data set's inputs and targets are uniformly normalized into [0, 1], whereas its size $S_t$ is 666 samples for the experiments presented here, this size being 2/3 of a data set that is originally partitioned into training and testing portions for estimation of the generalization mean square error (mse) (also, see Section VII-D for an alternative method on estimating the generalization error). Early stopping methodology is used for avoiding overfitting and the Levenberg–Marquardt algorithm [43] is used to train the ANN. This algorithm appears to be the fastest method for training moderate-sized feedforward ANNs and has given the highest performance training results among many other training algorithms employed.

The algorithm is terminated either when it converges to a good training mse value (e.g., this mse value depends on the topology of the network) or when the mse on the test data set increases (i.e., early stopping) or once a predefined large number of epochs (e.g., 2000 epochs) is completed.

For validation purposes, the BP approach has been tested on the well-known two-spiral classification benchmark [44]. A network with a total of 141 connection weights (two hidden layers with ten neurons each) successfully classifies all 194 points of this problem in 9950 epochs.

*2) Teacher-Based GA (TGA):* The TGA is a supervised learning evolutionary algorithm which endeavors to generate good collision-avoidance and target-achievement behaviors by rewarding Humans that follow the Animal's (i.e., the Teacher's) good paradigm of behavior. Hence, every Human that is placed into *Flatland* is rewarded or penalized for its actions by its own Animal-Teacher. More comprehensively, at each simulation step: the embedded near-optimal controller (i.e., Teacher) generates an action, and the Human (i.e., trainee) controller does too. The difference between them determines the evaluation of the Human. The trainee controller's action is then applied.

The TGA learning approach is built upon the GGA approach (presented in Section V-A). Both the GGA and the TGA follow the same algorithmic steps apart from Step 2) which is:

Step 2) Each Human $i$ is evaluated by calculating $f'$ [see (4)] for each of its clones $j$ giving $N$ clone fitness values $f'_{ij}$, which are averaged over $N$ [see (3)]

$$f''_i = \frac{\sum_{j=1}^{N} f'_{ij}}{N} \tag{3}$$

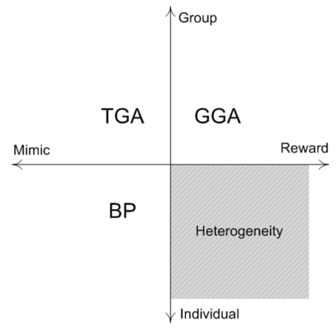


Fig. 5. The two axes that determine our research focus and classify the learning mechanisms used.

where

$$f'_{ij} = \frac{\sum_{t_s=1}^{e_p} \left\{ \left(o_{1,ij}^{t_s} - r_A^{t_s}\right)^2 + \left(o_{2,ij}^{t_s} - \alpha_A^{t_s}\right)^2 \right\}}{e_p} \tag{4}$$

and $f''_i$ is the evaluation function of Human $i$; $[o_{1,ij}^{t_s}, o_{2,ij}^{t_s}]$ is the ANN output of the clone $j$ of Human $i$ at the simulation step $t_s$; $[r_{A,j}^{t_s}, \alpha_{A,j}^{t_s}]$ are the normalized polar coordinates (i.e., distance and angle into [0, 1]) of the Animal, centered on the Human's $j$ clone, at the simulation step $t_s$ (Animal path).

We want to generate an approximately equal amount and pattern of given data for both supervised learning approaches. Thus, for the TGA experiments, we use the 80-agent environment ($N = 80$) and an evaluation period $e_p$ of eight simulation steps (i.e., $e_p = \mathrm{int}[S_t/N]$).

In contrast with the BP algorithm, the TGA approach has the advantage of retrieving real-time simulation Animal data from the whole group of agents appearing in *Flatland* instead of using a predefined data set of a single Animal. This way, the problem's designer spends minimal effort in obtaining appropriate training data sets (see Section IV). There is obviously a risk of retrieving insufficient or inappropriate data; however, it can be decreased by repeating the learning attempt. Additionally, the real-time data obtained do not necessarily represent Animals' trajectories in the environment (as in BP). Teachers may face situations that they would normally never see, because it is Humans that make the movement decision. Another crucial difference between TGA and BP is the learning environment. Agents in BP are trained individually outside the simulated world, whereas in TGA, agents are cloned and trained as a group in *Flatland*.

### C. Mechanisms' Basic Elements

The main features of the learning mechanisms presented can be demonstrated as two axes of research that this paper attempts to cover. The basic axis (horizontal in Fig. 5) determines the type of learning and the mechanisms are distinguished between those that attempt to mimic a good behavior and those that reward good features of a behavior as presented in Sections V-B and V-A, respectively. The secondary axis (vertical in Fig. 5) determines the environment in which agents learn. This feature distinguishes mechanisms where agents learn individually and mechanisms where agents learn within a group of their clones.

Learning mechanisms where agents learn individually by rewards are not covered in this paper since heterogeneity constitutes the primary prerequisite of this combination. Our learning approaches (group performance/team homogeneity) manage to cope with the agent credit assignment problem (ACAP) [45] since each agent is contributing in the same way to the group performance. Adding the possibility of heterogeneous agents in a multiagent domain adds a great deal of potential power at the price of increased complexity.

## VI. Minimizing Controller Size

We have developed a new evolutionary algorithm, called Evolving Connection Weights and Architectures Simultaneously (ECWAS), for designing ANNs automatically. This approach is inspired by the EPNet system developed by Yao and Liu [46]. ECWAS is a modified constructive algorithm that starts with a minimal ANN—one-hidden layer containing one-hidden neuron—and during the evolving process it adds new layers and neurons. Because pure constructive algorithms are susceptible to stick at structural local minima [47], the ECWAS algorithm allows deletion of layers and neurons as well. Other interesting bio-inspired approaches to minimal neural network design include Mattiussi's work [48]. According to his approach, neurons are extracted from a genome and connected according to their input and output gene encoding by applying a mapping that associates connection weights with pairs of gene sequences.

ECWAS is built upon the GGA presented in Section V-A. The only difference between the two approaches is on the mutation operator used. The modified mutation process (i.e., Step 5) of the GGA) contains three operators which occur in the sequence (see Fig. 6).

Step 5a) Connection weight mutation occurs as described in Section V-A. The mutated offspring is evaluated via (2) and compared with its parent. If the offspring is fitter, then the mutation process is terminated, else go to Step 5b).

Step 5b) A fully connected neuron is added to the network's current architecture. Both the neuron's connection weights and hidden layer are randomly selected from a uniform distribution. Once more, the mutated offspring is evaluated and compared with its parent. If the offspring is fitter, then the mutation process is terminated, else go to Step 5c).

Step 5c) A randomly selected neuron as well as its connections are deleted. At this step there is no evaluation of the offspring as it is selected by default. This way, we try to bias the search toward minimal ANN architectures. The mutation process is terminated at this step.

There is no upper bound for the number of hidden neurons in a hidden layer. On the other hand, the algorithm is constrained to design ANN architectures of the maximum number of three hidden layers. This constraint fulfils the well-known *Kolmogorov superposition theorem* [49].

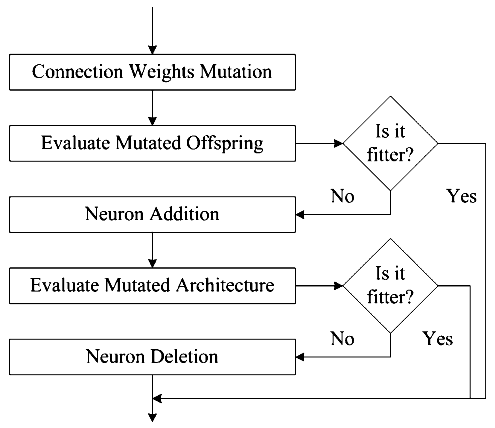

Connection Weights Mutation
Evaluate Mutated Offspring
Is it fitter?
No
Yes
Neuron Addition
Evaluate Mutated Architecture
Is it fitter?
No
Yes
Neuron Deletion

Fig. 6. The structure of the ECWAS mutation operator.

ECWAS learning mechanism is used (see Section VII-B1) to automatically draw *a priori* near-optimal ANN architectures. As experiments have shown, this algorithm constitutes an efficient preprocessing methodology for obtaining robust neural controllers of minimal size for the *Flatland* testbed.

## VII. Results

In this section, we present and compare results obtained from all learning mechanisms applied in *Flatland*, as presented in Section V. More specifically, in Section VII-A, we present a way of evaluating the performance of any experiment, in Section VII-B, we introduce a methodology to optimize the neural controller architecture—by the use of the ECWAS algorithm—and based on this we compare the performance, robustness, and effort cost of the mechanisms in the 20-agent *Flatland* environment. We expand our experiments in decreasing and growing complexity environments (Section VII-C). Finally, in Section VII-D, we approximate the ANN generalization error produced by supervised learning approaches which supports our arguments in the discussion that follows in Section VIII.

### *A. Performance Measurement*

A methodology for measuring the performance of a team of homogeneous agents is presented here. We place the agents in *Flatland* and record their total numbers of collisions $C$ and target achievements $T$ in $10^4$ simulation steps. In order to diminish the nondeterministic effect of the initialization phase (random choice of target points), we repeat the same procedure for ten simulation (i.e., evaluation) runs—we believe that this number of evaluation runs is adequate to illustrate a clear picture of the behavior—of different initial conditions, and we compute the numbers of total collisions $C_i$ and target achievements $T_i$ for each run $i$. In addition, the agents' mean speed $E\{V\}$, and the agents' mean absolute turn angle $E\{a\}$ in degrees are calculated. Subsequently, the performance $P_i$ of a team of agents in a single trial $i$ is obtained as follows:

$$P_i = \frac{\max\left\{1 - \frac{C_i}{C_{TA}}, 0\right\} + \min\left\{\frac{T_i}{T_A}, 1\right\}}{2} \quad (5)$$

where $C_{\mathrm{TA}}$ is the total number of collisions of $N$ TAs in $10^4$ simulation steps (for $N = 20, C_{\mathrm{TA}} = 2000$—see Table III); $T_A$ is the total number of target achievements of $N$ Animals in $10^4$ simulation steps (for $N = 20$, $T_A = 3200$—see Table III). The average performance over the ten trials is denoted by $P$.

The maximum value of (5) is 1.0 and it is obtained only when the agents do not collide at all and achieve as many target points as the Animals do $(T_A)$ or more. Additionally, the upper bound for the total number of collisions is the number that TAs produce $(C_{\mathrm{TA}})$ because they just move directly towards their target points and therefore, present the worst collision-avoidance behavior from our viewpoint. Hence, (5) produces a clear picture of how far the performance of each learning mechanism is from the optimal performance of Animals $(P = 1.0)$.

### *B. 20-Agent Flatland Environment*

Experiments presented in this subsection are tested in the 20-agent *Flatland* environment (i.e., $N = 20$). This environment constitutes the fundamental testbed for every investigation in *Flatland*.

*1) Optimal One-Hidden Layer Neural Architecture:* In order to efficiently compare every learning mechanism employed in *Flatland*, there is first a need for optimally designing the architecture of the neural controller. As previously mentioned in Section I, one of our main objectives is the building of minimal controllers of high performance.

We use a modified version of the ECWAS algorithm (see Table II for the algorithm's parameters) which constrains the search to one hidden layer and attempts to find minimal neural topologies for the *Flatland* case study. Even though this modification decreases the search space, it does not significantly affect the overall performance of the produced behavior. This hypothesis is verified experimentally since one-hidden layer neural architectures can produce behaviors of high performance (see Section VII-B2).

For finding the optimal one-hidden layer architecture, we experiment in the 20-agent *Flatland* environment by applying the following procedure: a) repeat the modified ECWAS learning attempt (run) 40 times (each time, a different random initialization of the connection weights' values is given); b) measure the performance of each run (see Section VII-A); and c) calculate the number of runs that present higher performance than specific performance threshold values (i.e., $P > P_{\mathrm{th}}$) for each neural architecture generated by the modified ECWAS. This number determines the success of the neural architecture for its corresponding performance threshold. The higher the performance threshold value, the more demanding the procedure.

Fig. 7 illustrates the outcome of the aforementioned procedure. The occurrences of the architectures automatically designed demonstrate that the modified ECWAS algorithm tends to find several different moderate size ANN architectures (i.e., fewer than 14 hidden neurons) capable of generating high-performance behaviors $(P > 0.9)$. On the other hand, experiments with the original three-hidden layer search ECWAS algorithm obtain a large variety of neural architectures which does not allow for clear conclusions to arise. Hence, the search space restriction helps ECWAS to investigate fewer architectures of

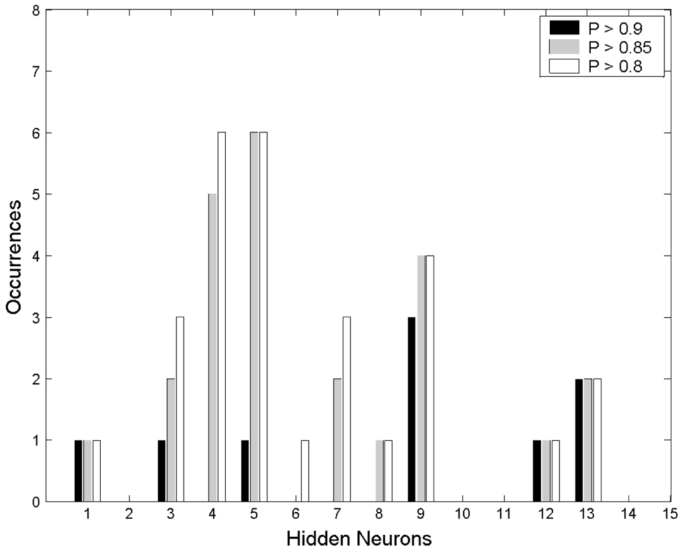


Fig. 7. Occurrences of one-hidden layer architectures found by the ECWAS algorithm.

TABLE I
MEAN PERFORMANCE $E\{P\}$ AND VARIANCE $\sigma^2$ OF THE FIVE MOST FREQUENT ONE-HIDDEN LAYER NEURAL ARCHITECTURES FOUND BY ECWAS

| Occurrences | Hidden Neurons | $E\{P\}$ | $\sigma^2$ $(\cdot 10^{-4})$ |
|---|---|---|---|
| 7 | 4 | 0.8520 | 22.7 |
| 6 | 5 | 0.8862 | 3.5 |
| 5 | 9 | 0.8782 | 29.9 |
| 4 | 7 | 0.8286 | 72.1 |
| 4 | 13 | 0.6418 | 916.2 |

potential near-optimal controllers and, therefore, makes the optimal neural architecture selection a much easier task.

To choose the optimal among ECWAS' generated neural controllers, we sort them in frequency order and calculate their occurrences' mean performance and variance (presented in Table I). We empirically select the architecture with five hidden neurons based on frequency, performance, and robustness criteria. This specific architecture gets the highest mean performance and the lowest variance between its occurrences (i.e., most robust architecture). Even though, the neural architecture with four hidden neurons is the most frequent, it does not produce high-performance results ($P > 0.9$, see Fig. 7), hence it is not selected.

To ensure that the selected architecture defines the optimal neural structure for gradient-based algorithms as well, we run the approach BP ten times for every moderate sized (i.e., fewer than 15 hidden neurons) one-hidden neural architecture. Results obtained from this procedure show that the most efficient and robust ANN architecture is again the one containing five hidden neurons. This architecture achieves the highest mean performance ($E\{P\} = 0.6403$) and lowest variance ($\sigma^2 = 0.016$) of any one-hidden layer architecture examined. Note that, the aforementioned procedure can be applied only for moderate sized one-hidden layer ANNs. For ANNs of two or more hidden layers it is computationally expensive to investigate all possible architectures. Therefore, the ECWAS algorithm is the preferred method for selecting near-optimal neural architectures of that size because of its ability to automatically design them.

The fully connected ANN with five hidden neurons in one hidden layer is the architecture used for the experiments presented in this paper. This controller proves to be the minimal sized and most efficient one-hidden ANN architecture produced from both genetic (ECWAS) and gradient (BP) search algorithms.

*2) Best Performance Comparison:* Table III illustrates the best obtained performances from all learning mechanisms applied in the 20-agent *Flatland* environment. The performance of a species of agents called "Random" ($P = 0.0010$) is also presented. These agents are randomly initialized Humans and the variance of their performance over the ten evaluation runs $\sigma^2$ equals $12.03 \cdot 10^{-7}$. The Random agents along with the TAs and the Animals are presented in Table III for comparison to any emergent Humans' behavior.

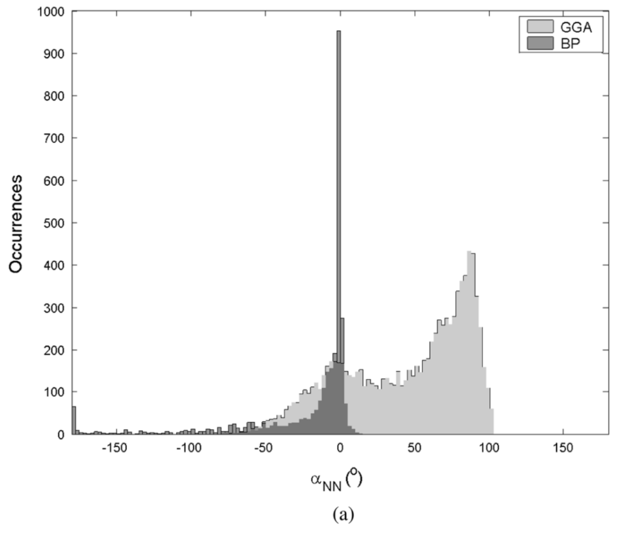
It is obvious that the GGA approach ($P = 0.9347, \sigma^2 = 12.5 \cdot 10^{-5}$) gets much closer to the desired behavior (i.e., Animals) than any other learning mechanism or any other "species" of agents. On the other hand, the best supervised learning performance, which is achieved by the BP approach, equals 0.8219. This large performance difference derives from the evidence that any "learning by rewards" attempt—in contrast with the approaches that attempt to mimic the Animal's behavior—generates Humans that manage to keep a big distance from each other in order to avoid collisions (see Fig. 9). Such behavior is characterized by large turn angles [see Fig. 8(a)], motion at almost maximum speed[1] [see Fig. 8(b)] and both frequent and large speed differences at each time step [see Fig. 8(c)].

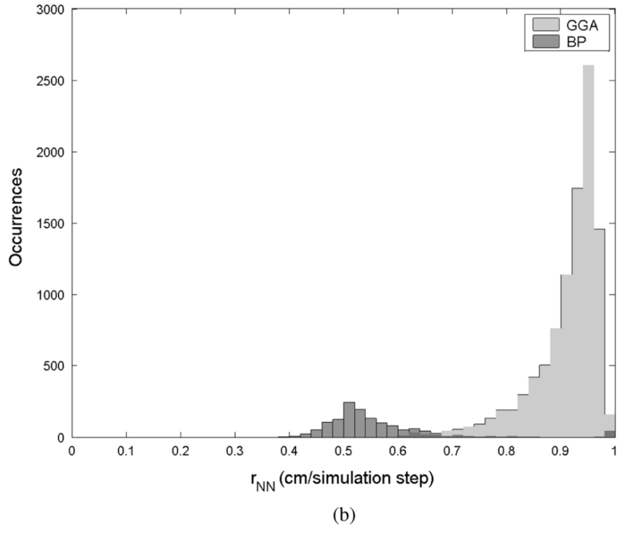
On the other hand, by observing simulations of supervised mechanisms' emerged Humans (see Fig. 8 for the BP approach) it appears that the agents in their attempt to mimic Animals behave as a TA-Animal hybrid. Small turn angles [see Fig. 8(a)], speed that approximates 0.5 cm/(simulation step) [see Fig. 8(b)] and low variation in speed change over the simulation steps [see Fig. 8(c)] illustrate the supervised trained behaviors which yield low performance.

These results lead to the important conclusion that simple evolutionary learning mechanisms can produce much better behaviors than those produced by exhausting supervised learning approaches in *Flatland*. Such successful solutions manage to exploit cooperative features built on the partial and implicit spatial communication amongst Humans. For the above experiments, the five-hidden neuron controller is used which, apart from the evidence presented in Section VII-B1, experimentally generates the best performance among all one-hidden layer feedforward neural controllers with up to 15 hidden neurons for all learning mechanisms.

*3) Robustness Comparison:* We are interested in obtaining a successful and robust learning mechanism with minimum efforts in our experiments. We can obviously experiment with parameter value adjustment of each mechanism, and can therefore find more effective neural controllers for the desired behavior.

[1]Twenty Animals designed to move at the average speed of the GGA agents [0.9 cm/(simulation step)] achieve 5401 target points and collide 1048 times; $P = 0.7380$.





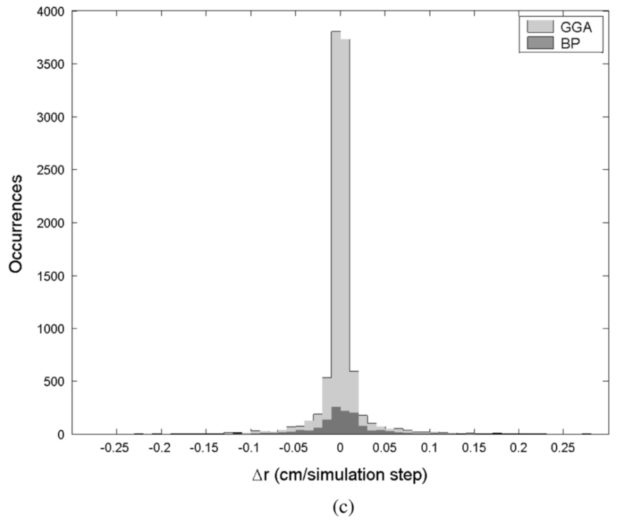


Fig. 8. Angle, speed, and speed difference histograms of a Human generated from both the GGA and the BP approach (simulation of $10^4$ steps). The dark gray color (not appearing in the legend), which indicates overlapping data, is produced due to the transparency of the colors used. (a) Angle; 6672 occurrences of $\alpha_{\rm NN} = 0$ for BP are not included. (b) Speed; 8495 occurrences of $r_{\rm NN} = 0.5$ for BP are not included. (c) Speed differences; 5925 occurrences of $\Delta r_{\rm NN} = 0$ for BP are not included.

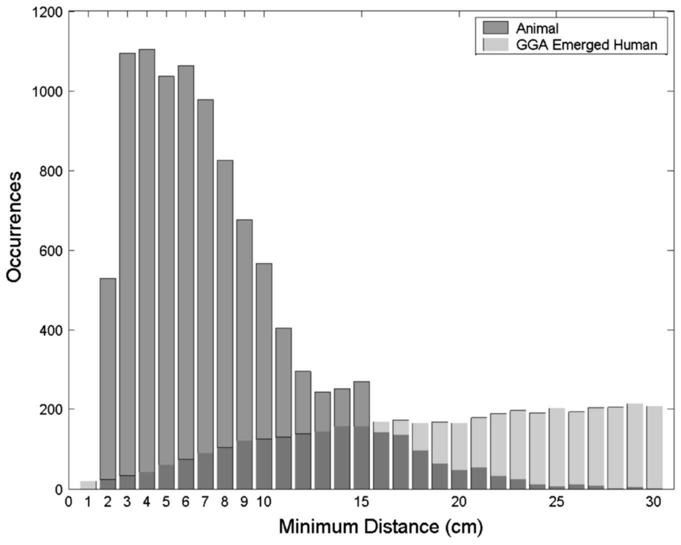


Fig. 9. 20-agent environment: Minimum distances' occurrences of an Animal and a Human emerged from GGA in $10^4$ simulation steps. The dark gray color (not appearing in the legend), which indicates overlapping data, is produced due to the transparency of the colors used.

However, if a successful controller is determined with the lowest computing cost, the applied methodology can be recommended.

To determine the effort that each learning mechanism has required to obtain a desirable robust neural controller, we assume that a single independent experiment is repeatedly run until a successful neural controller is found. A more efficient mechanism will have a smaller number of runs to find a successful neural controller [50], [51]. To test the robustness of the solutions given and to calculate the effort cost of each approach, we apply the following procedure. For each approach: a) repeat the learning attempt ten times; b) measure the performance of each run; and c) calculate the successes of the approach for a specific performance threshold (i.e., number of runs that present higher performance than the threshold value). Fig. 10 illustrates the number of successes of all learning mechanisms applied for ten values of $P_{\rm th}$. The approaches' parameters are the same as the experiment parameters presented in Section VII-B2 (see Table II).

The GGA is the most efficient and robust approach for every performance threshold (see Fig. 10). It even generates controllers (three successes) with $P \geq 0.9$, while the BP and TGA approaches' best performance is below 0.85 and 0.6, respectively. Note that, for $P_{\rm th} = 0.7$, GGA is 100% successful (i.e., ten out of ten times), while BP and TGA succeed only five and zero times, respectively. Thus, for $0.75 \leq P_{\rm th} \leq 0.8$, the effort cost of GGA can only be compared with that of BP because TGA fails completely (zero successes). Finally, for $P_{\rm th} \geq 0.85$, there cannot be any effort cost comparison between the mechanisms since GGA is the only approach capable of producing behaviors of that high performance.

Apart from supervised learning mechanisms, GGA also appears to be more robust and efficient than probabilistic approaches of evolutionary computation. Comparative studies [4] between GGA and estimation of distribution algorithms

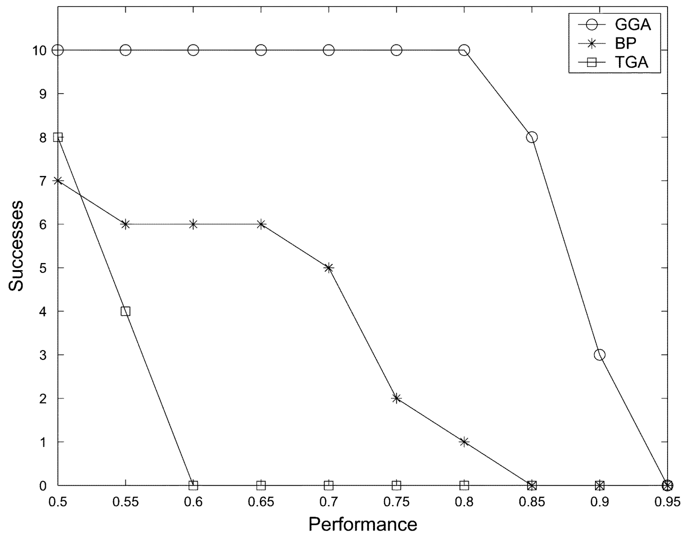


Fig. 10. Number of successes out of ten runs for specific performance values.

TABLE II
EXPERIMENT PARAMETERS FOR THE 20-AGENT ENVIRONMENT

| Learning Mechanism | Parameters |
|---|---|
| BP | $S_t = 666$ |
| TGA | N = 80, $p_m = 0.01$, $e_p = 8$, |
| GGA (ECWAS) | N = 20, $p_m = 0.01$, $e_p = 300$, $\mathrm{N_p} = 20$, $\mathrm{N_s} = 10\%$, $\mathbf{T} = 2000$ |

TABLE III
BEST PERFORMANCE COMPARISON TABLE—AVERAGE VALUES ARE OBTAINED FROM TEN EVALUATION RUNS ($10^4$ SIMULATION STEPS EACH) OF A 20-AGENT ENVIRONMENT

| Agents | $E\{C\}$ | $E\{T\}$ | $E\{\mathrm{r}\}$ | $E\{\alpha\}$ | $P$ |
|---|---|---|---|---|---|
| Random | 198620 | 7 | 0.46 | 174° | **0.0010** |
| TAs | 2000 | 3348 | 0.50 | 0° | **0.5000** |
| Animals | 0 | 3200 | 0.5 | 3.2° | **1.0000** |
| BP | 663 | 3121 | 0.51 | 7.8° | **0.8219** |
| TGA | 1513 | 2890 | 0.51 | 3.8° | **0.5733** |
| GGA | 261 | 3376 | 0.9 | 44.5° | **0.9347** |

(EDAs) [52] have demonstrated the GGA supremacy over evolutionary computation variants in the *Flatland* case study.

*4) Effort Cost Comparison:* Since the GGA approach is demonstrated to be more robust than the supervised learning approaches used, the next step is to compare these mechanisms via their effort cost interval (ECI) and mean effort cost (MEC). Hence, we pick decent high values of $P_{\mathrm{th}}$ (i.e., $P_{\mathrm{th}} \geq 0.7$) and proceed with a beta-distribution approximation (see the Appendix) of the ECI and the MEC [50], [51] for all three approaches. Learning mechanisms that experience zero successes out of ten runs are not considered in effort cost analysis.

Results from the effort cost comparison are presented in Table IV. More comprehensively, for each $P_{\mathrm{th}}$ value the number of successes ($\zeta$) and failures ($\eta$) of each approach is presented (as illustrated in Fig. 10). By use of (A.7) the lower and upper bound probability $\chi_l, \chi_u$ for each method is found; then the 95% confidence interval $[1/\chi_u, 1/\chi_l]$ is calculated. This interval represents the 95% confidence bounds on the

TABLE IV
EFFORT COST COMPARISON TABLE ($\varepsilon = 0.05$) $Q_{\mathrm{BP}} = 93.58$ s, $Q_{\mathrm{GGA}} = 796.12$ s

| $P_{th}$ | Approach | $\zeta$ | $\eta$ | ECI[2] | MEC[2] |
|---|---|---|---|---|---|
| 0.7 | BP | 5 | 5 | $[122.14, 400.26]$ | 205.87 |
| | GGA | 10 | 0 | $[797.96, 1113.30]$ | 875.73 |
| 0.75 | BP | 2 | 8 | $[180.73, 1554.49]$ | 514.69 |
| | GGA | 10 | 0 | $[797.96, 1113.30]$ | 875.73 |
| 0.8 | BP | 1 | 9 | $[226.70, 4104.39]$ | 1029.38 |
| | GGA | 10 | 0 | $[797.96, 1113.30]$ | 875.73 |
| 0.85 | GGA | 8 | 2 | $[847.12, 1651.02]$ | 1094.66 |
| 0.9 | GGA | 3 | 7 | $[1305.76, 7283.81]$ | 2919.10 |

[2]in seconds

TABLE V
$P$ AND GGA PARAMETERS FOR THE 10-, 40-, AND 80-AGENT ENVIRONMENT

| N | $P$ | | GGA | | |
|---|---|---|---|---|---|
| | $C_{TA}$ | $T_A$ | $e_p$ | $C_u$ | $T_u$ |
| 10 | 440 | 1650 | 1000 | 44 | 165 |
| 40 | 8000 | 6350 | 200 | 160 | 127 |
| 80 | 30000 | 8000 | 150 | 450 | 120 |

expected number of runs required to achieve the first successful outcome. The next column of Table IV shows the ECI $[Q_A/\chi_u, Q_A/\chi_l]$ for each approach, where $Q_A$ corresponds to the unit computing cost per run of the approach $A$. For the experiments presented here, $Q_A$ equals the average CPU time of the ten runs (every experiment presented here ran in the same 1 GHz processor). Finally, the mean effort cost is calculated with $(\zeta + \eta + 1/\zeta)Q_A$.

The important conclusion that arises from Table IV is that the BP approach is computationally preferred for low performance values (i.e., $P_{\mathrm{th}} \leq 0.75$) from the GGA approach. In other words, if there is need for a fast, relevantly low performance solution (i.e., $P_{\mathrm{th}} \leq 0.75$), the BP approach seems to be the most appropriate method. On the other hand, for $P_{\mathrm{th}} = 0.8$ the learning mechanism's effort cost interval and mean effort cost show a computational preference for the GGA approach against the BP competing approach. As previously stressed, only the GGA is capable of producing high-performance behaviors (i.e., $P_{\mathrm{th}} \geq 0.85$), and therefore no effort comparison can be explored for such demanding solutions.

Note that for both GGA and BP algorithms the effort cost estimates for lower values of $P_{\mathrm{th}}$ are likely to be overestimates: running the method until its performance value equals $P_{\mathrm{th}}$ will terminate earlier than the fixed length computation used here.

## C. Growing Flatland—Increasing Complexity

The effort cost analysis described in Section VII-B4 shows a clear distinction of the BP and GGA approaches against the TGA learning mechanism when applied in the 20-agent *Flatland* environment. However, in order to draw the overall picture of the aforementioned approaches' behavior in the *Flatland* problem, we need to experiment in less or more complex testbed environments.

Complexity depends on the number of agents in the environment (see Section I). Conceptually, the more crowded the

*Flatland* environment, the greater the probability of collisions. Consequently, given the Animals' strategy, the target-achievements $T_A$ increase logarithmically (and not linearly) with respect to $N$ (see Table V) because more collision-avoidance movements are required.[2] On the other hand, the TAs (no collision-avoidance strategy) total collisions $C_{\text{TA}}$ increase quadratically and their total target-achievements increase linearly (see Table V). Both behaviors lead to worse performance with respect to $N$, which gives an indication of the problem's complexity. For the experiments presented in this section, we pick the successful, in the 20-agent environment, BP and GGA approaches and test them in the environments of 10, 40, and 80 agents.

We follow the same procedure described in Sections VII-A, VII-B3, and VII-B4 for each one of the three environments. The experiment parameters that differ from the ones used in the 20-agent environment (see Table II) are presented in Table V.

Note that, for the BP method we are testing the same ten learning attempts for all environments. This procedure is followed since BP is trained in the most complex 80-agent environment. Furthermore, BP training attempts on data picked from less crowded environments perform much worse behaviors that simulate a TA's behavior (see Section V-B1).

Results obtained from these experiments are presented in Fig. 11. More comprehensively, the performance and the successes of both approaches for every testbed environment are illustrated in Figs. 11(a) and (b), respectively. Following the methodology presented in Section VII-B4, Table VI illustrates the effort cost comparison between BP and GGA $(P_{\text{th}} \geq 0.7)$ for the three environments.

The conclusion that arises from Fig. 11 is the absolute supremacy of the GGA over the BP learning mechanism for every environment tested. GGA manages to get high-performance behaviors $(P_{\text{th}} \geq 0.85)$ even in the 80-agent environment whereas in the same environment BP generates behaviors of very poor performances (even though it is trained on Animal data of this crowded environment). By observing the BP approach's behavior, it appears that the more complex the problem gets, the harder the ANN generalization becomes.

Given the obtained results and analysis, GGA seems to be the most robust learning mechanism applied in *Flatland*. This mechanism's overall performance remains at very high levels even in the 80-agent *Flatland* environment [see Fig. 11(a)]. Additionally, its required computational cost makes it preferable for high performance emergent solutions for every *Flatland* environment (see Table VI). As previously seen in experiments from the 20-agent environment (Section VII-B), GGA generates Humans capable of staying far away from each other and moving at almost maximum speed in all environments tested. This form of cooperation emerges due to the fact that Humans are trained to behave as a group of homogeneous agents and is built on implicit and partial communication.

## D. *Estimating Generalization Error*

In this section, we provide detailed evidence for the low-performance behaviors generated from both supervised learning mechanisms applied in *Flatland*. In order to do that, we approximate and simulate the error deviations in motion from the Animals' strategy produced by these mechanisms' emerged Humans as follows. Twenty Animals are placed in *Flatland* and simulated for $10^4$ steps. At each simulation step during which the Animal is interacting with another (i.e., its potential (1b) is significantly nonzero), we add an error vector to the Animal's position. The outcome of this experiment is to ascertain the performance of these Animals over the total mse produced by the error vector. This vector is given by either of the following.

1) **Circle perimeter error**: The error vector is defined by a random point in the perimeter of a circle centered on the Animal's position. The circle's radius constitutes the independent variable of the experiment.
2) **Gaussian noise**: The error vector is randomly picked from a Gaussian distribution. For each Cartesian dimension, the mean of this Gaussian distribution equals the Animals's

[2]Animals do not collide and achieve $T_A$ targets and according to (5) they generate an optimal performance value of 1.0 in all environments tested.

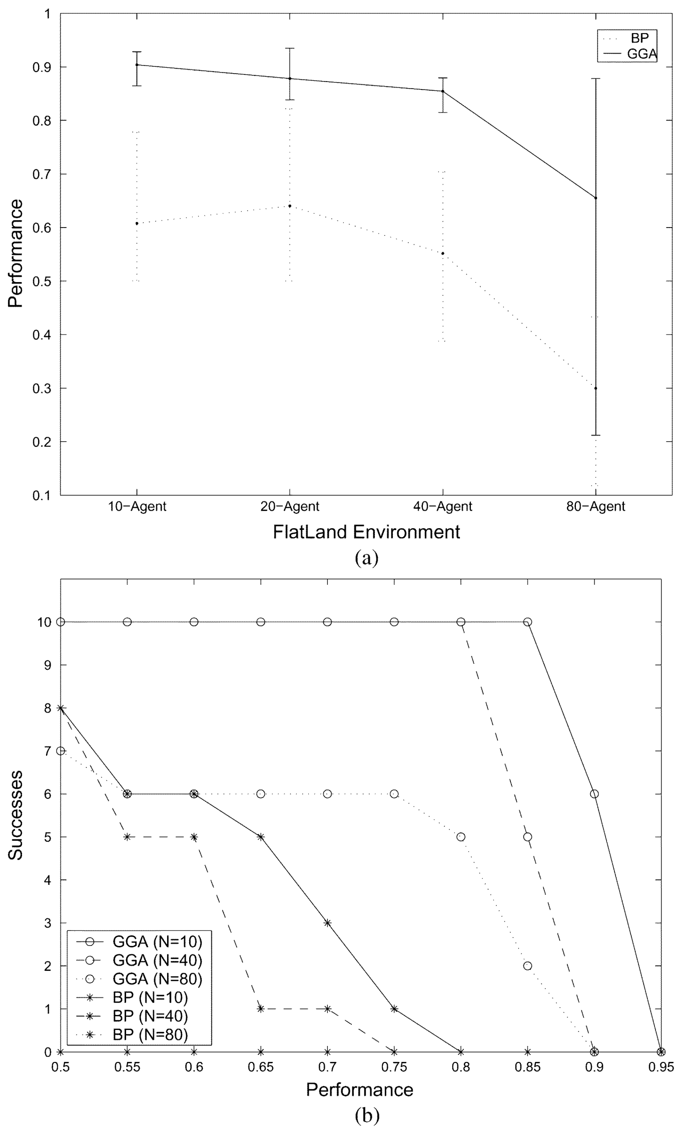


Fig. 11. Experiments over the number of agents appearing in *Flatland*. (a) Mean, best and worst performance of both mechanisms over the *Flatland* increasing population. (b) Number of successes out of ten runs for specific performance values for both mechanisms.

TABLE VI
EFFORT COST COMPARISON TABLE ($\varepsilon = 0.05$)$Q_{\mathrm{BP}} = 93.58$ s, $Q_{\mathrm{GGA}} = 457.28$ s FOR THE 10-AGENT ENVIRONMENT; $Q_{\mathrm{GGA}} = 763.83$ s FOR THE 40-AGENT ENVIRONMENT

| N | $P_{th}$ | | $\zeta$ | $\eta$ | ECI | MEC |
|---|---|---|---|---|---|---|
| 10 | 0.7 | BP | 3 | 7 | [153.5, 856.2] | 343.1 |
| | | GGA | 10 | 0 | [458.3, 639.5] | 503.1 |
| | 0.75 | BP | 1 | 9 | [226.7, 4104.4] | 1029.3 |
| | | GGA | 10 | 0 | [458.3, 639.5] | 503.1 |
| 40 | 0.7 | BP | 1 | 9 | [226.7, 4104.4] | 1029.3 |
| | | GGA | 10 | 0 | [765.6, 1068.1] | 840.2 |

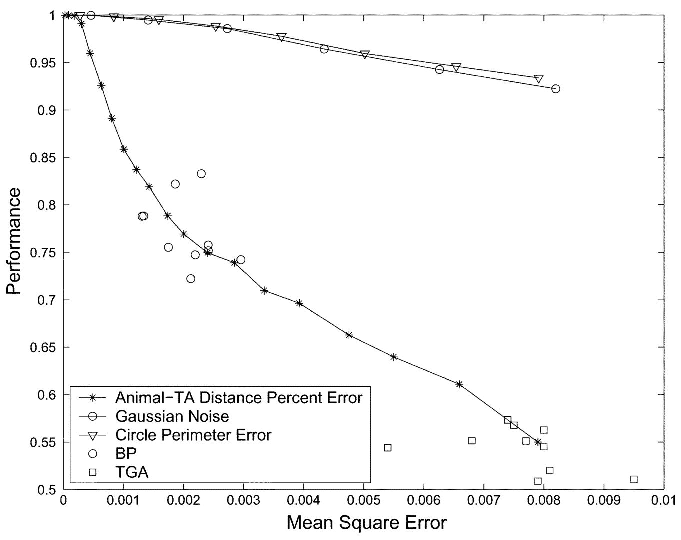


Fig. 12. Performance versus mse of 20 Animals in a noisy *Flatland* environment. BP and TGA performances (ten trials) over their training mse and fitness value (mse), respectively, are added for illustration.

coordinate. The standard deviation of the Gaussian noise constitutes the independent variable of the experiment.

3) **Animal-TA distance percent error**: For each Animal, we calculate the motion vector of a TA placed on the Animal's position. The error vector defines a step motion from the Animal's to the TA's position. The percentage of this distance (i.e., Animal-TA) constitutes the independent variable of the experiment. As this percentage increases the Animal's behavior transforms into a TA's behavior.

The outcome of this experiment is illustrated in Fig. 12. The *Animal-TA distance percent error* way of adding errors in the Animal's vector motion seems to be the best approximation of the motion errors that the supervised learning mechanisms produce. More analytically, Fig. 12 illustrates ten trials of well-behaved ($P > 0.5$) agents generated from both BP and TGA mechanisms. It appears that both approaches generate Humans that follow the approximation line produced by the *Animal-TA distance percent error* way of adding error vectors to the Animal motion (see Fig. 12).

Given that, we can assert that supervised learning mechanisms, in their attempt to mimic the Animal's behavior, generate suboptimal Humans that introduce crucial motion errors into the Animals near-optimal strategy. Thus, these methods fail to generate behaviors of high performance. This statement supports the observed visualized behavior of Humans produced from both supervised learning mechanisms which behave like a TA-Animal hybrid (see also Fig. 8). Furthermore, given their best mse performance, we can assume that such mechanisms are not likely to achieve high-performance behaviors that GGA produces. Finally, another important conclusion that arises from Fig. 12 is that even slight errors in the motion of the optimal "species" of Animals, may lead to a great number of collisions, and therefore poor performance behaviors.

## VIII. CONCLUSION

We introduced a top-down approach for obtaining robust neurocontrollers for cooperative spatial coordination with minimal effort through a comparison of various learning mechanisms. The prototype *Flatland* world was utilized for testing and applying our methodology. This case study shares common features of known (mainly artificial life and computer games) worlds used for studying the emergence of cooperative global behaviors which are based on local interactions. In addition, agents are explicitly given individual tasks and their communication is limited to "seeing" neighbor agents.

We saw that simple mutation-based evolutionary algorithms can generate high-performing and robust cooperative behaviors within the *Flatland* context. The learning ability of these algorithms is based on rewarding the overall behavior of a homogeneous team of clone agents. More specifically, the GGA approach proved to be the most robust and least computationally expensive method for every *Flatland* environment tested. On the other hand, supervised learning mechanisms failed to compete with the "learning by rewards" approaches.

Suppose that agents learn to behave within a group that consists of copies of themselves. This results in the emergence of an interesting form of abstract cooperation between the agent and its clones, which increases the global efficiency (performance). This is exactly what is demonstrated by the unsupervised learning approach applied. Conversely, in supervised learning, an agent is initially self-trained (e.g., BP) and then it clones itself to form a group. This procedure apparently does not leave any space for emergence of self-clone cooperation.

The question that arises here is: "Is it the mechanism itself (unsupervised, supervised) or the learning environment (clonal, individual) that allows cooperation to appear?" In other words, which of the two axes illustrated in Fig. 5 contributes the most to the generation of efficient solutions. The answer is supported through the evidence that supervised learning even within a team (i.e., TGA approach) did not manage to compete with the "learning by rewards" method. Thus, it appears that reinforcing good solutions is the key factor that affects cooperation rather than the learning environment used.

Moreover, no attempt to choose the most appropriate set of data (i.e., perception and action of Animals) for agents to learn from, proved able to outperform unsupervised learning. The complex dynamics of the environment and the strong training data set dependence constitute major obstacles towards a well performing solution. We saw that both mechanisms attempting to mimic the Animals' behavior managed to output suboptimal (i.e., TA-Animal hybrid) behaviors. This is explained through the confirmed hypothesis that even slight differences from the near-optimal hand-coded strategy can cause collisions, and therefore decrease the performance of a team of agents (see Section VII-D).

To summarize, suppose that a) we deal with fully dynamic multiagent environments where the communication of agents is passive and based on partial and implicit information; b) we want the agent-controllers to learn to behave cooperatively for the successful achievement of specific tasks (e.g., spatial coordination); c) there is a near-optimal (good) handcrafted behavior available; d) we have to make a choice between a supervised learning mechanism that attempts to mimic the good behavior and a mechanism that rewards the overall behavior of a group of agents; and e) in both types of mechanisms the performance evaluation is based on the behavior of a homogeneous group of generated solutions (agents). Given these, a "learning by rewards" approach tends to perform better by producing cooperative features within the emergent solutions. In addition, these solutions tend to be more robust than and computationally preferable to solutions generated by mimicking.

## Appendix
## Beta-Distribution

When it is assumed that $\zeta+\eta$ independent experiments of the approach $A$ experience $\zeta$ successes and $\eta$ failures, the distribution of success rate $p$ can be approximated with a Beta distribution. The Beta probability density function is given by

$$f(\zeta,\eta,p)=\frac{1}{B(\zeta+1,\eta+1)}p^{\zeta}(1-p)^{\eta} \qquad \text{(A.6)}$$

where $B(\zeta+1,\eta+1)=\int_0^1 p^{\zeta}(1-p)^{\eta}dp=(\zeta!\eta!/(\zeta+\eta+1)!)$. Assume that a random variable $X$ with a Beta distribution has the upper bound $\chi_u$ and the lower bound $\chi_l$ for confidence limits such that $P(\chi_l < X < \chi_u)=1-\varepsilon$ and $P(X \leq \chi_l)=\varepsilon/2$, where $\varepsilon$ is the confidence coefficient (in all results presented here $\varepsilon=0.05$). Then, we can assert that $[\chi_l,\chi_u]$ is a $(1-\varepsilon)\cdot 100\%$ confidence interval. If a probability $p$ is Beta distributed, the confidence limits $\chi_l,\chi_u$ can be obtained by solving the equations

$$\frac{\varepsilon}{2}=\int_0^{\chi_l}\frac{p^{\zeta}(1-p)^{\eta}}{B(\zeta+1,\eta+1)}dp \qquad \text{(A.7a)}$$

$$\frac{\varepsilon}{2}=\int_{\chi_u}^{1}\frac{p^{\zeta}(1-p)^{\eta}}{B(\zeta+1,\eta+1)}dp. \qquad \text{(A.7b)}$$

From the lower and upper bound probability $\chi_l,\chi_u$, the 95% confidence effort cost can be estimated with $[(1/\chi_u)Q_A,(1/\chi_l)Q_A]$, where $Q_A$ is the unit computing cost per run of the approach $A$ (i.e., in our experiments, $Q_A$ equals CPU time). Finally, the mean effort cost can be obtained from $(\zeta+\eta+1/\zeta)Q_A$.

## References

[1] J. A. Meyer and A. Guillot, "From SAB90 to SAB94: Four years of animat research," in *Proc. 3rd Int. Conf. Simulation Adaptive Behav., From Animals to Animats 3*, 1994.

[2] Z. Ganon, A. Keinan, and E. Ruppin, "Evolutionary network minimization: Adaptive implicit pruning of successful agents," in *Proc. 7th Eur. Conf. Artif. Life (ECAL), Advances Artif. Life*, W. Banzhaf, T. Christaller, P. Dittrich, J. T. Kim, and J. Ziegler, Eds., 2003, pp. 319–327.

[3] G. N. Yannakakis, J. Levine, J. Hallam, and M. Papageorgiou, "Performance, robustness, and effort cost comparison of machine learning mechanisms in Flatland," in *Proc. 11th Mediterranean Conf. Control Automation MED'03*, Jun. 2003.

[4] G. N. Yannakakis, J. Hallam, and J. Levine, "Evolutionary computation variants for cooperative spatial coordination," in *Proc. Congr. Evol. Comput.*, Edinburgh, U.K, Sep. 2005, pp. 2715–2722.

[5] C. W. Reynolds, "Flocks, herds, and schools: A distributed behavioral model," *Computer Graphics*, vol. 21, no. 4, pp. 25–34, 1987.

[6] T. Balch and R. C. Arkin, "Behavior-based formation control for multirobot teams," *IEEE Trans. Robot. Autom.*, vol. 14, no. 6, pp. 926–939, Dec. 1998.

[7] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *Int. J. Robot. Res.*, vol. 5, no. 1, pp. 90–99, 1986.

[8] F. Flacher and O. Sigaud, "Basc, a bottom-up approach to automated design of spatial coordination," in *Proc. 7th Int. Conf. Simulation Adaptive Behav., From Animals to Animats 8*, S. Schaal, A. Ijspeert, A. Billard, S. Vijayakumar, J. Hallam, and J.-A. Meyer, Eds., 2004, pp. 435–444.

[9] F. Flacher and O. Sigaud, "Spatial coordination through social potential fields and genetic algorithms," in *Proc. 7th Int. Conf. Simulation Adaptive Behav., From Animals to Animats 7*, B. Hallam, D. Floreano, J. Hallam, G. Hayes, and J.-A. Meyer, Eds., 2002, pp. 389–390.

[10] Icosystem, "The Game." [Online]. Available: http://www.icosystem.com/game.htm

[11] L. E. Parker, "Designing control laws for cooperative agent teams," in *Proc. IEEE/ICRA*, 1993, pp. 582–587.

[12] G. M. Werner, "Evolution of herding behavior in artificial animals," in *Proc. 2nd Int. Conf. Simulation of Adaptive Behav., From Animals to Animats 2*, J.-A. Meyer, H. L. Roitblat, and S. W. Wilson, Eds., 1993, pp. 108–115.

[13] G. Miller and D. Cliff, "Protean behavior in dynamic games: Arguments for the co-evolution of pursuit-evasion tactics," in *Proc. 3rd Int. Conf. Simulation Adaptive Behav., From Animals to Animats 3*, D. Cliff, P. Husbands, J.-A. Meyer, and S. W. Wilson, Eds., 1994, pp. 411–420.

[14] S. Luke and L. Spector, "Evolving teamwork and coordination with genetic programming," in *Proc. 1st Annu. Conf. Genetic Program.*, J. R. Koza, D. E. Goldberg, D. B. Fogel, and R. L. Riolo, Eds., 1996, pp. 150–156.

[15] J. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA: MIT Press, 1992.

[16] J. Togelius and S. Lucas, "Forcing neurocontrollers to exploit sensory symmetry through hard-wired modularity in the game of cellz," in *Proc. IEEE Symp. Comput. Intell. Games*, G. Kendall and S. Lucas, Eds., Colchester, U.K., Apr. 4–6, 2005, pp. 37–43.

[17] R. Salustowicz, M. Wiering, and J. Schmidhuber, "Evolving soccer strategies," in *Proc. 4th Int. Conf. Neural Inf. Proc., Progress in Connectionist-Based Inf. Syst.*, N. Kasabov, R. Kozma, K. Ko, R. O'Shea, G. Coghill, and T. Gedeon, Eds., 1997, vol. 1, pp. 502–505.

[18] G. Baldassarre, "Cultural evolution of 'guiding criteria' and behaviour in a population of neural-network agents," *J. Memetics—Evol. Models Inf. Trans.*, vol. 4, 2001, on-line Journal.

[19] C. W. Reynolds, "An evolved, vision-based behavioral model of coordinated group motion," in *Proc. 2nd Int. Conf. Simulation Adaptive Behav., From Animals to Animats 2*, J. Meyer, H. L. Roitblat, and S. W. Wilson, Eds., 1993, pp. 384–392.

[20] N. Zaera, D. Cliff, and J. Bruten, "(Not) evolving collective behaviors in synthetic fish," in *Proc. 4th Int. Conf. Simulation Adaptive Behav., From Animals to Animats 4*, P. Maes, M. Mataric, J.-A. Meyer, J. Pollack, and S. Wilson, Eds., 1996, pp. 635–644.

[21] G. N. Yannakakis and J. Hallam, "Evolving opponents for interesting interactive computer games," in *Proc. 8th Int. Conf. Simulation Adaptive Behav., From Animals to Animats 8*, S. Schaal, A. Ijspeert, A. Billard, S. Vijayakumar, J. Hallam, and J.-A. Meyer, Eds., Jul. 2004, pp. 499–508.

[22] G. N. Yannakakis, J. Levine, and J. Hallam, "An evolutionary approach for interactive computer games," in *Proc. Congr. Evol. Comput.*, Jun. 2004, pp. 986–993.

[23] G. N. Yannakakis and J. Hallam, "A generic approach for generating interesting interactive Pac-Man opponents," in *Proc. IEEE Symp. Comput. Intell. Games*, G. Kendall and S. Lucas, Eds., Colchester, U.K., Apr. 4–6, 2005, pp. 94–101.

[24] K. Stanley, B. Bryant, and R. Miikkulainen, "Real-time evolution in the NERO video game," in *Proc. IEEE Symp. Comput. Intell. Games*, G. Kendall and S. Lucas, Eds., Colchester, U.K., Apr. 4–6, 2005, pp. 182–189.

[25] M. Dorigo, V. Trianni, E. Sahin, R. Groß, T. Labella, G. Baldassarre, S. Nolfi, J.-L. Deneubourg, F. Mondada, D. Floreano, and L. Gambardella, "Evolving self-organizing behaviors for a swarm-bot," *Autonomous Robots*, vol. 17, no. 2–3, pp. 223–245, 2004.

[26] J. Pugh and A. Martinoli, "Multi-robot learning with particle swarm optimization," in *Proc. Int. Conf. Autonomous Agents Multiagent Syst.*, Hakodate, Japan, May 2006.
[27] E. A. Abbott, *Flatland*. New York: Signet Classic, Jun. 1984.
[28] G. N. Yannakakis, "Evolutionary computation: Research on emerging strategies through a complex multi-agent environment," M.S. thesis, Technical Univ. Crete, Chania, Greece, Sep. 2001.
[29] J. D. Funge, *Artificial Intelligence for Computer Games*. Wellesley, MA: A. K. Peters, Ltd., 2004.
[30] K. Støy, "Controlling self-reconfiguration using cellular automata and gradients," in *Proc. 8th Int. Conf. Intell. Autonomous Syst.*, Amsterdam, The Netherlands, Mar. 2004, pp. 693–702.
[31] M. Scheutz and P. Schermerhorn, "Steps towards a systematic investigation of possible evolutionary trajectories from reactive to deliberative control systems," in *Proc. 8th Int. Conf. Simulation Synthesis Living Systems (ALife-8)*, 2002, pp. 283–292.
[32] R. J. Collins and D. R. Jefferson, , C. G. Langton, C. Taylor, J. D. Farmer, and S. Rasmussen, Eds., "Antfarm: Towards simulated evolution," in *Artificial Life II*. Redwood City, CA: Addison-Wesley, 1992, pp. 579–601.
[33] R. C. Arkin, "Cooperation without communication: Multiagent schema based robot navigation," *J. Robot. Syst.*, vol. 9, no. 3, pp. 351–364, Apr. 1992.
[34] S. B. Emerson and S. K. Boyd, "Mating vocalizations of female frogs: Control and evolution mechanisms," *Brain, Behavior, and Evolution*, vol. 53, pp. 187–197, 1999.
[35] D. H. Ackley and M. L. Littman, "Interactions between learning and evolution," in *Artificial Life II*. Reading, MA: Addison-Wesley, 1992, pp. 478–507.
[36] D. Cliff and S. Grand, "The creatures global digital ecosystem," *Artificial Life*, vol. 5, pp. 77–94, 1999.
[37] L. Yaeger, "Computational genetics, physiology, metabolism, neural systems, learning, vision, and behaviour of polyworld: Life in a new context," in *Proc. Workshop on Artificial Life, Artificial Life III*, C. G. Langton, Ed., 1993, vol. XVII, pp. 263–298, Sante Fe Institute Studies in the Sciences and Complexity.
[38] C. W. Reynolds, "Evolution of corridor following behavior in a noisy world," in *Proc. 3rd Int. Conf. Simulation Adaptive Behav., From Animals to Animats 3*, D. Cliff, P. Husbands, J.-A. Meyer, and S. W. Wilson, Eds., 1994, pp. 402–410.
[39] J. H. Holland, *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI: Univ. Michigan Press, 1975.
[40] G. Syswerda, "Uniform crossover in genetic algorithms," in *Proc. 3rd Int. Conf. Genetic Algorithms*, Schaffer, Ed., 1989, pp. 2–9.
[41] D. J. Montana and L. D. Davis, "Training feedforward neural networks using genetic algorithms," in *Proc. 11th Int. Joint Conf. Artif. Intell.*, 1989, pp. 762–767.
[42] X. Yao, "Evolving artificial neural networks," *Proc. IEEE*, vol. 87, no. 9, pp. 1423–1447, 1999.
[43] M. T. Hagan and M. Menhaj, "Training feedforward networks with the marquardt algorithm," *IEEE Trans. Neural Netw.*, vol. 5, no. 6, pp. 989–993, Nov. 1994.
[44] K. J. Lang and M. J. Witbrock, "Learning to tell two spirals apart," in *Proc. Connectionist Models Summer School*, 1988.
[45] G. Weiss, , G. Weiss and S. Sen, Eds., "Adaptation and learning in multi-agent systems—Some remarks and a bibliography," in *Adaptation and Learning in Multi-Agent Systems*. Berlin, Germany: Springer-Verlag, 1996, vol. 1042, Lecture Notes in Artificial Intelligence, pp. 1–21.
[46] X. Yao and Y. Liu, "Towards designing artificial neural networks by evolution," *Appl. Math. Comput.*, vol. 9, no. 1, pp. 54–65, 1996.
[47] P. J. Angeline, G. M. Sauders, and J. B. Pollack, "An evolutionary algorithm that constructs recurrent neural networks," *IEEE Trans. Neural Netw.*, vol. 5, pp. 54–65, Jan. 1994.
[48] C. Mattiussi, M. Waibel, and D. Floreano, "Measures of diversity for populations and distances between individuals with highly reorganizable genomes," *Evol. Comput.*, vol. 12, 2004.
[49] A. N. Kolmogorov, "On the representation of continuous functions of several variables by superposition of continuous functions of one variable and addition," *Doklady Akademii. Nauk USSR*, vol. 114, pp. 679–681, 1957.
[50] W. P. Lee, "Applying genetic programming to evolve behavior primitives and arbitrators for mobile robots," Ph.D. dissertation, University of Edinburgh, Edinburgh, U.K., 1998.
[51] D. Kim, "A quantitative approach to the analysis of memory requirements for autonomous agent behaviours using evolutionary computation," Ph.D. dissertation, University of Edinburgh, Edinburgh, U.K., 2002.
[52] P. Larranaga and J. A. Lozano, *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*. Norwell, MA : Kluwer, 2001.

**Georgios N. Yannakakis** (S'04–M'05) received both the five-year Diploma degree in production engineering and management and the M.Sc. degree in financial engineering from the Technical University of Crete, Chania, Greece, in 1999 and 2001, respectively, and the Ph.D. degree in informatics from the University of Edinburgh, Edinburgh, U.K., in 2005.

He is a Postdoctoral Researcher at the Mærsk Institute for Production Technology, University of Southern Denmark, Odense. His research interests include artificial life, neuroevolution, emergent cooperation within multiagent systems, AI in computer games, and entertainment modeling.

**John Levine** (M'05) is a Senior Lecturer at the University of Strathclyde and a prominent member of the Strathclyde Planning Group (led by Prof. M. Fox). He is the Principal Investigator on a recently funded EPSRC grant concerned with evolving very high-quality search control heuristics for AI planning. His current research program consists of searching for intelligent behaviors for computer-based agents, especially agents that need to plan their future activities and make complex decisions; and searching for very high-quality solutions for difficult combinatorial optimization problems.

Dr. Levine is a member of the U.K. EPSRC Review College 2003–2005 and 2006–2009 and was the Chair of the PLANSIG 2001 Conference on AI Planning and Scheduling Systems.

**John Hallam** graduated with First Class Honors in mathematics from the University of Oxford, Oxford, U.K., in 1979. He received the Ph.D. degree from the Department of Artificial Intelligence, University of Edinburgh, Edinburgh, U.K., in 1984.

He joined the teaching Faculty in the Department of Artificial Intelligence, University of Edinburgh in 1985. He established the Edinburgh Mobile Robotics Research Group, having been active in mobile robotics research for almost 20 years. In 2003, he moved to the Mærsk Institute, University of Southern Denmark, Odense. He is a Director of 3 Lions Design Ltd., a small company that does commercial electronic design. The current focus of his catholic research interest in robotics is in biological modeling using robotic techniques, evolutionary robotics, and collective robotics. He has published around 100 journal and international conference papers on various robotic and nonsymbolic computing topics, and has designed electronic hardware both for research and teaching and commercially.

Dr. Hallam is President of the International Society for Adaptive Behavior and a member of the London Mathematical Society.