Novel Physicomimetic Bio-inspired Algorithm for Search and Rescue Applications

Rahul Rajan
U.S. Naval Research Laboratory
Washington, DC, USA
rahul.rajan@nrl.navy.mil

Michael Otte
National Research Council Postdoc
Washington, DC, USA
michael.otte.ctr@nrl.navy.mil

Donald Sofge
U.S. Naval Research Laboratory
Washington, DC, USA
donald.sofge@nrl.navy.mil

***Abstract*—Decentralized algorithms are often used for multi-agent search in scenarios in which it is difficult or computationally intractable to use a centralized control strategy. "Bottom-up" multi-agent search techniques, or algorithms that are based on local interactions between agents and their environment, allow for highly scalable and fault tolerant systems. Moreover, they often demonstrate remarkable emergent properties, such as foraging behavior and emulated states of matter, which can be found in bio-inspired swarm algorithms and physics-inspired methods. However, it is often difficult to control these systems, especially when area coverage and target localization are the primary goals. In this paper we propose using stigmergic techniques combined with physicomimetic force laws to guide a multi-agent system toward efficiently exploring a pre-defined search area. We found that our method provides a reliable and effective method for decentralized area coverage by a multi-agent system. Furthermore, we implemented our approach using Robot Operating System (ROS) and the Gazebo simulation environment.**

***Index Terms*—swarm, physicomimetics, stigmergy, pheromones, bio-inspired, search and rescue, coverage, bottom-up**

I. Introduction

Mass market production of inexpensive unmanned areal vehicles and lightweight cameras has increased the use of autonomous aircraft in search and rescue missions. The 2011 earthquake that hit Japan resulted in more than 4,000 people reported missing [1]. Hundreds of search and rescue workers risked their lives working through extremely hazardous conditions to rescue survivors and locate the missing. In 2017 multiple natural disasters including earthquakes in Mexico, and hurricanes Harvey, Irma, and Maria causing widespread destruction and flooding, left thousands stranded awaiting rescue. Unmanned robotic systems may be used to locate survivors while minimizing the risk to the lives of the search and rescue workers. Research is currently underway into use of low-cost UAV swarms for search and rescue, but efficient methods for controlling these robot swarms are still lacking. Key research is needed to develop algorithms that coordinate swarms of agents (robots) to effectively locate targets, or "survivors," in a search and rescue scenario.

In disaster relief scenarios it is important for these multi-agent systems to work in a decentralized system and to handle changes in agent population. This is especially critical in urban search and rescue where the probability of agents getting damaged and going offline is high due to hazardous conditions. In this paper we discuss a new bio-inspired and a physics-inspired decentralized method for guiding multi-agent systems towards high area coverage when communication and agent membership may be erratic. More specifically, repellent pheromones are used to guide agents toward unexplored regions of a given search space in order to increase the chances of locating a target. Agents employ specific rules that dictate when a pheromone should be placed. Additionally, they are repelled from previously laid pheromones using an inverse square law. In situations where a target's general location is known, attractive pheromones can be used to "encourage" the swarm to search the area where a target might be located. Furthermore, the approach is novel because it combines the versatility of physics-inspired solutions with the ability to remember past team behavior using stigmergy.

A. Inspiration from Nature

a) Stigmergy: Stigmergy [2, p. 518] refers to techniques that involve indirect communication between agents through the environment. The strategy is inspired by ant behavior. Ants drop attractive and repellent pheromone trails to communicate with other ants. Optimization techniques such as Ant Colony Optimization are heavily inspired by these pheromone trails. The basic premise is that as ants or "agents" look for food or "targets", they drop pheromone trails. Over time the trail with the highest scent ends up being the shortest path to the target, and the path more agents take. Moreover, the pheromone trails evaporate over time to prevent agents from converging on a locally optimal solution.

In this paper we look at pheromones not as trails, but as individual particles. Just as real pheromone trails decay in scent, our pheromones exhibit a similar evaporation model.

b) Physicomimetics: Physicomimetics refers to a class of swarm based algorithms that rely on using virtual forces to control multi-agent systems. This approach, sometimes referred to as "Artificial Physics" (AP) was introduced by William Spears and Diana F. Gordon in 1999 [3]. In their model, they used AP to create multi-agent formations solely using force laws between agents. Formations are typically achieved by having both repulsive and attractive zones around agents. This causes agents to maintain a fixed distance away from neighbors while still behaving as a group. Moreover, this rule allows swarms to form into geometric shapes such as

squares and hexagons. The forces are also restricted to act over a fixed radius around each agent, thereby ensuring locality and therefore scalability of the method with respect to the number of agents. Physicomimetics has since been applied to several different problem domains including path-planning [4], obstacle avoidance [5], surveillance [5], and search-and-rescue [6] tasks. Slightly changing the forces between agents can result in remarkably different behaviors, which may be useful for very different applications. For instance, in simulating swarms as states of matter, solids are useful in tasks that require a team formation, while gases can be useful in coverage tasks [7].

The key advantage of AP techniques is that the force only needs to be calculated at a given robot's position. As a result, it is not necessary to calculate the gradient of an entire potential field, allowing for significantly lower computational overhead. Spears and Gordon [3] cite this as the primary advantage of AP over previous potential field methods.

II. Previous Work

Multi-agent control has long been studied across a wide variety of fields, yet remains an active area of research. While centralized algorithms provide provably optimal solutions to many problems, the computational complexity of centralized methods limits their use to small numbers of robots and assume the existence reliable global communication. In contrast, decentralized algorithms can often scale beyond thousands of robots and only require intermittent local communication.

Decentralized coverage algorithms often rely on some form of a leader-follower paradigm for maintaining team formations [8, 9]. The use of pheromones or physicomimetics techniques to guide multi-agent systems is a popular technique for achieving distributed control. A number of papers in the last decade have investigated how stigmergy can be used to improve coverage and solve optimization problems. Specifically, the use of attractive and/or repellent pheromones to achieve coverage or target localization behavior in a team. Additionally, other research has built on the original work by [3] and developed AP-based algorithms for the purpose of surveillance. However, research into combinations of bio-inspired and physicomimetic techniques has been limited.

A. Pheromone Research

Research into using stigmergic techniques for multi-agent coverage typically uses repellent pheromones, and sometimes attractive pheromones to guide agents towards unexplored regions. Typically pheromones are used more as a heuristic to optimize another algorithm. For instance, [10] modeled the search space as a set of zones connected through a Markov model. Each zone then consists of connected "nodes"- as agents explore the zone they drop repulsive pheromones on explored regions and attractive pheromones on found targets. A probabilistic model based on pheromone concentrations in each node determines how an agent moves within the zone. However, this algorithm works concurrently with a Markov transition model, which uses transition probabilities obtained from a generational genetic algorithm to define agent mobility between zones. This approach makes a couple of key assumptions. First, a pre-processing step is necessary to divide the search area into zones and generate a Markov model. Second, it primarily relies on a global pheromone map updated and shared amongst all agents. Although a local map is used if the global map is not available, the algorithm is designed to work with global pheromone data.

Using probabilistic models based on pheromone concentrations is not a new concept. Other works have relied on agents preferring and moving towards regions with lower repellent pheromone count [11, 12, 13]. Other papers use pheromones to form attractive or repellent gradients or "trails" to move agents towards unexplored regions and away from explored, or currently being explored, regions [14]. We borrow and extend several ideas introduced in these papers to optimize our algorithm. For example, we place high weight repellent pheromones on the boundaries of the search space to keep agents inside, similar to a method described in [11]. Our use of repellent pheromones to prevent agents from exploring previously explored regions is discussed as a core aspect of many stigmergic coverage algorithms [10, 11, 12, 14]. The key difference in these algorithms is how these pheromones are stored and detected. Some use global pheromone maps to keep track of all dropped pheromones [10]. This approach may have an advantage over local map approaches [14, 11] in cases where the swarm has low connectivity, but a global map introduces the issue of a central point of failure and necessitates long range network connectivity that might not be available in a real scenario. Some papers also discuss using stigmergy in a more literal sense and dropping "real" pheromones in the environment [12]. Although this may advantageous due to its true bio-inspired nature, it is important to note that in a real implementation of their algorithm, a global pheromone map would likely be required.

Attractive pheromones, on the other hand, have often been used in target localization tasks [13]. When an agent locates a target, an attractive pheromone is placed to allow the team to move towards the target location. Although our algorithm could have incorporated this "feature" as well, it was outside the scope of our task, which was optimizing area coverage.

Pheromone based search methods tend to have a number of common limitations. For instance, they often exhibit poor control over individual agents. Simple pheromone approaches focus on overall swarm behavior, making it difficult to control or manage agent-agent interactions. This is especially important in tasks such as formation-keeping and new trajectory generation for exploration.

B. Physicomimetics Research

Research in physicomimetics algorithms for area coverage or surveillance tasks tends to revolve around modeling agents as gas particles. Some papers discuss using gas models because they allow swarms to achieve uniform coverage, even in obstacle laden areas [15]. Gas models allow for dispersion and can be useful in situations where accurate agent localization is not available [7]. Although gas models can guarantee coverage

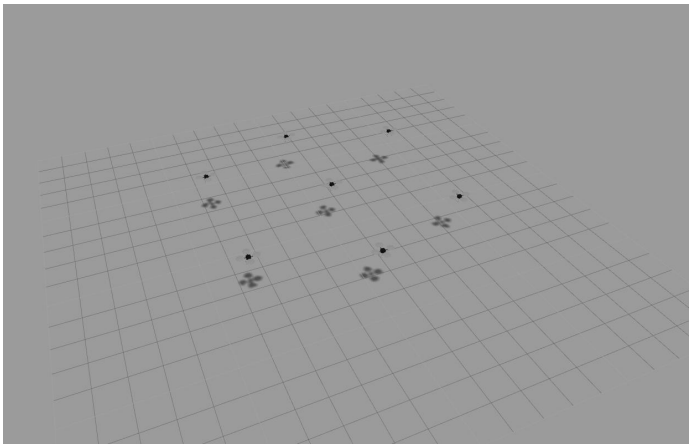

Figure 1. Low Energy Configuration for 7 Agents

in most environments, they are highly unpredictable and there's nothing preventing agents from revisiting previously explored regions. In [16] an approach is detailed combining gas dynamics through a Lennard-Jones potential with a temperature gradient model. When agents cover areas with no target present, the "temperature" below increases causing them to move down the temperature gradient toward colder regions.

Physicomimetics-based optimization techniques often run into the problem of agents not remembering past behavior. In an exploration scenario this can cause agents to unnecessarily cover the same area multiple times. Additionally, most AP algorithms work by minimizing the total potential energy of the system [7]. Thus, once this low energy state is achieved, it is difficult to "restart" the algorithm. Figure 1 shows one low energy configuration for 7 agents. For a gas-based model, external energy must be injected into the system.

III. Methodology

Our approach relies on combining ideas from typical stigmergic approaches with artificial force laws often found in multi-agent coverage solutions. Our goal is to show that combining physicomimetic and biomimetic techniques can be effective in area-coverage and target localization problems.

In the following sections, we describe the PherPhys algorithm, its features, and why it works.

A. Features

Our algorithm exhibits several features apparent in many similar decentralized, multi-agent search algorithms. For one, there is no global communication. All interactions occur between nearby agents. Pheromones placed by an agent in the environment are not immediately known to the entire team. Instead, this information spreads incrementally through local communication and as robots move through the environment. This allows for our approach to be scalable and robust to changes in agent membership. Second, this approach relies on virtual stigmergy, meaning pheromone locations and weights are updated on a virtual map, or in this case a grid representing the search area, that can be shared with other agents. This map is maintained locally by each agent and is shared with nearby neighbors, thus removing the need for global communication. Third, in order to prevent agents from being unable to access certain parts of the environment due to a high concentration of pheromones, a decay policy is used. After a predefined amount of time, all agents decay all of their "known" pheromones according to an exponential decay function. "Known" pheromones refers to pheromones stored in an agent's local pheromone map. In cases where complete connectivity is not possible, some agents may not have an updated map of pheromones.

$$w = w \cdot e^{-\delta} \qquad \text{Repeat every } \Delta t \text{ seconds}$$

The decay constant δ and the update interval Δt are especially important because together they define how long it takes for a given agent to revisit a certain area.

B. Functionality

The heart of the algorithm is inspired by Coulomb's law. The premise is that both pheromones and agents exhibit a charge. The agents' charge is left as a constant, while the pheromones' charge is defined as its "weight". The charge thus decreases over time according to the above decay function. Pheromones placed in the environment thus exhibit a charge on the agents in the environment. Thus, these forces can be summed vectorially to define the motion of the agents. The following equations define the motion of an agent as a result of a Coulombic force with total force F_T, number of particles/agents N, particle charge q_i, Coulomb's constant k, position vector $\vec{r}_i$, acceleration vector $\vec{a}$, particle/agent mass m, velocity vector $\vec{v}$, and time step Δt:

$$\vec{F}_T = \sum_{i=2}^{N} \frac{k q_1 q_2}{\|\vec{r}_i - \vec{r}_1\|^3} \cdot (\vec{r}_i - \vec{r}_1) \tag{1}$$

$$\vec{a} = \frac{\vec{F}_T}{m} \tag{2}$$

$$\vec{v} = \vec{a} \cdot \Delta t \tag{3}$$

As shown, the acceleration vector obtained from the summed Coulombic force is converted into a velocity vector by multiplying by a time step size defined in the simulation environment. The rule for dropping pheromones is based on area covered. More specifically, an agent drops a pheromone every time it covers a certain amount of area. This number is directly correlated with the density of pheromones in the search space and thus how long it takes an agent to revisit a region of the search area. Moreover, this constant together with δ and Δt must be set appropriately depending on the target detection model and topology of the search space. If the sensor model is defined by a high false negative rate, it is important that agents can come back to previously visited areas more quickly. However, if the sensor model has a low false negative rate, it is important that agents visit new areas more often. Nevertheless, a decay is still necessary to prevent agents from getting stuck in regions of the search space.

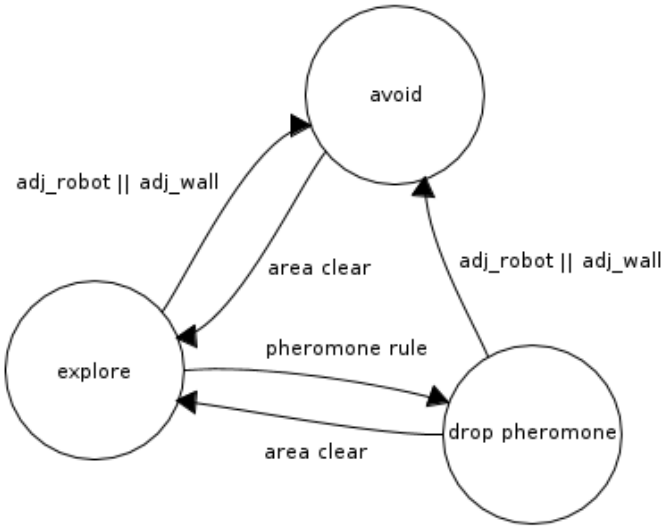

Figure 2. Finite State Machine

The actions of an agent are defined by a Finite State Machine with 3 states: explore, avoid, and drop pheromone. The transitions between these states are shown in Figure 2. It is important to note that the avoid state is given the highest priority as it prevents agents from colliding with other agents and leaving the search space. The "pheromone rule" is defined by the amount of area covered before a pheromone is dropped.

C. Solutions to Key Issues

There are a couple of key issues that had to be acknowledged and resolved in order for our approach to work effectively. The most important one was getting stuck in local minima. This primarily occurs when agents are located at the intersection of two potential fields from nearby pheromones. This results in agents oscillating between two pheromones for a long duration of time.

This is a very common problem and often mentioned in literature on physicomimetics and pheromone-based algorithms. For example, some research has been looking into using agents as "rescuers" when an agent is stuck in a local minimum [17]. The premise is that a rescuer would use a local force interaction to push an agent out of a local minimum. We adopted a similar policy to get agents out of local minima. However, most of the strategies we implemented involved directly changing the velocity of a stuck agent instead of the force acting on it. This allowed us to get agents "unstuck" fairly quickly. The methods we used to force agents out of local minima were:

- Add a small random velocity vector to each agent.
- Incorporate a "minimum distance" rule that only adds forces from pheromones that are at least a certain distance away from an agent.
- Implement a check to see if agents are exhibiting oscillatory behavior. If the check is true, we temporarily remove the effect of the pheromones and add a random velocity vector towards the center of the world for a small duration of time.

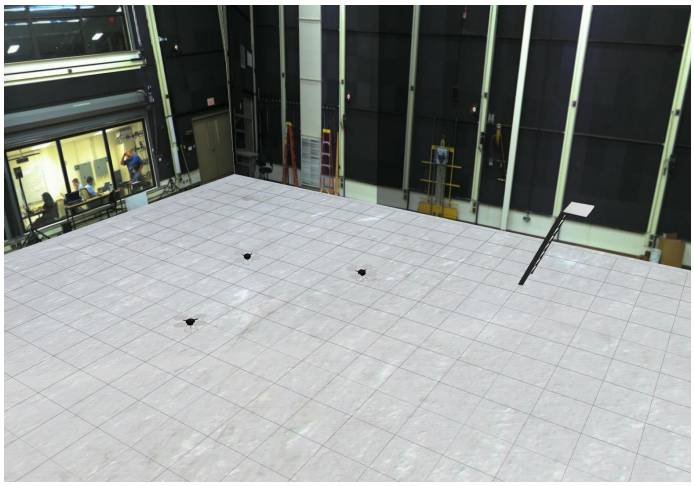

Figure 3. Model of NRL LASR's Prototyping High Bay in Gazebo

Together these approaches effectively resolved the issue of agents getting stuck in local minima without sacrificing exploration time.

Another issue we faced was keeping agents inside the search area. To resolve this we added permanent high-weight pheromones around the boundaries of the search area. This effectively kept agents in the search space and allowed them to continue exploration. Agents were still able to cover areas near the boundaries because the wall pheromones were only sensed when an agent was very close to an edge.

IV. Implementation

We employed ROS (Robot Operating System) to control the behavior of the agents and Gazebo for simulation. ROS is an established framework for writing robot software. It provides models, tools, and libraries that make it ideal for quickly testing algorithms in simulation and on real hardware. In this effort we used the $hector\_quadrotor$ package to model and control a simple quadrotor. The quadrotors or "agents" were simulated in the Gazebo simulation software because it provides a robust simulation platform and easily interfaces with ROS. Using a realistic simulation is important when testing multi-agent algorithms because we need to be able to simulate how they work in the real world. This is especially crucial when dealing with agent-agent interactions and ensuring possible collisions are eliminated. Figure 3 displays the simulation environment and model we used to test our algorithm.

A. Distributed Simulation

Simulating multi-agent systems can be extremely computationally expensive. To account for this we designed a distributed simulation framework that can merge multiple Gazebo instances running on different computers into a single simulation using an open source UDP Multicast Protocol. We

implemented this technique on an isolated network with wired connections to reduce network latency. Thus, this allowed us to effectively test our algorithm with high fidelity on a large number of agents. Further detail on this framework is left out as it is beyond the scope of this paper.

B. Agent Communication

Because agents resided on different computers, we employed a UDP protocol called Lightweight Communications and Marshalling (LCM) [18] to allow for local communication between nearby robots. Moreover, for the purposes of simulation, each agent knows the location of the other agents only to detect nearby agents. In a real scenario, proximity sensors would be employed for this task. As agents explore the search space and drop pheromones, they keep an internal map of their known pheromone locations. Each pheromone can be represented as a vector:

$$Pher =< x, y, w > \tag{4}$$

Where x and y define the location of the pheromone and w defines the weight. Agents store pheromones in a vector and can append new ones or remove existing ones. When agents need to merge their pheromones Algorithm 1 is employed where P and M are the pheromone vectors to be merged, and $Add()$ appends an item to a vector. In this approach, if both agents have copies of the same pheromone, the algorithm will choose the pheromone with a lower weight when merging.

Algorithm 1 Pheromone Map Merging Algorithm

```
0:  function MERGEPHER(P,M)
1:  for i ← 0 to M.Length do
2:      match ← −1
3:      for j ← 0 to P.Length do
4:          if M[i][0] = P[j][0] AND M[i][1] = P[j][1] then
5:              if P[j][2] > M[i][2] then
                    P[j][2] ← M[i][2]
6:              end if
7:              match ← 1
8:          end if
9:      end for
10:     if match = −1 then
            P.Add(M[i])
11:     end if
12: end for
13: return P
```

Although Algorithm 1 is $O(n^2)$, which may be computationally expensive when dealing with large numbers of pheromones, it is only used when agents are in close proximity, and pheromones with 0 weight are automatically removed.

C. Agent Control

The algorithm for calculating the velocity of an agent is much less expensive, with a complexity of $O(n)$. Algorithm 2 describes how the forces are summed and the resulting velocity is calculated where P is an agent's vector of known

Algorithm 2 Agent Movement Algorithm

```
0:  function PHERPHYS(P,quad)
    my_x ← getX(quad)
2:  my_y ← getY(quad)
    dt ← 0.001
4:  for i ← 0 to P.Length do
        pher_x ← P[i][0]
6:      pher_y ← P[i][1]
        pher_weight ← P[i][2]
8:      dx ← my_x − pher_x
        dx ← my_y − pher_y
10:     if Distance(pher_x, pher_y, my_x, my_y) < 4 then
            dP_mag ← ((dx ^ 2 + dy ^ 2) ^ 0.5) ^ 3
12:         fx ← ((k ∗ q ∗ pher_weight)/(dP_mag)) ∗ dx
            fy ← ((k ∗ q ∗ pher_weight)/(dP_mag)) ∗ dy
14:         Ft_x ← Ft_x + fx
            Ft_y ← Ft_y + fy
16:     end if
    end for
18: ax ← Ft_x/quad.mass
    ay ← Ft_y/quad.mass
20: vx ← (ax ∗ dt + getRandom())
    vy ← (ay ∗ dt + getRandom())
22: Publish(vx, vy)
```

pheromones, $quad$ represents the agent, my_x and my_y define the current location of the agent, $pher\_x$ and $pher\_y$ define the location of a pheromone, $pher\_weight$ defines the weight of a pheromone, ax, ay, vx, and vy define the calculated acceleration and velocity. Note the addition of a random component $getRandom()$ to keep agents from becoming stuck in local minima.

The logic used in Algorithm 2 is also employed when agents come near each other to prevent collisions. To keep the vector math simple, the heading or yaw of all the agents is kept constant. Agents are moved by controlling their linear velocities in the x, y, and z directions.

Agent mobility is handled primarily by the Gazebo physics engine. Our algorithm publishes velocities for each agent to the simulation server and Gazebo automatically determines a way to change the agent's velocity realistically. This is important because abrupt changes in velocity are not possible in a real world situation and may lead to unrealistic results.

V. EXPERIMENTS

We compared our $PherPhys$ algorithm to that of a random walk. We performed repeated trials in simulation, and recorded area coverage over time as well as number of targets detected. We experimented with 3 team sizes: 3, 5, and 7. We performed 20 trials for each algorithm and team size, with 1 randomized target location per trial. The parameters used for each algorithm are shown in Table I.

In each case the $PherPhys$ algorithm performed at least 10% better than a pure random search. Our implementation of a random walk involved agents changing their velocities

Algorithm	Decay Factor	Grid Size	Velocity Limit
PherPhys	0.3	20m x 20m	0.6 m/s
Random	N/A	20m x 20m	0.6 m/s

Table I
EXPERIMENTAL PARAMETERS

3 Agent Statistics		
Group	**PherPhys**	**Random**
Mean	310.20	267.00
SD	22.21	18.31
SEM	7.02	5.52
N	20	20

Table II
STATISTICS FROM 3 AGENT COVERAGE FOR PHERPHYSAND RANDOM

by a small amount at every time step. To ensure agents stay inside the search space and don't collide with other agents, Coulombic forces are used only between agents and walls. The force is kept small to ensure we had a proper baseline to compare our algorithm against. The results show that agents in the $PherPhys$ algorithm have a higher chance of visiting new search areas. We found the difference between the overall coverage in all team sizes was statistically significant with better than 95% confidence and p-value < 0.0001. Moreover, the standard deviation of final coverage is much lower in the $PherPhys$ compared to random, indicating high consistency in area coverage. Tables II, III, and IV show the statistical results for 3, 5, and 7 agents respectively. We proved statistical significance using an unpaired t-test.

The results for target detection are shown in Table V. Overall our algorithm detects more targets compared to Random. Although the difference may not seem significant, it is important to note that in a search and rescue scenario, finding just 1 survivor is significant. That being said, further experiments will have to be conducted to measure how much better our algorithm is compared to other approaches.

When comparing agent sizes on our algorithm, we notice that increasing team size has a noticeable and significant

5 Agent Statistics		
Group	**PherPhys**	**Random**
Mean	338.87	290.8
SD	16.23	18.13
SEM	4.19	4.05
N	20	20

Table III
STATISTICS FROM 5 AGENT COVERAGE FOR PHERPHYSAND RANDOM

7 Agent Statistics		
Group	**PherPhys**	**Random**
Mean	356.10	293.85
SD	4.31	14.92
SEM	1.36	3.34
N	20	20

Table IV
STATISTICS FROM 7 AGENT COVERAGE FOR PHERPHYSAND RANDOM

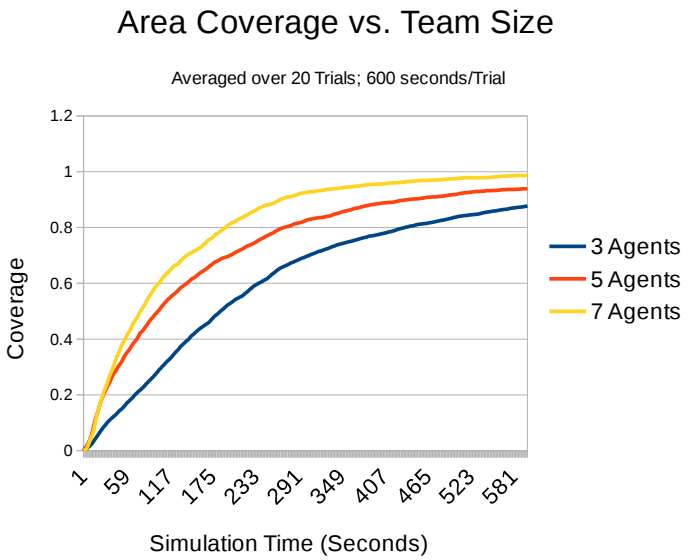

Figure 4. Area Covered vs. Team Size Using PherPhys Algorithm

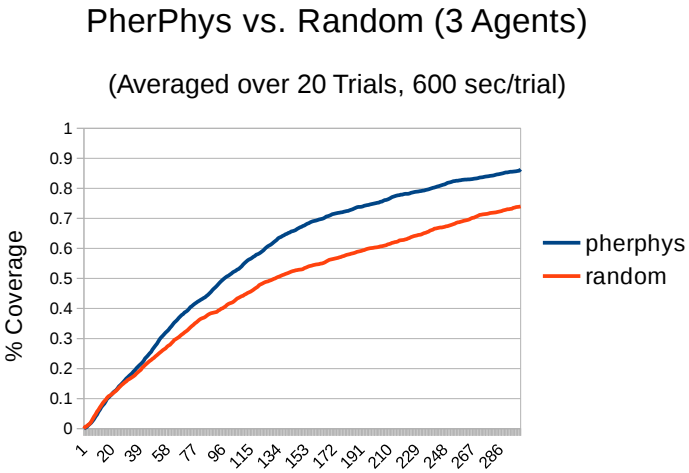

Figure 5. PherPhys vs. Random Algorithm with 3 Agents (300 Steps. 2 seconds/step)

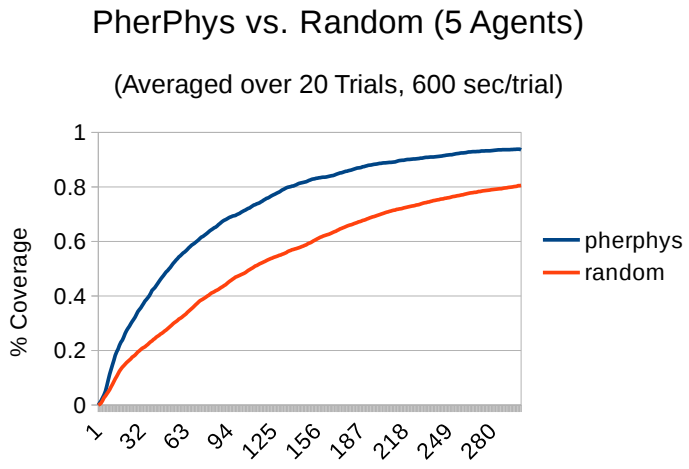

Figure 6. PherPhys vs. Random Algorithm with 5 Agents (300 Steps. 2 seconds/step)

Agents	PherPhys	Random
3	19	15
5	18	18
7	20	18

Table V
STATISTICS FOR TARGET DETECTION

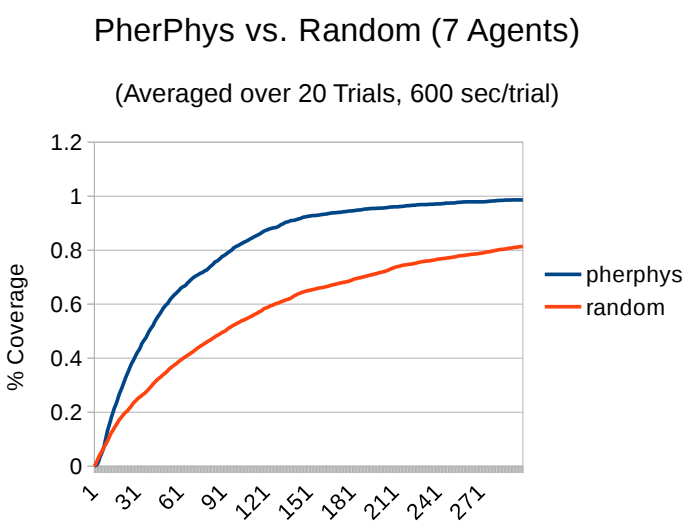

Figure 7. PherPhys vs. Random Algorithm with 7 Agents (300 Steps. 2 seconds/step)

impact on the area coverage. The results show a clear advantage for the $PherPhys$ over a pure random walk. We used a random walk as an equivalent to the gas-model in a physicomimetics-based exploration strategy. To serve as a baseline, we also implemented a pheromone-only algorithm. In this approach, each agent checks the pheromone density in 4 locations: up-left, up-right, down-left, down-right. The agent then moves towards the quadrant with the least density. However, even though this is a pheromone-only approach, we still included the physicomimetic repulsion between agents for the sole purpose of collision avoidance. We also kept a repulsive force from the pheromones on the edges of the search space to keep agents in. Experiments evaluating this model are still ongoing but preliminary results indicate that our pheromone-only model fails to reach optimal coverage. Future experiments will test a more accurate version of this model. Based on these results, it can be concluded that a combination of physicomimetic and stigmergic techniques is a viable solution for the coverage problem. Future work will compare our approach to centralized approaches such as lawnmower paths and Boustrophedon coverage algorithms.

VI. CONCLUSION

In this paper we presented a new distributed, bio-inspired, and physics-inspired search algorithm for solving the classic area coverage or surveillance problem, specifically focusing on applications in search and rescue scenarios. The $PherPhys$ algorithm was both scalable and robust due to its highly decentralized nature. We were able to demonstrate that the algorithm performs better than a baseline random walk. Moreover, we showed that the algorithm performs better in terms of area coverage with more agents.

In future work we will incorporate an attractive pheromone model that can be used when the general location of a target is known. A possible application of this would be in a search and rescue scenario where a human operator knows the approximate location of a survivor and can guide the robot team towards that area. We will use machine learning and evolutionary techniques to optimize starting pheromone weight, decay factor, and a policy for dropping a pheromone to optimize area coverage.

REFERENCES

[1] M. Kazama and T. Noda, "Damage statistics (summary of the 2011 off the pacific coast of tohoku earthquake damage)," *Soils and Foundations*, vol. 52, no. 5, pp. 780 – 792, 2012. Special Issue on Geotechnical Aspects of the 2011 off the Pacific Coast of Tohoku Earthquake.

[2] D. Floreano and C. Mattiussi, *Bio-Inspired Artificial Intelligence: Theories, Methods, and Technologies*. The MIT Press, 2008.

[3] W. M. Spears and D. F. Gordon, "Using artificial physics to control agents," in *Proceedings 1999 International Conference on Information Intelligence and Systems (Cat. No.PR00446)*, pp. 281–288, 1999.

[4] C. W. Warren, "Global path planning using artificial potential fields," in *Proceedings, 1989 International Conference on Robotics and Automation*, pp. 316–321 vol.1, May 1989.

[5] W. Kerr, D. Spears, W. Spears, and D. Thayer, *Two Formal Gas Models for Multi-agent Sweeping and Obstacle Avoidance*, pp. 111–130. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005.

[6] P. Song and V. Kumar, "A potential field based approach to multi-robot manipulation," in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*, vol. 2, pp. 1217–1222, 2002.

[7] W. Kerr, "Gas-mimetic swarms for surveillance and obstacle avoidance," in *Physicomimetics*, pp. 193–221, Springer, 2011.

[8] P. Dasgupta, T. Whipple, and K. Cheng, "Effects of multi-robot team formations on distributed area coverage," *Int. J. Swarm. Intell. Res.*, vol. 2, pp. 44–69, Jan. 2011.

[9] A. Agarwal and S. R. Sahoo, "Target centric area coverage control using formation in a multi-agent system," in *2017 Indian Control Conference (ICC)*, pp. 315–320, Jan 2017.

[10] E. Kieffer, G. Danoy, P. Bouvry, and A. Nagih, "Hybrid mobility model with pheromones for uav detection task," in *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1–8, Dec 2016.

[11] E. Kuiper and S. Nadjm-Tehrani, "Mobility models for uav group reconnaissance applications," in *2006 Interna-*

tional Conference on Wireless and Mobile Communications (ICWMC'06), pp. 33–33, July 2006.

[12] P. Gaudiano, B. Shargel, E. Bonabeau, B. Clough, and I. C. C. MA., *Swarm Intelligence: A New C2 Paradigm with an Application to Control Swarms of UAVs*. AD-a418 490, Defense Technical Information Center, 2003.

[13] C. Atten, L. Channouf, G. Danoy, and P. Bouvry, *UAV Fleet Mobility Model with Multiple Pheromones for Tracking Moving Observation Targets*, pp. 332–347. Cham: Springer International Publishing, 2016.

[14] M. Antal, I. Tamas, T. Cioara, I. Anghl, and I. Salomie, "A swarm-based algorithm for optimal spatial coverage of an unknown region," in *2013 IEEE 9th International Conference on Intelligent Computer Communication and Processing (ICCP)*, pp. 7–13, Sept 2013.

[15] W. M. Spears and D. F. Spears, *Physicomimetics: Physics-Based Swarm Intelligence*. Springer Publishing Company, Incorporated, 2012.

[16] N. Sydney, D. A. Paley, and D. Sofge, "Physics-inspired motion planning for information-theoretic target detection using multiple aerial robots," *Autonomous Robots*, vol. 41, pp. 231–241, Jan 2017.

[17] L. S. Marcolino and L. Chaimowicz, "No robot left behind: Coordination to overcome local minima in swarm navigation," in *2008 IEEE International Conference on Robotics and Automation*, pp. 1904–1909, May 2008.

[18] A. S. Huang, E. Olson, and D. Moore, "LCM: Lightweight communications and marshalling," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Taipei*, pp. 4057–4062, oct 2010.