Performance, robustness and effort cost comparison of machine learning mechanisms in *FlatLand*

Georgios N. Yannakakis, *Student Member, IEEE,* John Levine, John Hallam
and Markos Papageorgiou, *Fellow, IEEE*

***Abstract*— This paper presents the first stage of research into a multi-agent complex environment, called “*FlatLand*” aiming at emerging complex and adaptive obstacle-avoidance and target-achievement behaviors by use of a variety of learning mechanisms. The presentation includes a detailed description of the *FlatLand* simulated world, the learning mechanisms used as well as an efficient method for comparing the mechanisms' performance, robustness and required computational effort.**

***Index Terms*— back-propagation, genetic algorithms, machine learning, multi-agent, simulated worlds.**

I. Introduction

MULTI-AGENT systems is a prominent area of research. Designing agents for such systems could be a repetitive and tedious procedure. This task is getting even more difficult when the multi-agent environment is fully dynamic and non-deterministic. When designing controllers for autonomous simulated agents for such environments, there is little guidance on how complex the controller must be for the agent to achieve good performance in particular tasks. Furthermore, when trying to emerge such a performance via a learning mechanism, there is little knowledge about the mechanism's complexity.

We have developed a novel simulated world called “*FlatLand*” for studying genetic and gradient-search optimization techniques. The *FlatLand* world is a two-dimensional multi-agent complex environment. The agents living in *FlatLand* appear as circular artificial creatures. Our first objective in developing this world is to create a novel environment for comparing and testing various controllers and learning techniques. The two tasks that our agents are tested in are the competing strategies of obstacle-avoidance and target-achievement. The work presented here is focused on the evolution of agents' artificial controllers in emerging the aforementioned strategies in an adaptive fashion, using various forms of learning procedures.

This paper is organized as follows. In Section II we present a detailed description of *FlatLand* as well as the agents' controllers employed. In Section III we discuss about the difficulties and points of importance of this simulated world. The learning mechanisms used are analytically described in Section IV. Results obtained as well as comparison of performance, robustness and effort cost between the different learning approaches are presented in Section V. Finally, the most important conclusions of the *FlatLand* research are outlined in Section VI.

Manuscript received January 30, 2003; revised April 13, 2003.

G. N. Yannakakis and J. Levine are with the School of Informatics, Centre for Intelligent Systems and their Applications, The University of Edinburgh, EH8 9LE, UK (e-mail: g.yannakakis@sms.ed.ac.uk; johnl@inf.ed.ac.uk).

J. Hallam is with the Maersk Mc-Kinney Moller Institute for Production Technology, University of Southern Denmark, Campusvej 55, DK-5230 Odense M, Denmark (e-mail: john@mip.sdu.dk).

M. Papageorgiou is with the Department of Production Engineering and Management, Dynamic Systems and Simulation Laboratory, Technical University of Crete, Chania, Greece (e-mail: markos@dssl.tuc.gr).

II. The *FlatLand* Simulated World

The name “*FlatLand*” is inspired from E. Abbott's book title [1] and its fundamental concept is based on previous research by Yannakakis [2]. The main purpose of this simulated world is to be used as a test-bed environment for investigating evolutionary [3] and gradient-based (in a lesser degree) learning techniques and furthermore, their ability to emerge complex and adaptive obstacle-avoidance and target-achievement behaviors. In this section, we present a detailed description of this simulated world.

FlatLand is a two-dimensional multi-agent square environment. The world's dimensions are predefined so that actions take place in a closed frictionless plane. There are two simple figures visualized in *FlatLand* (as illustrated in Fig. 1): 1) white circles (radius=5mm) that represent the agents - artificial creatures and 2) dashed straight lines connecting the agent's current position and its target point on the surface.

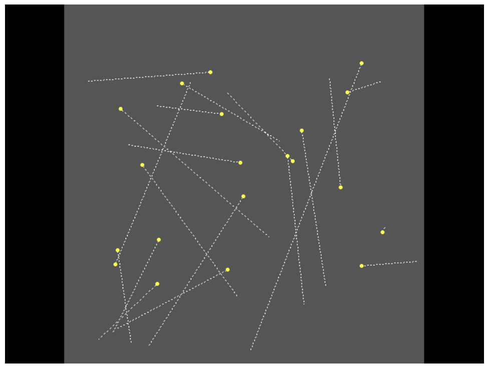

Fig. 1. *FlatLand* world interface (the plane's dimensions are 80 cm $\times$ 80 cm in the experiments presented here)

The population used consists of some twenty 2D circular agents-creatures, called “Humans”. Their motion is controlled by a neural network. It is worth mentioning that one of Humans' properties is their permeability in case of a possible collision with each other. Therefore, their motion is not affected when they collide as they pass through each other.

Even though this is an unwanted behavior (from the viewpoint of *FlatLand*'s obstacle-avoidance goal behavior), it contributes to the simplicity of the environment. As mentioned before, each Human holds a target point on the environment's surface. This point keeps changing during its life and therefore, as soon as a Human achieves its current target (i.e. manages to reach a circle of 5mm around the target point), then a new target point is selected. The new target point is picked from a uniform random distribution in a specified distance of 30cm from the agent's center. The simulation procedure of *FlatLand* can be described as follows. Humans are placed randomly in *FlatLand* (initial positions) via a uniform distribution. Then, the following occur at each simulation step:

1) Each Human gathers information from its environment (see Subsection II-A.1)
2) It takes a movement decision (see Subsection II-A.2)
3) Total number of collisions and target-achievements as well as the average speed of the Humans are recorded.
4) New randomly picked target points are given to those Humans that have achieved their target points.

FlatLand's aim is to focus and research over the agent's ability to emerge an efficient and robust obstacle-avoidance and target-achievement behavior. Therefore, the design of the simulated agents used in this environment is deliberately kept abstract. Finally, it is worth mentioning that there is no wall avoidance strategy implemented yet (it constitutes one of our future research steps).

A. Neural Controller

Neural networks appear to be the most promising means of emerging adaptive behaviors in complex multi-agent environments, as stressed in [4] and [5]. Therefore, a feedforward neural controller is employed to manage the agents' motion and is described in this subsection. Apart from the neural controller, an Artificial Potential Field employed for controlling the agents' movement is also introduced in Subsection II-B.

1) Input: Using its sensors, each Human inspects the environment from its own point of view and decides about its next action. Both the input information and the neural controller's architectures are analytically presented in this subsection.

The neural controller's input data and format can be described as follows. Each Human receives information from its environment expressed in the neural network's input array of dimension D:

$$D = 2z + 1 \tag{1}$$

where z defines the number of the closest Humans that each Human takes into account via its sensors. Thus, the input array consists of: (a) the polar coordinates $(\mathrm{a_i}, \mathrm{d_i})$ - based on the axis determined by the current position of the Human and its target point (see Fig. 2) - of the z $(\mathrm{i} = 1, \ldots, z)$ closest Humans and (b) an additional input that defines the distance between the Human's current position and its target point $(\mathrm{d_T})$. Fig. 2 illustrates the Human's sensoring information as described above.

All input values are linearly normalized into [0, 1] before they are entered into the neural controller. The input's format

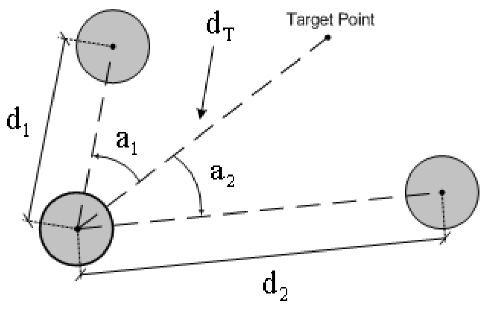

Fig. 2. Human's input data in polar coordinates $(z = 2)$

in polar coordinates is based on Reynolds' work in artificial critters [6]. For the experiments presented in this paper $z = 2$, as it stresses the minimal amount of information for a Human to successfully achieve the desired behavior (i.e. for $z = 1$ neural controllers are not able to emerge satisfactory obstacle-avoidance strategies).

2) Architecture: There has been research on many different feedforward neural network architectures. Our potential target when we first developed *FlatLand* was to find the simplest neural controller capable of emerging the desired behavior. Therefore, a two-layered fully connected feedforward neural network has been used for the experiments presented here (as shown in Fig. 3). The sigmoid function is employed at each neuron.

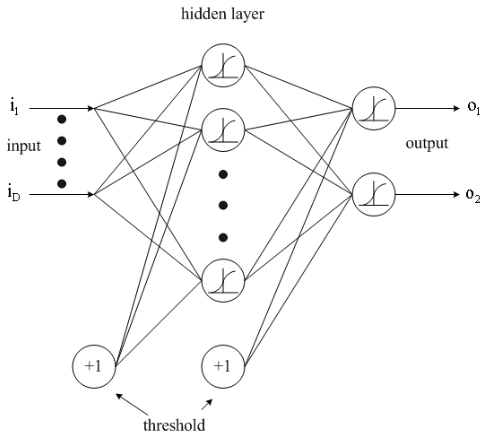

Fig. 3. Two-layer feedforward neural network controller

The connection weights take values from -5 to 5 while the neural network's output is a two-dimensional vector $[o_1, o_2]$ with respective values from 0 to 1. This vector represents the Human's step motion and is converted into polar coordinates according to (2) and (3).

$$d_{NN} = o_1 M \tag{2}$$

$$a_{NN} = (2o_2 - 1)\pi \tag{3}$$

where:
d_{NN}: Human's step motion (in cm/time step); a_{NN}: Human's turn angle from the axis determined by the Human's current position and its target point (in degrees); M: Human's maximum speed; in experiments presented in this paper, M=1 cm/time step.

B. Artificial Potential Field Strategy

Using the same environment, we explored another “species” of agents as well. These agents are called “Animals” and their only difference from Humans is in the control of their locomotion. Instead of a neural network, 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 forces and obstacles (other Animals) in the environment are repulsive forces [7]. 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=5mm) that slides down the surface plotted in Fig. 4. This surface is plotted by each

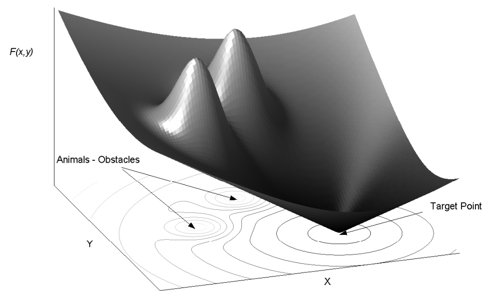

Fig. 4. APF - Situation of two obstacles - closest Animals ($z = 2$).

Animal in every simulation step and represents the function:

$$F(x,y) = \frac{M}{2}\sqrt{(x-x_{\mathrm{T}})^2+(y-y_{\mathrm{T}})^2} \tag{4a}$$

$$+\,\delta\sum_{i=1}^{z} e^{-\left[\left(\frac{\Delta x_i}{4R}\right)^2+\left(\frac{\Delta y_i}{4R}\right)^2\right]} \tag{4b}$$

where:

$$\Delta x_i = x - x_i$$
$$\Delta y_i = y - y_i$$

and:
$F(x,y)$: potential field value for the cartesian Animal’s coordinates x, y; $[x_{\mathrm{T}}, y_{\mathrm{T}}]$: coordinates of Animal’s target point; $[x_i, y_i]$: coordinates of Animal’s i closest obstacle’s (other Animal’s) center; δ: parameter that defines the height of the exponential “mountain-like” function presented at (4b).

It is obvious that the surface plotted by each Animal alters at every time step as a result of *FlatLand*’s dynamics (moving obstacles - other Animals). The Animals’ motion therefore consists of a fixed non-linear strategy that does not evolve and is determined by the two-dimensional discontinuously time-varying potential field represented by (4). While, in theory, the APF solution may be prone to getting stuck through local minima, in practice, in the dynamic *FlatLand* world the probability of such cases to occur is significantly low and therefore, it can be ignored.

Any motion strategy that guides an agent to quickly achieve its target, avoiding any possible collisions and keeping the straightest and fastest possible trajectory to its target, is definitely a “good” strategy in terms of *FlatLand* world. Therefore, Animals present a “good” (near optimum) behavior in our simulated world and furthermore a reference case to compare it to any Humans’ behavior. This is the major reason for the use of this species of agents, along with the fact that data from the Animals’ motion strategy can be used to train the Humans’ neural network controller (see Subsection IV-A).

III. Hardness of the problem

In this section we provide evidence of the problem’s complexity and learning difficulty as well as its importance in the multi-agent systems research area. In fact, *FlatLand* is a hard environment for an agent to learn to live in because of its following distinct features:

- **Fully dynamical multi-agent**. Humans are moving continuously. Each Human faces a number of moving obstacles (i.e. potentially 19 other Humans) in a specific squared environment whereby it has no *a priori* knowledge about their motion. Prediction of motion can be emerged by complicating the sensor system of each Human (i.e. addition of speed vectors of moving obstacles).
- **Partial information**. One of the major difficulties of the problem is that Humans communicate just by “seeing” (i.e. spatial polar coordination) each other (see Fig. 2). This kind of communication regarding these kind of tasks (i.e. obstacle-avoidance and target-achievement) is very common in the animals world (e.g. predator-prey behaviors) as well as in human beings (e.g. crowded streets).
- **Discontinuous time-varying information** The Human’s input information suffers from discontinuity because of frequent alterations of the z closest Humans that takes into account via its sensors. Hence, the values of the polar coordinates $\mathrm{a_i}, \mathrm{d_i}$ $(\mathrm{i} = 1, \ldots, z)$ alter in a discontinuous fashion.
- **Supervised training**. If we try to train Humans under the near-optimum APF (Animals) strategy, we have to face problems such as missing information (data) for many cases. These are cases that Animals would never get into (e.g. case of $\mathrm{d_i} \leq 2R$) but trained Humans do.
- **Very few collision examples**. One of the difficulties of the *FlatLand* world is the small value of collisions per time step. Even in the worst obstacle-avoidance behaviors we experienced, 20 Humans collide approximately about 3000 times in 10^4 simulation steps. Hence, each Human collides less than 1.5% of its lifetime on average. Therefore, it is both hard and computationally expensive to emerge an obstacle-avoidance strategy by rewarding good examples of this strategy.

FlatLand’s basic concept and features make the proposed test-bed quite interesting for the multi-agent artificial life research area.

- **Emerged cooperation**. *FlatLand* is a simulated world that we expect to emerge cooperative behaviors without any information exchange apart from spatial coordination (see above). Therefore, cooperation is emerged from 1) the way Humans move and 2) the way they interact with their environment (see Section V).

- **Strong creature-environment interaction**. There is a strong interaction and relation between the simulated creatures-agents and their environment. In other words, any living creature in *FlatLand* faces an environment of a two-dimensional space that includes a number of other creatures. Creatures in FlatLand are part of their own environment. Furthermore, *FlatLand*'s main feature, as an environment, is its own living creatures. This feature defines an important point in the Artificial Life research of two-dimensional multi-agent simulated worlds.

IV. Learning Mechanisms

As mentioned in Section I, the main purpose of the *FlatLand* test-bed environment is to be used for investigation of evolutionary and gradient-based learning techniques and furthermore, their ability to emerge complex and adaptive obstacle-avoidance and target-achievement behaviors. In this section we present the learning mechanisms used. The main approach (presented in Subsection IV-B) is evaluating Humans from their own actions in the *FlatLand* environment (generational genetic algorithm) whereas the alternative back-propagation approach (presented in Subsection IV-A) consists of supervised learning attempting to train Humans on Animals' behavior.

A. Back-Propagation

The data set used for the supervised back-propagation (BP) training of the neural controllers consists of inputs and actions of an Animal. Hence, the use of this alternative supervised learning approach is based on Humans' evaluation by promoting any behavior that mimics the Animals' strategy (data set). This near-optimum Animal path is taken from a simulation of 80 Animals in the *FlatLand* environment. Crowded *FlatLand* environments (e.g. 80 agents) produce more obstacle-avoidance cases, than normal population (e.g. 20 agents) environments. Therefore, the data set taken from an Animal path in this environment contains more obstacle-avoidance examples in it. Furthermore, because of the fact that obstacle-avoidance is harder than target-achievement behavior for an agent to emerge, such data sets define good samples for training. The size of the training data set is 666 (i.e. this being 2/3 of a data set that was originally partitioned into training and testing portions) for the experiments presented here.

We are using the Levenberg-Marquardt algorithm [8] to train neural controllers. This algorithm appears to be the fastest method for training moderate-sized feedforward neural networks.

The algorithm is terminated when either it converges to a good mean square error (mse) value (e.g. for the experiments presented here, good training performance of the network is achieved when $mse = 0.0018$) or a predefined large number of epochs (e.g. 1000 epochs) is achieved.

B. Genetic Algorithm

As previously stressed, our aim is to emerge complex behaviors by evaluating Humans from their own actions in *FlatLand*. One of the difficulties we encounter in this approach is that there is no *a priori* fitness function for evaluating a genome. Hence, a simple generational genetic algorithm (GA) was implemented, which uses an "endogenous" evaluation function that emerges from the Humans' actions in the environment and promotes good collision-avoidance and target-achievement behaviors. Humans that learn to behave in this fashion, are fit enough to be considered as good solutions of the problem.

The neural networks that determine the behavior of the Humans are themselves evolved. In the work presented here, the evolving process is limited to the connection weights of the neural network. Evolving architectures and transfer functions are some of the ideas for future work.

The evolutionary procedure used can be described as follows. Each Human has a genome that encodes the connection weights of its neural network. A population of 20 (we keep this number low because of the increasing computational cost) neural networks (Humans) is initialized randomly while initial real values that lie within [-5, 5] for their connection weights are picked randomly from a uniform distribution. Then, at each time step:

1) Every Human in the population is cloned 20 times. These 20 clones are placed in the *FlatLand* environment and tested for an evaluation period (e.g. 200 simulation steps). The outcome of this test is to ascertain the total number of collisions C and target achievements T.
2) Each Human is evaluated via the following function:
$$f_i = \frac{max\left\{1 - \frac{C_i}{C_u}, 0\right\} + min\left\{\frac{T_i}{T_u}, 1\right\}}{2} \qquad (5)$$
where:
f_i: evaluation function of the i Human; C_i: total number of collisions of the i Human's 20 clones; C_u: total number of collisions' upper bound; T_i: total number of target achievements of the i Human's 20 clones; T_u: total number of target achievements' upper bound.
3) A pure elitism selection method is used where only the 2 best fit solutions determine the members of the intermediate population and therefore, are able to breed.
4) Each of the two parents clones 9 offspring.
5) Mutation occurs in each gene-connection weight of each offspring's genome with a small probability (e.g. 0.01). A uniform random distribution is used again to define the mutated value of the connection weight.

The algorithm is terminated when either a good fit Human ($f_i \geq 0.99$) is found or a large number of time steps (e.g. 2000) is achieved.

As mentioned before, a suitable evaluation function for the generational GA approach promotes good obstacle-avoidance and target-achievement behaviors in an "endogenous" way. Furthermore, by using (5), we promote Humans (their 20 clones) that do not crash and achieve a determined number of targets (T_u) during an evaluation period (e.g. 200 simulation steps). By this evaluation, we mainly promote twenty clones of the same Human (solution) capable of cooperating in order to successfully achieve the aforementioned desired behavior. Due to this, we are able to emerge very interesting cooperative behaviors within the same solution (see Section V).

V. Results

In this section we present and compare results obtained from the two learning mechanisms (BP, GA) applied in *FlatLand* as presented in Section IV.

A. Performance Measurement

We introduce an efficient method for testing and comparing different learning mechanisms' ability to emerge successful controllers. For each learning mechanism used, we pick up the best (in terms of the optimization function used) neural controller (Human). Then, we record the total number of both collisions C and target achievements T of a population of 20 copies of this agent in a specific number of simulation steps (e.g. 10^4) by placing these agents in *FlatLand* and running the simulation.

Since the initialization phase picks random numbers for initial positions and target points of the agents, it constitutes an important factor for any result. Therefore, 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 average numbers of total collisions $E\{C\}$ and target achievements $E\{T\}$. We used 10^4 simulation steps for measuring and evaluating any behavior (collisions, target achievements) since we believe it is a long enough period for evaluating a behavior of a population of agents in an efficient way. Results of this kind can be obtained from Table I.

In order to classify the efficiency of each learning mechanism used, we need a performance measurement. This measurement can be obtained from:

$$P = \frac{max\left\{1 - \frac{E\{C\}}{C_{TA}}, 0\right\} + min\left\{\frac{E\{T\}}{T_A}, 1\right\}}{2} \tag{6}$$

where:
P: performance function; C_{TA}: total number of collisions of 20 "Target Achievers" (i.e. agents that move directly towards their target points with constant speed - $a_{NN} = 0^{\circ}, d_{NN} = 0.5$ cm/time step) in 10^4 simulation steps ($C_{TA} = 2000$, see Table I); T_A: total number of target achievements of 20 Animals in 10^4 simulation steps ($T_A = 3200$, see Table I).

The maximum value of (6) is 1.0 and it is obtained only when the agents do not collide at all ($E\{C\} = 0$) 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 the Target Achievers (TAs) produce (C_{TA}) because they just move directly towards their target points and therefore, present the worst collision-avoidance behavior from our viewpoint. Hence, (6) produces a clear picture of how far the performance of each learning mechanism is from the optimal performance of Animals ($P = 1.0$).

B. Performance Comparison

Table I illustrates results from both learning mechanisms used (BP, GA). The neural controller employed is a 5-hidden neuron feedforward neural network. This controller emerges the best behavior (in terms of performance), among many feedforward neural controllers, for both learning mechanisms applied. Additionally, this is the only feedforward neural architecture presented here due to space considerations. Results presented in Table I represent the best (in terms of performance) obtained results for each learning mechanism.

TABLE I
Best Performance Comparison Table - Average Values are Obtained from 10 Evaluation Runs (10^4 Simulation Steps Each) of a 20-Agent Environment

Agents	$E\{C\}$	$E\{T\}$	$E\{V\}$	P
Random	198620	7	0.46	**0.0010**
TAs	2000	3348	0.50	**0.5**
BP	663	3121	0.51	**0.8219**
GA	268	3179	0.9	**0.9297**
Animals	0	3200	0.5	**1.0**

In Table I we introduce the best obtained performance of a species of agents called "Random" ($P = 0.0010$). These agents are randomly initialized Humans and the variance of their performance over the 10 evaluation runs s^2 equals to $12.03 \cdot 10^{-7}$. The Random agents along with the Target Achievers and the Animals are presented in Table I for comparison to any emergent Humans' behavior.

It is obvious that the GA approach ($P = 0.9297$, $s^2 = 12.5 \cdot 10^{-5}$) gets closer to the desired behavior (i.e. Animals) than BP ($P = 0.8219$, $s^2 = 31.8 \cdot 10^{-5}$) or any other "species" of agents. This high-performance behavior is achieved because the GA approach - in contrast with the BP approach that attempts to mimic the Animal's behavior - emerges Humans that manage to keep a big distance with each other in order to avoid collisions. Furthermore, they move with an almost maximum speed (i.e. $V = 0.9$) to achieve as many target points as possible. These results lead to the important conclusion that simple evolutionary learning mechanisms can produce much better behaviors than the ones produced from exhausting supervised learning approaches in *FlatLand*.

C. Robustness Comparison

As previously stressed, the 5-hidden neuron feedforward neural controller produces the best behavior (in terms of the performance function P) for both learning mechanisms employed. Hence, due to space considerations, results presented in this subsection are obtained from experiments of only this feedforward neural architecture's application.

In order to test the robustness of the solutions given and to calculate the effort cost of each approach, we are applying the following procedure. For each approach a) we repeat the learning attempt (run) ten times (each time, a different random initialization of the connection weights' values is given); b) the method presented in Subsection V-A is used to measure the performance of each run; c) the number of runs that present higher performance than a specific performance threshold value (i.e. $P > P_{threshold}$) determine the successes of the approach for this performance threshold. The higher the performance threshold value, the more demanding the procedure and robust the solutions. Fig. 5 illustrates the number of successes of both learning mechanisms (BP, GA) for ten different values of $P_{threshold}$.

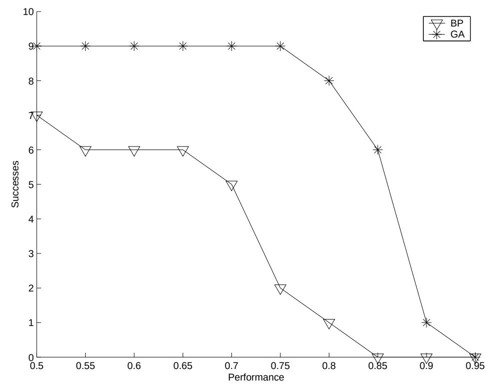

Fig. 5. Number of successes out of 10 runs for specific performance values

The generational GA is more efficient and robust approach than BP for every performance function threshold (see Fig. 5). It even emerges controllers (1 success) with $P \geq 0.9$ whereas BP approach's best performance is below 0.85. It is worth mentioning that for $P_{threshold} = 0.75$, GA succeeds in 9 out of 10 times while BP succeeds only 2 times. Finally, for $0.85 \leq P_{threshold} < 0.95$ there cannot be any effort cost comparison between the learning mechanisms because only the GA approach is capable of emerging behaviors of that high performance.

D. Mean Effort Cost Comparison

Since the GA approach is proven to be more robust learning mechanism than the BP approach, the next step is to compare these mechanisms via their mean effort cost. Hence, we pick decent high values of $P_{threshold}$ (i.e. $P_{threshold} \geq 0.75$) and proceed with a beta-distribution approximation of the mean effort cost [9] for both approaches. Due to space considerations, we do not present the detailed description of this standard statistical method here.

TABLE II
MEAN EFFORT COST COMPARISON TABLE

Approach	CPU time[1]	$P_{threshold}$	α	β	Mean Effort Cost[1]
BP	93.58	0.7	5	5	205.87
		0.75	2	8	514.69
		0.8	1	9	1029.38
GA	374.45	0.7	9	1	457.65
		0.75	9	1	457.65
		0.8	8	2	514.86

[1] in seconds

Results from the effort cost comparison via the beta-distribution statistical method for three different values of $P_{threshold}$ are presented in Table II. Thus, for each $P_{threshold}$ value the number of successes (α) and failures (β) of each approach is presented (as illustrated in Fig. 5). The unit computing cost per run Q is the average CPU time of the ten runs (every experiment presented here run under the same 1GHz processor). Finally, the mean effort cost is calculated with $\frac{\alpha+\beta+1}{\alpha}Q$.

The important conclusion that arises from Table II is that the BP approach is computationally preferred for low performance values (i.e. $P_{threshold} < 0.75$) from the GA approach. On the other hand, the GA learning mechanism's mean effort cost is much lower than the respective effort cost of the BP approach for demanding high-performance solutions (i.e. $P_{threshold} \geq 0.75$).

Finally, it is worth mentioning that the BP approach results presented in this paper are the best (in terms of performance) obtained from a variety of different training data sets as well as training improvement techniques (e.g. data preprocessing).

VI. CONCLUSION

We introduced both a hard and important problem for the multi-agent dynamic simulated world research area. We saw that a simple GA approach can emerge high-performance and robust behaviors as far as the hardness of the *Flat-Land* world is concerned. Additionally, in comparison with the supervised learning BP approach, the GA approach is both computationally preferred and more efficient for high-performance behaviors. Supervised learning BP approaches fail to compete evolutionary learning techniques because of their strong training data set dependence and the complex dynamics of the problem.

REFERENCES

[1] E. A. Abbott, *Flatland*. Signet Classic, June 1984.
[2] G. N. Yannakakis, "Evolutionary computation: Research on emerging strategies through a complex multi-agent environment," Master's thesis, Technical University of Crete, Chania, Greece, September 2001.
[3] A. Blair and E. Sklar, "Exploring evolutionary learning in a simulated hockey environment," in *Congress on Evolutionary Computation*. IEEE Service Center, 1999, pp. 197–203.
[4] D. H. Ackley and M. L. Littman, "Interactions between learning and evolution," in *Artificial Life II*, C. G. Langton, C. Taylor, J. D. Farmer, and S. Rasmussen, Eds., Sante Fe Institute Studies in the Sciences and Complexity. Reading, MA: Addison-Wesley, 1992, pp. 478–507.
[5] L. Yaeger, "Computational genetics, physiology, metabolism, neural systems, learning, vision, and behaviour of polyworld: Life in a new context," in *Artificial Life III: Proceedings of the Workshop on Artificial Life*, C. G. Langton, Ed., vol. XVII, Sante Fe Institute Studies in the Sciences and Complexity. Reading, MA: Addison-Wesley, 1993, pp. 263–298.
[6] C. W. Reynolds, "Evolution of corridor following behavior in a noisy world," in *From Animals to Animats 3: SAB-94*, D. Cliff, P. Husbands, J.-A. Meyer, and S. W. Wilson, Eds.
[7] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *Int. J. for Robotics Research*, vol. 5, no. 1, pp. 90–99, 1986.
[8] M. T. Hagan and M. Menhaj, "Training feedforward networks with the marquardt algorithm," *IEEE Transactions on Neural Networks*, vol. 5, no. 6, pp. 989–993, 1994.
[9] D. Kim, "A quantitative approach to the analysis of memory requirements for autonomous agent behaviours using evolutionary computation," Ph.D. dissertation, University of Edinburgh, 2002.