

DAEDALUS for Agents with Obstructed Perception

Suranga Hettiarachchi and William M. Spears
 Computer Science Department
 University of Wyoming
 Laramie, WY82071
 Email: {suranga, wspears@cs.uwyo.edu}

Abstract—Traditional approaches to designing multi-agent systems are offline (in simulation), and assume the presence of a global observer. In the online (real world), there may be no global observer, performance feedback may be delayed or perturbed by noise, agents may only interact with their local neighbors, and only a subset of agents may experience any form of performance feedback. Under these circumstances, it is much more difficult to design multi-agent systems. DAEDALUS is designed to address these issues, by mimicking more closely the actual dynamics of populations of agents moving and interacting in a task environment. This paper addresses the feasibility of DAEDALUS for agents moving towards a goal through an obstacle field, where the obstacles can obstruct perception.

I. INTRODUCTION

Engineering multi-agent systems is difficult due to numerous constraints, such as noise, limited range of interaction with other agents, delayed feedback, and the distributed autonomy of the agents. One potential solution is to automate the design of multi-agent systems in simulation, using evolutionary algorithms (EAs) [1], [2]. In this paradigm, the EA evolves the behaviors of the agents (and their local interactions), such that the global task behavior emerges. A global observer monitors the collective and provides a measure of performance to the individual agents. Agent behaviors that lead to desirable global behavior are hence rewarded, and the collective system is gradually evolved to provide optimal global performance.

There are several difficulties with this approach. First, a global observer may not exist. Second, some (but not all) agents may experience some form of reward for achieving task behavior, while others do not. Third, this reward may be delayed, or may be noisy. Fourth, the above paradigm works well in simulation (offline) but is not feasible for real-world online applications where unexpected events occur. Finally, the above paradigm may have difficulty evolving different individual behaviors for different agents (heterogeneity vs homogeneity).

In our prior work [3], we introduced “Distributed Agent Evolution with Dynamic Adaptation to Local Unexpected Scenarios” (DAEDALUS), for engineering multi-agent systems that can be used either offline or online. We will show how DAEDALUS can be used to achieve global aggregate behavior of agents that move through an obstacle field towards a goal. The obstacles obstruct the perception of the agents (i.e. they act to degrade the interactions between agents).

A. DAEDALUS

With the DAEDALUS paradigm, we assume that agents (whether software or hardware) move throughout some environment. As they move, they interact with other agents. These agents may be of the same species or of some other species [4]. Agents of different species have different roles in the environment. The goal is to evolve agent behaviors and interactions between agents, in a distributed fashion, such that the desired global behavior occurs. Let us further assume that each agent has some procedure to control its own actions in response to environmental conditions and interactions with other agents. The precise implementation of these procedures is not relevant, thus they may be programs, rule sets, finite state machines, real-valued vectors, force laws, or any other procedural representation. Agents have a sense of self-worth or “fitness”.

Each robot of the swarm is an individual in a population that interacts with its neighbors. Each robot contains a *slightly mutated* copy of the optimized control procedure found with offline learning with an offline EA. This ensures that our robots are not completely homogeneous. We allowed this slight heterogeneity because when the environment changes, some mutations perform better than others. The robots that perform well in the environment will have higher fitness than the robots that perform poorly. When low fitness robots encounter high fitness robots, the low fitness robots ask for the high fitness robot’s rules. Hence, better performing robots share their knowledge with their poorer performing neighbors. To ensure the capability of adapting to further changes in the environment, robots also occasionally mutate their own rules, according to a pre-defined mutation rate attached to that robot. In our original version of DAEDALUS, the robots do not exchange mutation rates when they exchange the rules.

B. Obstructed Perception

When a robot can not see another robot, due to the presence of obstacles, we call this “obstructed perception.” When the robot’s line of sight lies along an edge of an obstacle, the robots are capable of sensing each other. Surprisingly, this is not generally modeled in prior work in this area [12]. Figure 1 shows an example scenario of obstructed perception. The larger circle represents an obstacle, and A and B are robots. We define $minD$ to be the minimum distance from the center of the obstacle to the line of sight of robot A and

robot B , and r is the radius of an obstacle. If $r > \min D$, the robot A and robot B have their perception obstructed.

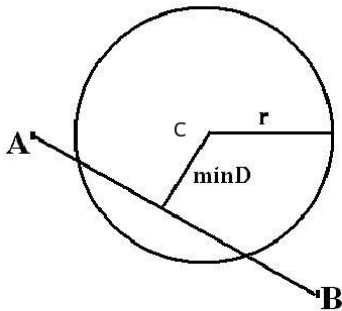


Fig. 1. Sensing capability of two robots (A, B) is obstructed by a large obstacle (C).

We utilize a parameterized description of a line segment [10] to find the $\min D$.

$$\begin{aligned} \text{term}_1 &= (((1 - q) * X_a + q * X_b) - X_c)^2 \\ \text{term}_2 &= (((1 - q) * Y_a + q * Y_b) - Y_c)^2 \\ \min D &= \sqrt{[\text{term}_1 + \text{term}_2]} \end{aligned} \quad (1)$$

where X_a, X_b are the x positions of robots A and B , Y_a, Y_b are the y positions of robots A and B , X_c and Y_c are the x and y positions of the center of an obstacle, and q is the minimum function that is defined by

$$\frac{((X_c - X_a) * (X_b - X_a) + (Y_c - Y_a) * (Y_b - Y_a))}{((X_b - X_a)^2 + (Y_b - Y_a)^2)} \quad (2)$$

C. Obstacle Avoidance

In prior work [5] we have shown how our artificial physics framework can be used to self-organize swarms of mobile robots into hexagonal lattices (networks) that move towards a goal (see Figure 2). We extended the framework to include motion towards a goal through an obstacle field. An offline EA evolved an agent-level force law, such that robots maintained network cohesion, avoided the obstacles, and reached the goal. The emergent behavior was that the collective moved as a viscous fluid [6]. In our prior work obstacles did not obstruct perception. The addition of obstructed perception makes the task far more difficult, especially as obstacle size increases.

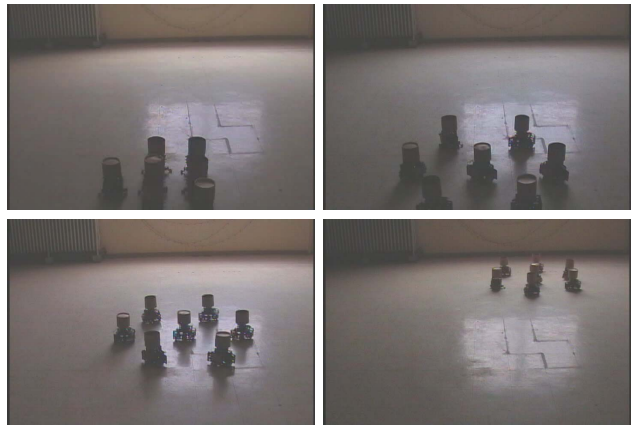


Fig. 2. Seven robots form a hexagon, and move towards a light source.

II. THE ARTIFICIAL PHYSICS FRAMEWORK

In our artificial physics (AP) framework, virtual physics forces drive a swarm robotics system to a desired configuration or state. The desired configuration is one that minimizes overall system potential energy, and the system acts as a molecular dynamics ($\vec{F} = m\vec{a}$) simulation.

Each robot has position \vec{p} and velocity \vec{v} . We use a discrete-time approximation of the continuous behavior of the robots, with time step Δt . At each time step, the position of each robot undergoes a perturbation $\Delta\vec{p}$. The perturbation depends on the current velocity, i.e., $\Delta\vec{p} = \vec{v}\Delta t$. The velocity of each robot at each time step also changes by $\Delta\vec{v}$. The change in velocity is controlled by the force on the robot, i.e., $\Delta\vec{v} = \vec{F}\Delta t/m$, where m is the mass of that robot and \vec{F} is the force on that robot. F and v denote the magnitude of vectors \vec{F} and \vec{v} . A frictional force is included, for self-stabilization.

From the start, we wished to have our framework map easily to physical hardware, and our model reflects this design philosophy. Having a mass m associated with each robot allows our simulated robots to have momentum. Robots need not have the same mass. The frictional force allows us to model actual friction, whether it is unavoidable or deliberate, in the real robotic system. With full friction, the robots come to a complete stop between sensor readings and with no friction the robots continue to move as they sense. The time step Δt reflects the amount of time the robots need to perform their sensor readings. If Δt is small, the robots get readings often, whereas if the time step is large, readings are obtained infrequently. We have included a parameter F_{max} , which provides a necessary restriction on the acceleration a robot can achieve. Also a parameter V_{max} restricts the maximum velocity of the robots (and can always be scaled appropriately with Δt to ensure smooth path trajectories).

In this paper we utilize a generalized Lennard-Jones (LJ) force law (which models forces between molecules and atoms) as the control procedure of our robots.

$$F = 24\epsilon \left[\frac{2dR^{12}}{r^{13}} - \frac{cR^5}{r^7} \right] \quad (3)$$

$F \leq F_{max}$ is the magnitude of the force between two robots

i and j , and r is the distance between the two robots. R is the desired separation between robot i and all other neighboring robots. The variable ϵ affects the strength of the force, while c and d control the relative balance between the attractive and repulsive components. In order to achieve optimal behavior, the values of ϵ , c , d , and F_{max} must be determined. Our motivation for using the LJ force law is that (depending on the parameter settings) it can easily model crystalline solid formations, liquids, and even gases.

III. EXPERIMENTAL METHODOLOGY

To deal with obstacle avoidance, we have separate force laws for robot-robot interactions, robot-goal interactions, and robot-obstacle interactions. Hence ϵ , c , d , and F_{max} must be optimized for all three forms of interactions, resulting in 12 parameters. Robot-robot and robot-obstacle interactions are local (i.e., robots can only sense nearby robots and obstacles). The robots are trained with an offline EA, in an offline environment. The environment is 900×700 with 90 randomly positioned obstacles and each of radius 10. This yields about 5% obstacle coverage, which is typical of most studies in this area [12]. The robots move with maximum velocity 20 units/sec. The EA does not have great difficulty producing an optimized LJ force law that avoids obstacles while allowing all robots to reach the goal.

However, the online environment is far more difficult. The online 2D world is 1600×950 , and each of the 90 obstacles has a radius of 30 compared to the offline obstacle radius of 10. Therefore more than 16% of the online environment is covered with the obstacles, tripling the obstacle density. We also increase the maximum velocity of the robots to 30 units/sec from 20 units/sec, making the robots move 1.5 times faster than in the offline environment. Obstructed perception occurs in both the offline and online environments.

For the online environment, each robot of the swarm contains a slightly mutated copy of the optimized LJ force law rule set found with offline learning. There are five goals to achieve in a long corridor, and between each randomly positioned goal is a different obstacle course with 90 randomly positioned obstacles. The LJ force law learned in offline mode is not sufficient for this more difficult environment, producing robots that never reach the goal (due to the high percentage of obstacles).

Robots that are left behind (due to obstacle cul-de-sacs) do not proceed to the next goal, but robots that collide with obstacles and make it to the goal are allowed to proceed to the next goal. We assume that damaged robots can be repaired once they reach a goal. Although the noise in dynamic environments is not specifically modeled in our simulation, it has been shown with actual robots that the Artificial Physics framework is robust to modest amounts of noise [5]. In fact, noise can actually improve performance by overcoming local optima in the behavior space [8], [9].

In our prior work [3] [6], we have shown that the robots easily learned to avoid colliding with obstacles, so our focus in this paper is on the survivability of the robots (i.e. the number of robots that reach a goal). When the robots are left behind

in cul-de-sacs, the number of robots that survive to reach a goal reduces, and this causes the cohesion of the formation to be reduced. We utilized two different methods to improve the survivability:

- if robot $_i$ is not moving (due to an obstacle in the way) and a neighboring robot $_j$ is moving, then robot $_i$ receives robot $_j$'s robot-robot interactions.
- if robot $_i$ is not moving (due to an obstacle) and robot $_i$ has no neighbors, then robot $_i$ mutates it's own robot-goal interactions. This mimics "panic behavior" seen in animals.

We focus on the first method in this paper.

IV. EXPERIMENTAL RESULTS

We compared DAEDALUS to three control studies. In the first control study, we train the robots with an offline EA on small obstacles, and then test them again on small obstacles to verify their performance. In the second control study, we train the robots with an offline EA on large obstacles and test them on large obstacles. The purpose of this control study is to clarify the difficulty of the task. Finally, in the third control study, we train the robots with an offline EA on small obstacles and test them on large obstacles. The purpose of this study was to see how well the knowledge learned while avoiding small obstacles transferred to large obstacles.

Figure 3 shows the results. The y-axis gives the number of robots that survived to reach the goal at each stage for the four different experiments. The top performance curve is for the first control study. Note that learning with small obstacles in offline mode is not hard, and the robots perform very well in the online environment. This is due to the fact that the small obstacles make the environment less dense providing the robots sufficient space to navigate. Out of 60 initial robots released in the online environment, 93.3% survived to reach the last goal. With such small obstacles (which is the maximum density examined in the related literature), obstructed perception is not an important issue.

In our prior work [3], robots that learned without obstructed perception on larger obstacles had a reasonably high survival rate (78%). The bottom (dashed) performance curve shows the effect of obstructed perception (the second control study). Learning with large obstacles in offline mode with obstructed perception is very difficult, and the test results show that out of 60 robots released initially into the online environment only 35% (21 robots) survived to reach the last goal. This is due to the fact that the environments with larger obstacles create large numbers of cul-de-sacs that obstruct perception.

The third control study, where offline training occurs with small obstacles and testing occurs with large obstacles, is surprisingly good (see "NO DAEDALUS (small-large)"). Despite an initial drop in performance, performance at the fifth goal is quite acceptable (out of the initial 60 robots, 40% (24 robots) survived to reach the final goal). This is a 5% improvement over the robots that were trained on larger obstacles. These results run counter to accepted wisdom, which states that it is best to train on the hardest environments that you will encounter. In fact, this example demonstrates that training

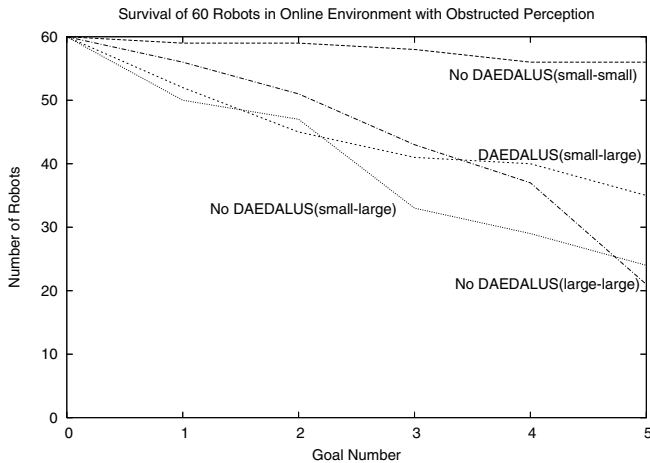


Fig. 3. Four Different Experiments of Number of Robots Surviving - All Robots are Trained with Obstructed Perception and Tested with and without DAEDALUS. Results averaged over 100 independent runs.

on simpler problems and applying the knowledge gained to harder problems can potentially provide superior results. Why is this so? As with developmental psychology, one does not train children on hard problems immediately, instead, we train them on easier problems first, in the hopes that they will learn the “basics” (which are important building blocks for solving other, more difficult, problems) more quickly.

If we extend the developmental psychology analogy further, we note that we encourage children to experiment and modify their behavior, based on changes in the environment. Furthermore, they share the lessons learned. This is precisely what the DAEDALUS system does. The final performance curve in Figure 3 shows the results. With an initial 60 robots, 58.3% or 35 robots survived to reach the last goal. This is a 23.3% improvement over the robots that learned in an environment with the larger obstacles, and a 18.3% improvement over the robots that learned with small obstacles and tested with the larger obstacles without DAEDALUS. These preliminary results are very promising. Although encouraging the robots (or children) to explore and experiment does provide an early drop-off in performance (compared to the “NO DAEDALUS (large-large)” curve), the results after four goals are superior. This is a classic example of “exploration” vs “exploitation”. Pure exploitation of learned knowledge is good up to a point, but will eventually fail as the problems become more difficult. Exploration provides the key to adapt to these changing environments. DAEDALUS provides just this form of exploration.

A. HOMOGENEOUS DAEDALUS RESULTS

For the DAEDALUS performance curve given above, all robots had the same mutation rate, which was 5%. Hence, each robot had the same rate of exploration. Although the rules for each robot may differ, their mutation rates are identical, and we refer to this system as “Homogeneous DAEDALUS”. However, there are numerous problems with this approach. First, the results may depend quite heavily on choosing the correct mutation rate. How is this mutation rate to be chosen? Second, the best mutation rate may also depend on the

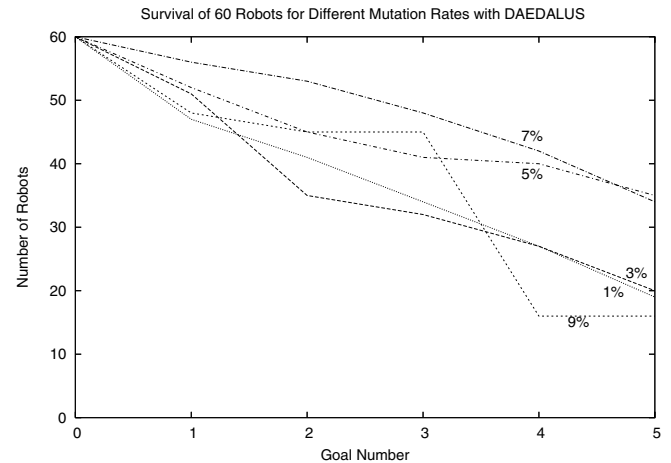


Fig. 4. Five Different Mutation Experiments of Robots Surviving. - All Robots are Trained with Obstructed Perception and Tested with DAEDALUS. Results averaged over 100 independent runs.

environment, and should potentially change as the environment changes. How is this to be accomplished?

Since the mutation rate may have a major effect on performance, we decided to explore this effect by conducting several experiments with different mutation rates. Figure 4 shows five independent experiments of Homogeneous DAEDALUS. Five different mutation rates were used: 1%, 3%, 5%, 7%, and 9%. The results are quite striking. Of the five different mutation rates, only 5% and 7% did well (with about 35 robots surviving to the last goal). Recall that the DAEDALUS performance curve shown in Figure 3 resulted from an arbitrarily chosen mutation rate of 5%. As it turns out, we were extremely fortunate in our design decision. For example, with mutation rates of 1%, 3%, and 9%, at most 20 robots survive to reach the final goal. The performance curve for the 9% mutation rate is especially interesting. Although promising at first, it appears as if the mutation rate is so high that it eventually causes an extremely deleterious mutation to appear. Mutation rates of 1% and 3% are too low to cope with the changed environment.

B. HETEROGENEOUS DAEDALUS RESULTS

In an attempt to address the problem of choosing the correct mutation rate, we divided the robots into five groups of equal size. Each group of 12 robots was assigned a mutation rate of 1%, 3%, 5%, 7%, and 9%, respectively. This mimics the behavior of children that have different “comfort zones” in their rate of exploration. Since different robots have different mutation rates, we refer to this system as “Heterogeneous DAEDALUS”. Figure 5 shows the results, in comparison with the three control studies shown in Figure 3. The label “Het.DAEDALUS (small-large)” shows the survivability of robots with pre-assigned mutation rates. Out of the initial 60 robots, 27 or 45% robots survived to reach the final goal. Although this is higher than our second and third control studies, it did not produce results as good as the results achieved with Homogeneous DAEDALUS using a 5% mutation rate (as shown in Figure 4). In fact, the result at the final goal

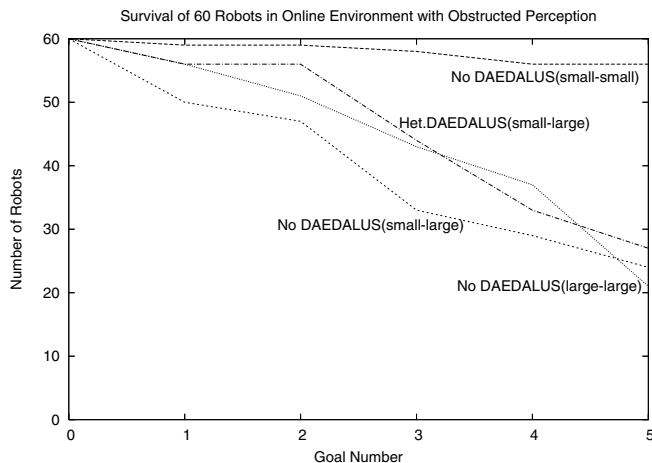


Fig. 5. Number of Robots Surviving with Predefined mutation rates. Mutation Rates are not Exchanged. - All Robots are Trained with Obstructed Perception and Tested with or without DAEDALUS. Results averaged over 100 independent runs.

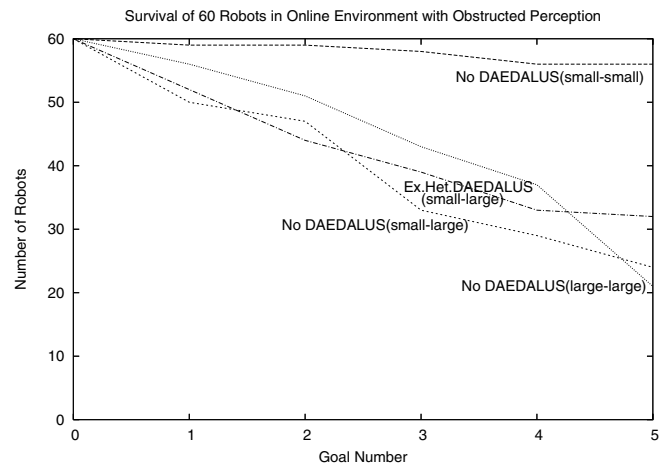


Fig. 6. Number of Robots Surviving with Predefined Mutation Rates. Mutation Rates are Exchanged. - All Robots are Trained with Obstructed Perception and Tested with or without DAEDALUS. Results averaged over 100 independent runs.

is essentially identical to the average of the five performance curves shown in Figure 4.

C. EXTENDED HETEROGENEOUS DAEDALUS RESULTS

In an attempt to improve performance, we again borrowed from the analogy of a “swarm” of children learning some task. Not only do they share useful information as to the rules they might use, but they also share meta-information as to the level of exploration that is actually safe! Very bold children might encourage their more timid comrades to explore more than they would initially. On the other hand, if a very bold child has an accident, the rest of the children will become more timid. In “Extended Heterogeneous DAEDALUS”, five groups of children are again initialized with mutation rates of 1%, 3%, 5%, 7%, and 9%. However, in this situation, if a robot receives the rules from a neighbor (which, again, occurs if that robot is in trouble), it also receives the neighbor’s mutation rate. In this implementation, children in trouble not only change their rules, but their mutation rate. Figure 6 shows the results of this study. The curve labeled with “Ex.Het.DAEDALUS (small-large)” refers to the survivability of robots with pre-assigned mutation rates that also allows the robots to receive a neighbor’s mutation rate, if the robot receives the neighbor’s rules. The behavior is quite good. On average, 32 robots survive to reach the final goal, which is very close to the optimum value of 35 found by the best Homogeneous DAEDALUS experiment.

V. SUMMARY

This paper addresses the important issue of “obstructed perception” in learning behaviors for swarms of robots that must avoid obstacles while reaching a goal. This issue has been largely absent from the literature. Our obstacle density is also three times higher than the norm, making obstacle avoidance a far more difficult task. First, we summarize our prior work in having swarms of robots respond to changing environments in real time (we refer to this system as “DAEDALUS”). Since obstructed perception makes the task far more difficult,

DAEDALUS had to be extended. Our first extension was to allow different robots to have different rates of exploration, which affects the rate at which they change their behavioral rules. The second extension allows robots to also share their rates of mutation, allowing robots to find the right balance between exploration and exploitation. Results of the extended system are almost as good as the best results we were able to achieve when the exploration rates were controlled by hand. In summary, this paper provides a framework that allows swarms of robots to not only learn and share behavioral rules in changing environments (in real time), but also to learn the proper amount of behavioral exploration that is appropriate.

VI. RELATED WORK

Most of the swarm robotics literature can be subdivided into *swarm intelligence*, *behavior-based*, *rule-based*, *control-theoretic* and *physics-based* techniques. Swarm intelligence techniques are ethologically motivated and have had excellent success with foraging, task allocation, and division of labor problems [15], [16]. Both behavior-based and rule-based systems [14], [13], [20] have proven quite successful in demonstrating a variety of behaviors in a heuristic manner. Behavior-based and rule-based techniques do not make use of potential fields or forces. Instead, they deal directly with velocity vectors and heuristics for changing those vectors (although the term “potential field” is often used in the behavior-based literature, it refers to a field that differs from the strict Newtonian physics definition). Control-theoretic approaches have also been applied effectively (e.g., [17]). Our approach does not make the assumption of having leaders and followers, as in [18], [19].

In the specific context of obstacle avoidance, the most relevant papers are [13], [12] and [14]. Balch [13] examines the situation of four robots moving in formation through an obstacle field with 2% coverage. In [12] he extends this to an obstacle field of 5% coverage, and also investigates the behavior of 32 robots moving around one medium size

obstacle. Fredslund and Matarić [14] examine a maximum of eight robots moving around two wall obstacles.

The work done in [11] uses an embedded network distributed throughout the environment to approximate the path-planning space and use the network to compute a navigational path using GNATs when the environment changes. The dynamism of the environment is modeled with an opening and closing door in the experimental setup. However, the embedded network is immobile, whereas our network is completely mobile.

VII. CONCLUSION AND FUTURE WORK

Future work of this research will focus on two aspects. First, we believe that we can accelerate the learning of the mutation rates. For example, currently, when a robot is in trouble, it receives the rules and mutation rate of a neighbor that is not in trouble. But this same neighbor could also query the robot in trouble to find out its mutation rate. Then the neighbor could spread this information further, to inform other robots that this particular mutation rate might be problematic. Second, we will address the issue of credit assignment, when fitness feedback is sporadic. Current work in classifier systems uses mechanisms such as “bucket-brigade” or “profit sharing” to allocate rewards to individual agents appropriately [7]. However, these techniques rely on global blackboards and assume that all robots can potentially act with all others through a bidding process. We intend to modify these approaches so that they are fully distributed and appropriate for online systems.

Also, as suggested by the reviewers of this paper, we will conduct studies to test the feasibility of DAEDALUS in structured environments (i.e. connecting rooms separated with walls).

REFERENCES

- [1] Grefenstette, J.: A system for learning control strategies with genetic algorithms. *Proceedings of the Third International Conference on Genetic Algorithms*. Morgan Kaufmann (1989) 183–190
- [2] Wu, A., Schultz, A., Agah, A.: Evolving control for distributed micro air vehicles. *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation*. IEEE Press (1999) 174–179
- [3] Hettiarachchi, S., Spears, W., Green, D., and Kerr, W.: Distributed Agent Evolution with Dynamic Adaptation to Local Unexpected Scenarios. *Proceedings of the 2005 Second GSFC/IEEE Workshop on Radical Agent Concepts*. (in Press).
- [4] Spears, W.: Simple Subpopulation Schemes. *Proceedings of the Evolutionary Programming Conference*. World Scientific (1994) 296–307
- [5] Spears, W., Spears, D., Hamann, J., Heil, R.: Distributed, Physics-Based Control of Swarm of Vehicles. *Autonomous Robots*. Kluwer **17** (2004) 137–164
- [6] Hettiarachchi, S., Spears, W.: Moving Swarm Formations Through Obstacle Fields. *International Conference on Artificial Intelligence*. CSREA Press **1** (2005) 97–103
- [7] Grefenstette, J.: Credit Assignment in Rule Discovery Systems Based on Genetic Algorithms. *Springer-Verlag* **3** (1988) 225–245
- [8] Martinson, E., Payton, D.: Lattice Formation in Mobile Autonomous Sensor Arrays. *Lecture Notes in Computer Science*, **3342**. Springer (2005) 98–111
- [9] Spears, W., Spears, D.: Using Artificial Physics to Control Agents. *IEEE International Conference on Information, Intelligence, and Systems*, (1999)
- [10] Haack, N.: Minimum Distance Between a Point And a Line, <http://www.simdesign.nl/tips/tip001.html>. (2002)
- [11] O’Hara, K. J., Bigio, V. L., Dodson, E. R., Irani, A., Walker, D. B., and Balch, T. R.: Physical Path Planning Using the GNATs. *IEEE International Conference on Robotics and Automation*, Barcelona, Spain, (2005)
- [12] Balch, T., Hybinette, M.: Social Potentials for Scalable Multi-Robot Formations. *IEEE International Conference on Robotics and Automation*. (2000)
- [13] Balch, T., and Arkin, R.: Behavior-based Formation Control for Multi-Robot Teams. *IEEE Transactions on Robotics and Automation* **14** (1998) 1–15
- [14] Fredslund, J., Matarić, M.: A General Algorithm for Robot Formations Using Local Sensing and Minimal Communication. *IEEE Transactions on Robotics and Automation* **18** (2002)
- [15] Bonabeau, E., Dorigo, M., Theraulaz, G.: *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press, Santa Fe Institute Studies in the Sciences of Complexity, (1999)
- [16] Hayes, A., Martinoli, A., Goodman, R.: Swarm Robotic Odor Localization. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, (2001)
- [17] Fax, J., Murray, R.: Information Flow and Cooperative Control of Vehicle Formations. *IFCA World Congress*, (2002)
- [18] Desai, J., Ostrowski, J., Kumar, V.: Controlling Formations of Multiple Mobile Robots. *IEEE International Conference on Robotics and Automation*, (1998)
- [19] Desai, J., Ostrowski, J., Kumar, V.: Modeling and Control of Formations of Nonholonomic Mobile Robots. *IEEE Transactions on Robotics and Automation*, **17**(2001) 905–908
- [20] Schultz, A., Parker, L.: *Multi-Robot Systems: From Swarms to Intelligent Automata*. Kluwer (2002)