International Conference on Control, Automation and Systems 2010
Oct. 27-30, 2010 in KINTEX, Gyeonggi-do, Korea

An Evolutionary Approach to Swarm Adaptation in Dense Environments

Suranga Hettiarachchi

Computer Science Department
Indiana University Southeast
New Albany, Indiana, USA
(E-mail: suhettia@ius.edu)

Abstract: The ability for a swarm of mobile agents to quickly adapt in unknown environments and reach a goal while avoiding obstacles and maintaining a formation is extremely important in time critical tasks. We utilize a physics-based autonomous agent framework combined with our DAEDALUS paradigm which allows the agents to learn from the neighboring agents. In traditional approaches, a swarm of agents learn the task in simulation(offline) combined with an evolutionary/genetic algorithm, and a global observer optimizes the swarm performance. In real world(online), the swarm of agents may have to rapidly adapt in unfamiliar environments. When there is no global observer and the online(real world) environment is dense with obstacles compared to offline environment, the performance feedback may be delayed or perturbed by noise, and the rules learned in simulation(offline) may not be sufficient to overcome the navigational difficulties, leaving the swarm to rapidly adapt in new environment. DAEDALUS is a paradigm 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 presents an analysis of swarm adaptation using DAEDALUS in high obstacle density environments where agent interactions could be obstructed by obstacles.

Keywords: autonomous agents, DAEDALUS, adaptive swarms, dense environments, physics-based, swarm survival.

1. INTRODUCTION

Swarm engineering 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) [8], [17]. In this paradigm, an EA evolves the behaviors of the agents (and their local interactions), such that the global task behavior emerges. A global observer monitors the swarm and provides a measure of performance to the individual agents. Agent behaviors that lead to desirable global behavior are hence rewarded, and the swarm system is gradually evolved to provide the optimal global performance.

There are several difficulties with the above evolutionary 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 [10], we introduced "Distributed Agent Evolution with Dynamic Adaptation to Local Unexpected Scenarios"(DAEDALUS), as a paradigm for engineering multi-agent systems that can be used either offline or online and showed how DAEDALUS can be used to achieve global aggregate behavior of agents that move through an obstacle field towards a goal, where obstacles obstruct the perception of the agents (i.e. they act to degrade the interactions between agents).

This paper explores the performance of DAEDALUS in high obstacle density environments when the gents are only capable of observing the environment partially due to sensor obstruction. We define partial observability or *obstructed perception* as when an agent's line of sight to another agent is obstructed by an obstacle. We use "swarm survival" (i.e. the amount of agents reaching a goal within a predefined time interval) as our performance metric. We present a detail analysis showing how DAEDALUS can be used to improve swarm survival in online, partially observable, high obstacle density environments.

2. DAEDALUS PARADIGM

The DAEDALUS paradigm assumes 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 [15]. 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 agent of the swarm is an individual in a population that interacts with its neighbors. Each agent contains a *slightly mutated* copy of the optimized control procedure found with offline learning with an offline EA. This ensures that our agents are not completely homogeneous. We allowed this slight heterogeneity because when the environment changes, some mutations perform better than others. The agents that perform well in the environment will have higher fitness than the agents that perform poorly. When low fitness agents encounter high fitness agents, the low fitness agents ask for the high fitness agent’s rules. Hence, better performing agents share their knowledge with their poorer performing neighbors. To ensure the capability of adapting to further changes in the environment, agents also occasionally mutate their own rules, according to a predefined mutation rate attached to that agent.

3. THE ARTIFICIAL PHYSICS FRAMEWORK

In our physics-based(AP) framework, virtual physics forces drive a swarm of agents 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 agent has position $\vec{p}$ and velocity $\vec{v}$. We use a discrete-time approximation of the continuous behavior of the agents, with time step Δt. At each time step, the position of each agent 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 agent at each time step also changes by $\Delta\vec{v}$. The change in velocity is controlled by the force on the agent, i.e., $\Delta\vec{v} = \vec{F}\Delta t/m$, where m is the mass of that agent and $\vec{F}$ is the force on that agent. F and v denote the magnitude of vectors $\vec{F}$ and $\vec{v}$. A frictional force is included, for self-stabilization.

The time step Δt reflects the amount of time the agents need to perform their sensor readings. If Δt is small, the agents 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 agent can achieve. Also a parameter V_{max} restricts the maximum velocity of the agents (and can always be scaled appropriately with Δt to ensure smooth path trajectories).

We utilize a generalized Lennard-Jones(LJ) force law as the control procedure of our agents. The LJ potential function models two distinct forces between neutral molecules and atoms. The forces are based on the distances between the molecules; at long ranges the attractive force makes the molecules move closer and at short ranges the repulsive force makes the molecules move apart, causing the molecules to maintain a natural balance. We derive the force function (i.e. negated derivative of the potential function) for interaction between two agents as:

$$F_{i,j} = 24\epsilon \left[\frac{2d\sigma^{12}}{r^{13}} - \frac{c\sigma^6}{r^7} \right] \tag{1}$$

$F_{i,j} \leq F_{max}$ is the magnitude of the force between two agents i and j, and r is the distance between the two agents. σ is the desired separation between agent i and agent j (i.e. all other neighboring agents). 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.

4. EXPERIMENTAL SETUP

In the offline environment, an EA is trained using 40 agents with 5% obstacle density and maximum velocity of 20 units/time steps in an 900 $\times$ 700 2D world. The circular obstacles and the goal are randomly positioned in the world.

We utilize separate force laws for agent-agent interactions, agent-goal interactions, and agent-obstacle interactions to achieve the best performance of the EA. Hence ϵ, c, d, and F_{max} of the LJ force law must be optimized offline for all three forms of interactions, resulting in 12 parameters. The EA generates offspring using one-point crossover with a crossover rate of 60%. Mutation adds/subtracts an amount drawn from a $N(0, \delta)$ Gaussian distribution. Each parameter has a $1/L$ probability of being mutated, where L is the length of the individual. Mutation ensures that parameter values stay within accepted ranges.

Since we are using an EA that minimizes, the offline performance of an individual is measured as a weighted sum of penalties:

$$w_1 P_{Collision} + w_2 P_{NoCohesion} + w_3 P_{NotReachGoal}$$

The weighted multi-objective fitness function consists of three criteria: collision avoidance, maintaining cohesion, and reaching the goal within a time limit (measurement of survival).

The agents in offline environment do not make use of DAEDALUS paradigm and the *obstructed perception* does not occur in EA learning. The EA optimized(offline) rule sets of LJ force law is slightly mutated again with the amounts drawn from a $N(0, \delta)$ Gaussian distribution and used in the online environment. All units are in pixels.

In the online environment, There are five goals to achieve in a long corridor where each segment (from one goal to next goal) of the corridor is an 900 $\times$ 700 2D world, and between each randomly positioned goal is an obstacle course where obstacles are once again positioned randomly. We vary the radius of circular obstacles to attain different obstacle densities in different control studies while keeping the number of obstacles constant. We also increase the maximum velocity of the agents to 30 units/time step from 20 units/time step used in offline world, making the agents move 1.5 times faster in online world.

The 40 agents initially start their navigation from the bottom left corner of the rectangular world and navigates towards a goal at an end of a corridor segment. The agents that do not reach a goal within a 2000 time steps do not proceed to the next goal. Some agents may get stuck behind cul-de-sacs formed by obstacles due to high density and may not reach a goal.

5. RESULTS AND ANALYSIS

The results consists of four separate experiments: the first experiment (Table 1) shows the number of agents survived to reach a goal without DAEDALUS and without *obstructed perception*, the second experiment (Table 2) shows the number of agents survived to reach a goal without DAEDALUS and with *obstructed perception*, the third experiment (Table 3) shows the number of agents survived to reach a goal with DAEDALUS and with *obstructed perception*, and the fourth experiment (Table 4) shows the number of agents survived to reach a goal with DAEDALUS and without *obstructed perception*. Each experiment consists of several control studies where each control study is conducted with a different obstacle density. All control studies show the number of agents survive to reach a goal with the obstacle density of the environment. All results are averaged over 50 independent runs.

Table 1 : The number of agents survived to reach a goal without DAEDALUS and without *obstructed perception*.

	Goal Number				
Density	Goal 1	Goal 2	Goal 3	Goal 4	Goal 5
5%	39	38	37	37	36
10%	38	37	34	31	30
15%	35	33	28	22	21
20%	32	27	20	15	13
25%	31	20	13	10	7
30%	22	10	6	3	0

The results in the Table 1 (experiment 1) shows the number of agents survive to reach a goal without DAEDALUS and without *obstructed perception* in the online environment. Out of 40 agents 36 agents or 90% of the agents survive to reach the last goal when the obstacle density is 5%. The agents do not have a significant difficulty performing in this environment since this is similar to the offline behavior learned using EA. Though not significant, the 10% (4 agents) decrease in survival can mainly be attributed to the difference in online(corridor with five segments) vs offline obstacle course. When the obstacle density is 15%, 52% or 21 of the agents reach the final goal. None of the agents survive to reach the final goal when the obstacle density is 30%, where the density is six times higher than the offline environment.

In our second experiment, we introduced *obstructed perception* to agent-agent interactions. When an agent can not sense another agent, due to the presence of obstacles, we call this the "obstructed perception." When the agent's line of sight lies along an edge of an obstacle, the agents are capable of sensing each other. The results are shown in the Table 2 .

Table 2 : The number of agents survived to reach a goal without DAEDALUS and with *obstructed perception*.

	Goal Number				
Density	Goal 1	Goal 2	Goal 3	Goal 4	Goal 5
5%	38	38	38	36	36
10%	37	36	35	31	29
15%	32	32	27	13	12
20%	27	26	22	5	5
25%	13	11	8	2	0

The results in the Table 2 shows the number of agents survive to reach a goal without DAEDALUS and with *obstructed perception* in the online environment. Out of 40 agents 36 agents or 90% of the agents survive to reach the last goal when the obstacle density is 5%. The number of agents reaching the goal in this control study is very similar to the first control study in the first experiment. When the obstacle density is 15%, only 30% or 12 of the agents reach the final goal, and this is a 22% (9 agents) decrease in agent survival compared to the same control study in the first experiment. These differences in agent survival show that the *obstructed perception* has a major effect on swarm adaptation in online environment, and requires DAEDALUS like paradigm to overcome the navigational difficulties. No agents survive to reach the last goal when the obstacle density is 25%.

We apply our DAEDALUS paradigm to further improve swarm survival. The results of the third experiment presented in the Table 3 shows the number of agents survived to reach a goal with DAEDALUS and with *obstructed perception*.

In the first control study, out of 40 agents 36 agents or 90% of the agents survive to reach the last goal when the obstacle density is 5%. Again, the agents do not have a significant difficulty performing in this environment. For

Table 3 : The number of agents survived to reach a goal with DAEDALUS and with *obstructed perception*.

	Goal Number				
Density	Goal 1	Goal 2	Goal 3	Goal 4	Goal 5
5%	39	38	37	36	36
10%	38	37	35	33	31
15%	37	36	34	28	25
20%	35	33	30	27	22
25%	32	26	24	16	13
30%	28	23	21	13	9
35%	22	14	11	5	0

obstacle densities 15% through 25% the swarm performance is significantly better compared to the results in the experiment 2 (see Table 2). When the obstacle density is 15%, more than 62.5% or 25 of the agents reach the final goal. In the same control study in the second experiment where DAEDALUS is not utilized, only 30% or 12 of the agents reach the final goal, and this is a 32.5% (13 agents) improvement. The control studies conducted with 20% and 25% also show significant improvements of agent survival over the same control studies in the experiment without DAEDALUS. No agents survive to reach the last goal when the obstacle density is 35%.

To better understand the impact of *obstructed perception* on agents, we conducted the fourth experiment. The results presented in the Table 4 shows the number of agents survived to reach a goal with DAEDALUS and without *obstructed perception*.

Table 4 : The number of agents survived to reach a goal with DAEDALUS and without *obstructed perception*.

	Goal Number				
Density	Goal 1	Goal 2	Goal 3	Goal 4	Goal 5
5%	38	37	37	36	36
10%	38	35	35	31	27
15%	37	34	34	29	26
20%	35	31	30	25	23
25%	34	29	27	24	21
30%	29	23	19	14	12
35%	26	21	18	10	8
40%	21	15	12	5	0

Out of 40 agents 36 agents or 90% of the agents survive to reach the last goal when the obstacle density is 5%. Once again, the number of agents reaching the last goal in this control study is very similar to the first control study in the first experiment. When the obstacle density is 15%, 65% or 26 of the agents reach the last goal, which is similar to 62% or 25 of the agents reaching the last goal of the same control study in the third experiment. At 35% obstacle density, 20% or 8 of the agents survive to reach the last goal. No agents survive to reach the last goal when the obstacle density is 40%. These results show that DAEDALUS effectively improve the agent survival in dense environments, even when adapting to new environments is much more difficult when the agent interactions are obstructed by obstacles.

6. SUMMARY AND CONCLUSION

The ability for a swarm of mobile agents to quickly adapt in unknown environments and reach a goal while avoiding obstacles and maintaining a formation is extremely important in time critical applications. Our approach to this is a physics-based autonomous agent framework combined with our DAEDALUS paradigm which allows the agents to learn from the neighboring agents by sharing the force law rules. We explore the feasibility of DAEDALUS in dense environments using "swarm survival" as our performance metric, when the agents are only capable of observing their environment partially due to *obstructed perception*. We present our results with an analysis which shows that the application of DAEDALUS paradigm significantly improves online swarm survival by allowing the agents to adapt in dense environments. The results also show that obstacle avoidance is a much more difficult task in dense environments, and DAEDALUS significantly enhances the swarm adaptation by lowering the rate of agents that get stuck behind cul-de-sacs.

7. RELATED WORK

We can subdivide most of the swarm literature based on the techniques such as *behavior-based*, *rule-based*, *swarm intelligence*, *control-theoretic* and *physics-based*. Both behavior-based and rule-based systems presented in [7], [1], [14] applied in a heuristic manner have proven quite successful in demonstrating a variety of behaviors. 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). Swarm intelligence techniques are ethologically motivated and have had excellent success with foraging, task allocation, and division of labor problems [3], [9]. Control-theoretic approaches have also been applied effectively (e.g., [6]). Our AP framework does not make the assumption of having leaders and followers, as in [4], [5].

Some of the relevant papers in obstacle avoidance context are [1], [2], [7]. Balch [1] examines the situation of four agents moving in formation through an obstacle field with 2% coverage. In [2] he extends this to an obstacle field of 5% coverage, and also investigates the behavior of 32 agents moving around one medium size obstacle. Fredslund and Matarić [7] examine a maximum of eight agents moving around two wall obstacles. We have utilized our AP framework combined with DAEDALUS paradigm to produce comparable but scalable results in obstacle avoidance task in more complex and dynamic environments [10], [11].

The work presented in [16], [12], [13] utilizes different multi-agent learning paradigms. Tan, [16] uses reinforcement learning to address agent learning by sharing instantaneous information, episodes, and learned policies. The task environment is a 10 by 10 grid world with maximum of 4 agents, 2 hunters and 2 prey. Our work is conceptually similar and was developed independently. The cooperative learning discussed in [16] is fundamentally offline, whereas in our approach learning is both offline and online where agents continue to adapt to their changing environment through cooperation. The agents depending only on offline learning approach can be problematic due to complex nature and the unexpected changes in the environment. The work in [12] utilizes fuzzy automata to address autonomous mobile robot learning reactive obstacle avoidance task using two robots. The robots share their experiences while learning the task simultaneously. Their results clearly show that sharing experience makes learning faster and more repeatable than the individual learning. This again attest the observations we presented in this paper. Pugh and Martinoli in [13] explore how varying sensor offsets and scaling factors affect parallel swarm robotic learning of obstacle avoidance behavior using both a genetic algorithm and particle swarm optimization. In our work, all agents have the same sensor and we assume that the sensor readings have no variations. The results show that both algorithms are able to withstand small variations of sensor offsets and large variations of sensor scaling factors while showing poor performance with high offset variations. We intend to utilize sensor offsets and scaling factors in our future work.

REFERENCES

[1] Balch, T.; and Arkin, R. Behavior-based Formation Control for Multi-Robot Teams. IEEE Transactions on Robotics and Automation, Vol 14: 1–15, 1998.

[2] Balch, T.; and Hybinette, M. Social Potentials for Scalable Multi-Robot Formations. IEEE Proceedings of the International Conference on Robotics and Automation, 73–80, 2000.

[3] Bonabeau, E.; Dorigo, M.; and Theraulaz, G. Swarm Intelligence: From Natural to Artificial Systems. Santa Fe Institute Studies in the Sciences of Complexity: Oxford University Press, 1999.

[4] Desai, J.; Ostrowski, J.; and Kumar, V. Controlling Formations of Multiple Mobile Robots. Proceedings of the IEEE International Conference on Robotics and Automation, 2864–2869, 1998.

[5] Desai, J.; Ostrowski, J.; and Kumar, V. Modeling and Control of Formations of Nonholonomic Mobile Robots. IEEE Transactions on Robotics and Automation 17: 905–908, 2001.

[6] Fax, J.; and Murray, R. Information Flow and Cooperative Control of Vehicle Formations. IFCA World Congress, 2002.

[7] Fredslund, J.; and Matarić, M. A General Algorithm for Robot Formations Using Local Sensing and Minimal Communication. IEEE Transactions on Robotics and Automation, Vol. 18(5): 837–846, 2002.

[8] Grefenstette, J. A System for Learning Control Strategies with Genetic Algorithms. *Proceedings of the Third International Conference on Genetic Algorithms, Morgan Kaufmann,* 183–190, 1989.

[9] Hayes, A.; Martinoli, A.; and Goodman, R. Swarm Robotic Odor Localization. IEEE/RSJ Proceedings of the International Conference on Intelligent Robots and Systems, 1073–1078, 2001.

[10] Hettiarachchi, S. and Spears, W. DAEDALUS for Agents with Obstructed Perception. *Proceedings of the 2006 IEEE Mountain Workshop on Adaptive and Learning Systems, IEEE Press,* 195–200, 2006.

[11] Hettiarachchi, S. Improving Swarm Survival Using DAEDALUS. *Proceedings of the 21st Midwest Artificial Intelligence and Cognitive Science Conference,* 36–43, 2010.

[12] Kelly, I. D. and Keating, D. A. Faster learning of control parameters through sharing experiences of autonomous mobile robots. *International Journal of System Science,* Vol. 29(7), 783–793, 1998.

[13] Pugh, J. and Martinoli, A. Multi-Robot Learning with Particle Swarm Optimization. *International Conference on Autonomous Agents and Multiagent Systems, Hakodate, Japan,* 441–448, 2006.

[14] Schultz, A.; Parker, L. Multi-Robot Systems: From Swarms to Intelligent Automata. Proceedings of the International Workshop on Multi-Robot Systems, Springer, 2002.

[15] Spears, W. Simple Subpopulation Schemes. *Proceedings of the Evolutionary Programming Conference, World Scientific,* 296–307, 1994.

[16] Tan M. Multi-Agent Reinforcement Learning: Independent vs. Cooperative Agents. In Proceedings of the Tenth International Conference on Machine Learning, 330–337, Morgan Kaufmann, 1993.

[17] Wu, A., Schultz, A. and Agah, A. Evolving Control for Distributed Micro Air Vehicles. *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation,* 174–179, 1999.