A stochastic, swarm-based control law for emergent system-level area coverage by robots

Adam M. Schroeder

University of Toledo

Follow this and additional works at: http://utdr.utoledo.edu/theses-dissertations

Recommended Citation
http://utdr.utoledo.edu/theses-dissertations/2107

This Thesis is brought to you for free and open access by The University of Toledo Digital Repository. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of The University of Toledo Digital Repository. For more information, please see the repository's About page.
A Thesis

entitled

A Stochastic, Swarm-Based Control Law for Emergent System-Level Area Coverage by Robots

by

Adam M. Schroeder

Submitted to the Graduate Faculty as partial fulfillment of the requirements for the Masters of Science Degree in Mechanical Engineering

________________________
Dr. Manish Kumar, Committee Chair

________________________
Dr. Abdollah Afjeh, Committee Member

________________________
Dr. Brian Trease, Committee Member

________________________
Dr. Patricia R. Komuniecki, Dean College of Graduate Studies

The University of Toledo

May 2016
An Abstract of

A Stochastic, Swarm-Based Control Law for Emergent-System-Level Area Coverage by Robots

by

Adam M. Schroeder

Submitted to the Graduate Faculty as partial fulfillment of the requirements for the Master of Science Degree in Mechanical Engineering

The University of Toledo

May 2016

This work proposes a stochastic, swarm-based control law for providing system-level area coverage in robots. In the first half of the work, it was investigated how a decentralized, ant-inspired, virtual pheromone-based method of area coverage performed when important parameters like rate of pheromone diffusion, rate of pheromone evaporation, and introduced noise were varied. Although this type of control scheme has been studied in literature, the interdependent sensitivity to these parameters has not been investigated. It was shown that the most influential of these parameters is the introduced noise.

Part of this investigation included devising appropriate performance metrics, which were selected to measure the rate, exhaustivity, and frequency of area coverage. After optimal values for diffusion, evaporation, and noise were obtained, these values were used in the second half of this work. The investigation was then expanded to study the effect of using gradient following in combination with Lévy flight, which takes variable path lengths from a power-law distribution. Lévy flight had been shown to be effective in robot search, but had not yet been applied to area coverage. It was shown that this combination of
gradient following and Lévy flight provides superior area coverage and pop-up threat detection.
This work is dedicated to Michelle, and Linus, and to learning more quickly than we forget.
Acknowledgements

I gratefully acknowledge the many contributions of Dr. Manish Kumar to this work. His time and efforts were indispensable, and especially in hindsight, I recognize the wisdom and prudence of his council throughout the research process.

I also would like to humbly acknowledge the support and kindness of my lab partners Ali, Reza, Sarim, Jisheng, Yixuan, and Padma who made it a joy to work together.
# Table of Contents

Abstract ........................................................................................................................................ iii

Acknowledgements .................................................................................................................... v

Table of Contents .................................................................................................................... vi

List of Tables .............................................................................................................................. x

List of Figures ............................................................................................................................ xi

List of Abbreviations ................................................................................................................ xiii

List of Symbols ........................................................................................................................ xiv

1 Introduction and Background ................................................................................................. 1

1.1 Area Coverage .................................................................................................................... 2

1.1.1 Area Coverage Applications ....................................................................................... 2

1.1.2 Single Agent versus Multi-Agent Systems ................................................................. 4

1.1.3 Area Coverage as a Special Form of Search .............................................................. 4

1.2 Chemotaxis and Random Walks ....................................................................................... 5

1.2.1 Ant Model of Chemotaxis ........................................................................................... 5

1.2.2 E. Coli Chemotaxis and Random Walks ................................................................... 7

1.2.3 Lévy Flight and Brownian Motion ............................................................................ 8

1.2.4 Lévy Foraging Hypothesis ......................................................................................... 9

1.3 Swarm Intelligence ............................................................................................................ 10

1.3.1 Return to Any Model ................................................................................................ 11
1.3.2 Other Examples of Swarm Intelligence in Nature ......................11
1.3.3 Central vs. Decentral/Distributed Control ............................12
1.3.4 Advantages of Swarm Intelligence Approach ..........................13
1.4 Combining Area Coverage, Biased Walks, and Swarm Intelligence ..........13
1.5 Literature Review and Research Gaps ..............................................14
  1.5.1 Pheromone-Based Research ..................................................14
  1.5.2 Other Swarm Research .......................................................17
  1.5.3 Lévy Flight Swarm Research ................................................18
  1.5.4 Other Multi-Agent Area Coverage Approaches ..........................18
1.6 Objective and Organization of Thesis ...........................................19

2 Problem Formulation ........................................................................21
  2.1 Defining a Mathematical Model ..................................................21
  2.2 Defining Performance Metrics ....................................................23
    2.2.1 Metric 1-Percent Area Coverage Integral ...............................23
    2.2.2 Metric 2-Visit Entropy .......................................................25
    2.2.3 Metric 3-Pop-Up Threat Detection ........................................28
    2.2.4 Metric Shortcomings .......................................................29

3 Implementation ...............................................................................30
  3.1 Summary of Cases .......................................................................30
  3.2 General Implementation Details ..................................................31
    3.2.1 Biased Walk Implementation Details .....................................31
    3.2.2 Constant and Variable Path Length Implementation Details ....33
  3.3 Performance Metric Implementation ..........................................37
### Contents

3.4 Implementation Sequence .................................................................................................38

- 3.4.1 Comprehensive Survey of Evaporation, Diffusion, and Noise ..........................38
- 3.4.2 Introduction of Lévy Flight and Comparison of Cases ......................................39

3.5 Implementation in MATLAB ..........................................................................................40

4 Results ..................................................................................................................................41

- 4.1 Survey of Evaporation, Diffusion, and Noise ..........................................................41
- 4.2 Comparison of Three Cases .........................................................................................48

5 Discussion .............................................................................................................................52

- 5.1 Survey of Evaporation, Diffusion, and Noise ..........................................................52
  - 5.1.1 Impact of Noise ........................................................................................................52
  - 5.1.2 Impact of Diffusion ..................................................................................................53
  - 5.1.3 Impact of Evaporation ............................................................................................53
  - 5.1.4 General Survey Results .........................................................................................54
- 5.2 Comparison of Three Cases ..........................................................................................56
  - 5.2.1 Area Coverage ........................................................................................................56
  - 5.2.2 Entropy .....................................................................................................................57
  - 5.2.3 Pop-Up Threat Detection .........................................................................................60
  - 5.2.4 General Comparison Results ..................................................................................62
  - 5.2.5 Contrasting Control Law and Lévy Foraging Hypothesis .....................................63
- 5.3 Limitations and Open Research Areas .........................................................................63

6 Conclusion ............................................................................................................................67
References .................................................................................................................................72

A  Program Code Written in MATLAB ....................................................................................78
List of Tables

3.1 Overview of Three Cases (Combinations of Gradient Following and Lévy)........30
3.2 Overview of Parameter Combinations and Cases Tested in 1st Round .................38
3.3 Overview of Parameter Combinations and Cases Tested in 2nd Round.................40
List of Figures

1-1 Pheromone Trails Established between Ants’ Nest and Food Sources ...............7
1-2 E. Coli in Both Movement Phases .................................................................8
1-3 Heavy-tailed Power-Law Distribution .............................................................9
1-4 Centralized, Decentralized, Distributed Control Comparison .......................12
2-1 Typical Curve of Fraction of Area Covered over Time ..................................24
3-1 Typical Pheromone Field and Corresponding Position History .......................33
3-2 Heavy-tailed Distribution for Different Alpha Values ......................................34
3-3 Single Agent Path: Constant-Length Segments vs. Variable-Length Segments ...36
3-4 Multi-Agent Path: Constant-Length Segments vs. Variable-Length Segments ....36
4-1 $P_1$ for Range of Noise, Diffusion, and Evaporation .....................................42
4-2 $P_1$ for a Range of Diffusion Rates with Noise Values in Separate Frames ....44
4-3 $P_1$ for a Range of Diffusion Rates with Evaporation Values in Separate Frames..45
4-4 $P_1$ for a Range of Evaporation Rates with Noise Values in Separate Frames ......46
4-5 $P_1$ for a Range of Evaporation Rates with Diffusion Values in Separate Frames..47
4-6 $P_1$ for all Cases in Part II over a Range of Alpha Values ...............................49
4-7 $P_2$ for all Cases in Part II over a Range of Alpha Values ...............................49
4-8 $P_3$ for ‘Simple’ Threats for all Cases in Part II over a Range of Alpha Values ....51
4-9 $P_3$ for ‘Smart’ Threats for all Cases in Part II over a Range of Alpha Values ......51
5-1 Different Stages of Area Coverage over Time ................................................55
<table>
<thead>
<tr>
<th>Page</th>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>5-2</td>
<td>Distribution of Visit Frequencies for all Cases in Part II</td>
</tr>
<tr>
<td>5-3</td>
<td>Distribution of Visit Frequencies for Two Different Alpha Values</td>
</tr>
<tr>
<td>5-4</td>
<td>Final Pheromone Field for Gradient Following with and without Lévy flight</td>
</tr>
<tr>
<td>5-5</td>
<td>Simple Example of System Entropy for Two Vastly Different Scenarios</td>
</tr>
</tbody>
</table>
List of Abbreviations

FTCS..........................Forward-Time Center Space (Discretization Scheme)
List of Symbols

\( \delta \) ....................... Dirac delta function
\( \Delta \) ....................... Gradient
\( \nabla \) ....................... Laplacian
Log .......................... Base 2 logarithm

\( \alpha \) ....................... Parameter for power-law distribution
\( \beta \) ....................... Intermediate value used in FTCS scheme
\( \chi \) ....................... Chemotactic sensitivity
\( \gamma \) ....................... Rate of evaporation
\( \rho \) ....................... Average areas visited per unit time
\( \mu \) ....................... Binary value tracking if threat has been detected
\( \sigma \) ....................... Magnitude of noise
\( \varphi \) ....................... Visitation frequency
\( \bar{\varphi} \) ....................... Average visitation frequency

\( \mathbf{A} \) ....................... Number of agents
\( \mathbf{A}_{\text{total}} \) ....................... Total area
\( \mathbf{A}_{\varphi k} \) ....................... Total area visited with frequency \( \varphi_k \)
\( \mathbf{A}_{\varphi \text{ijk}} \) ....................... Binary value tracking if \( \mathbf{A}_{ij} \) visited at frequency \( \varphi_k \)
\( \mathbf{a} \) ....................... Concentration of agents
\( \mathbf{b} \) ....................... Concentration of pheromone
\( \mathbf{D}_a \) ....................... Diffusion coefficient of agents
\( \mathbf{D}_b \) ....................... Diffusion coefficient of pheromone
\( \mathbf{e} \) ....................... Tessellation index
\( f \) ....................... Death/growth rate
\( g \) ....................... Pheromone production rate
\( H \) ....................... Entropy
\( h \) ....................... Pheromone deterioration rate
\( i \) ....................... Spatial index in 1\textsuperscript{st} dimension
\( j \) ....................... Spatial index in 2\textsuperscript{nd} dimension
\( k \) ....................... Visitation frequency index
\( m \) ....................... Fraction of area explored
\( n_{\text{visit}} \) ....................... Number of visits
\( P_{1} \) ....................... Percent area coverage integral performance metric
P_2.......................... Visitation entropy performance metric
P_3.......................... Pop-Up threats detected performance metric
Prob...................... Probability
p............................ Threat index
Q............................ Cumulative distribution function
q............................ Probability density function
R............................ Real domain
R_a......................... Position of agent a
\dot{R}_a................... Velocity of agent a
r............................ Continuous space domain
s............................ Intermediate value in finding variable path length
T............................ Number of tessellations
t............................ Time
u............................ Uniformly distributed random variable
v_a......................... Velocity of agent a
visited........................ Binary value in continuous space tracking if area has been visited
visited_{ij}................ Binary value in discrete space tracking if area ij has been visited
W............................ Wiener process vector
x............................ Position in 1st dimension
y............................ Position in 2nd dimension
Chapter 1

Introduction

This work represents the confluence of three distinct topics and ideas: (1) area coverage and measures of performance, (2) biased random walks, and (3) swarm intelligence. As each topic is individually introduced below, the rationale for employing a swarm-based control law that is stochastic, decentralized, ant-inspired, and virtual pheromone-based will become clear. In such a pheromone-based system, behavior is driven by environmental mechanisms that control the pheromone’s spatial distribution and concentration. The first part of this work studied the influence of each of these environmental mechanisms as well as the contribution of noise, which makes the behavior of the system stochastic.

The second part of this work investigated how utilizing a particular type of random walk, called Lévy flight (pronounced like ‘levee’), can improve area coverage performance. After this introduction and background, the context of this work within the existing body of research and explanation of major research gaps that this work fills are detailed.
1.1 Area Coverage

Area coverage, the first of three distinct topic areas in this work, is a term used in the context of robotics to mean the generalized behavior of modifying or gaining information about a particular area. In the case of mobile robots, this is often understood to mean physically moving throughout this particular area. Area coverage is a worthy field of study because many diverse and powerful applications of area coverage behavior can be realized. To grasp the breadth of applications and important criteria when implementing area coverage, consider the following classification paradigm.

1.1.1 Area Coverage Applications

An effective way of classifying area coverage applications is to consider what performance measure is most appropriate for evaluating performance. In some applications, exhaustive coverage, or guaranteed coverage of the entire area is the highest priority. A good example of an application where exhaustive coverage is critical is the detection and decommissioning of land mines. Additional examples of this type are searching for carbon monoxide leaks within a home, mowing a lawn, and painting the body of a car. For all of these tasks, success is contingent upon providing one hundred percent coverage.

A second application type emphasizes the rate of area coverage, potentially without requiring exhaustive coverage. Searching for new mineral deposits or oil reserves does not require exhaustive coverage; rather the emphasis is on quickly locating the most promising sites. Animals foraging for food are often searching for sparse food sources, and thus covering area quickly is more important than exhaustively searching an area. A further example is a rover used for planetary exploration, which is interested in exploring only
discrete sites of interest that warrant further investigation. For these tasks, some concessions can be made to the completeness of the coverage if the rate of coverage is sufficiently fast.

Another application set emphasizes the need to detect pop-up threats. A wildfire is a fitting example of a dynamic environment where detection of hot spots, or areas where a wildfire would be most likely to spread, is the priority. The term pop-up threat is commonly used, but does not necessarily mean a military threat. Rather, a pop-up threat can more generally mean a spatiotemporal condition within a dynamic environment. An agricultural example of this type of application is the detection of harmful weeds within a field. Another example would be in the detection of the locations of probable avalanches. These occurrences of wildfire hotspots, weeds, and avalanches are all a type of pop-up threat.

In reality, most applications cannot be simply classified and may require a combination of exhaustive coverage, a fast rate of coverage, and detection of pop-up threats, among other criteria. Consider the application of removing snow and ice from a large commercial parking lot. Ideally, all snow would be removed, the entire area could be covered quickly, and especially icy locations or drifts could be detected and treated. Some of these criteria are competing, and the task of designing an optimal area coverage behavior is a multi-objective optimization problem.

For some systems, minimizing the energy consumed is the most important criteria. When considering a physical system, this often takes the form of minimizing abrupt directional changes which require deceleration and subsequent acceleration. A more concrete example is the change in orbital position of a satellite used to provide aerial
imagery. The high energy cost of such a change in position means that minimizing energy consumed is paramount.

Predictability is another criterion, but it is unique in that in some cases it is beneficial to be unpredictable, while in others it is advantageous to be predictable or consistent. For example, predictability in a surveillance application could allow the surveillance targets to evade detection by exploiting regular coverage patterns. However, in most industrial applications, for a myriad of reasons it is desirable that a situation perform the same during every cycle.

Three metrics were developed that are measures of coverage exhaustivity, rate of coverage, and coverage frequency and these new metrics can be used to help evaluate the suitability of an area coverage scheme for diverse applications.

There are some existing measures of area coverage performance, although a standard does not exist. Measures of area in use include: probability of time between visits [1], visitation frequency [2][3], time to complete 95% or 99% coverage [3][4], percent of area covered [1]–[3], [5]–[7], percent of targets detected/success rate in detecting targets[5][8], and time to discover/destroy targets [9][10].

1.1.2 Single Agent versus Multi-Agent Systems

The task of providing area coverage can be performed more effectively if multiple agents are used. Thus, the focus of this work was on multi-agent systems. The consequent challenge is how to coordinate the individual agents to maximize total system performance.

1.1.3 Area Coverage as Special Form of Search

Along with area coverage, search represents another generalized robotic behavior. The two are, in fact, intertwined in many cases. Examples given above regarding locating
new mineral and oil deposits, foraging for food, and detecting wildfire hotspots could also be considered a form of search. These examples all illustrate the search for *discrete* objects, but this can also be extended to the search for *continuous* objects or quantities. Reconsider the example of removing snow from a parking lot. Instead of viewing this scenario as an area coverage application, it can be phrased as a search for a continuous resource (snow), with a particular spatial distribution. Moreover, the object or resource being searched for need not be a physical quantity. Observing an area can be considered a search for *information*, a non-physical quantity. More information can be gained by visiting an area that has not yet been visited, or has not been recently visited.

The importance of this relationship is that it allows almost any area coverage problem to be generalized to a search problem. Search is a very well-researched field with a wide range of tools and techniques. Phrasing the area coverage problem in this way allows these well-developed search tools to be used in a new way.

### 1.2 Chemotaxis and Random Walks

The topic of biased random walks is the second of three critical topic areas. Chemotaxis, which can result in a biased random walk, is the movement of an organism in response to a chemical stimulus. This includes either moving toward something beneficial, like food, or moving away from something harmful, like a poison. Chemotaxis is of interest because it often is used as foraging technique which can then be potentially applied to the area coverage problem.

#### 1.2.1 Ant Model of Chemotaxis

An excellent model of chemotaxis in nature occurs in the foraging of ant colonies. As individual ants return from a food source, they emit a chemical pheromone. Other ants
foraging for food can detect this pheromone and its relative concentration and they make probabilistic decisions of where to move based on this information. If the pheromone concentration is relatively strong, they are likely to follow that pheromone trail. If the relative pheromone concentration is relatively weak, they are more likely to proceed in another direction. This simple movement pattern based on the strength of the chemical pheromone is a type of biased random walk. The term biased indicates that the movement is not completely random, but rather biased in the direction of the strongest pheromone concentration.

Pheromone serves as an indirect means of communication. It is indirect because ants are not communicating directly, but rather through modification of their environment. This type of communication is called stigmergy and it enables a single ant to communicate with many more ants than any type of direct communication could accomplish.

The efficacy of stigmergy is enhanced through environmental mechanisms; namely diffusion and evaporation. Over time, the pheromone diffuses so that not only is it detectable at its original point source, but also in the vicinity of the original source. This diffusion effectively disseminates the information which has been embedded as pheromone. Evaporation of the pheromone over time provides a mechanism for ‘forgetting’ old information. In a dynamic environment, if a food source expires, the pheromone trail leading to this food source will eventually deteriorate. This means ants will make decisions based only on more recent information. Figure 1-1 illustrates how pheromone trails to and from food sources are formed and how older trails from expired food sources evaporate over time.
Another classic example of chemotaxis is in the foraging patterns of individual E. Coli cells. E. Coli is a type of a bacteria which uses its flagella for locomotion. It can rotate the flagella counter-clockwise, which aligns the flagella into a single rotating bundle, and thus allows the bacteria to swim in a straight line. Alternatively, the flagella can be rotated clockwise, which breaks the flagella out of a bundle and causes the bacteria to tumble and realign itself, as illustrated in Figure 1-2. These two capabilities allow the bacteria to move in two phases: (1) swim phase, and (2) tumble phase. The probability of being in either phase depends on the relative strength of a chemical concentration, which indicates the presence of a food source. If the bacteria detect that the chemical concentration is strengthening, it is most likely to continue swimming in a straight direction. If it detects a weakening in the gradient, it is likely that the bacteria will tumble, which will reset its direction. Like for the ants, this simplified movement model is an example of a biased random walk.

**Figure 1-1**: Pheromone trails established between the nest and food sources and trails leading to expired food sources deteriorating. Ants discover new food sources and begin to establish new pheromone trails.

### 1.2.2 E. Coli Chemotaxis and Random Walk

Another classic example of chemotaxis is in the foraging patterns of individual E. Coli cells. E. Coli is a type of a bacteria which uses its flagella for locomotion. It can rotate the flagella counter-clockwise, which aligns the flagella into a single rotating bundle, and thus allows the bacteria to swim in a straight line. Alternatively, the flagella can be rotated clockwise, which breaks the flagella out of a bundle and causes the bacteria to tumble and realign itself, as illustrated in Figure 1-2. These two capabilities allow the bacteria to move in two phases: (1) swim phase, and (2) tumble phase. The probability of being in either phase depends on the relative strength of a chemical concentration, which indicates the presence of a food source. If the bacteria detect that the chemical concentration is strengthening, it is most likely to continue swimming in a straight direction. If it detects a weakening in the gradient, it is likely that the bacteria will tumble, which will reset its direction. Like for the ants, this simplified movement model is an example of a biased random walk.
1.2.3 Lévy Flight and Brownian Motion

Pure random walks, as opposed to the biased random walks of ants and E. coli, are paths consisting of a sequence of random steps. Random walks have been used to model a bevy of probabilistic phenomena including stock market movements [11], animal foraging [12], and the spread of health epidemics [13].

Within the broader scope of random walks, some specific examples are of interest. Lévy Flight, named after a French mathematician, is a type of random walk where the step size for each path segment is taken from a ‘heavy-tailed’ probability distribution, as shown in Figure 1-3. Due to the heavy-tailed distribution, it is most likely that a shorter path length is chosen, but occasionally a longer path will be chosen (from the extended tail). This type of distribution is also often referred to as a power-law distribution. The term Lévy walk, in contrast to Lévy Flight, is also used which indicates that steps are taken in more than one direction, isotropically. For this work these terms will be used interchangeably.

Figure 1-2: E. Coli in both movement phases. In the swim phase when the food gradient is strengthening, flagella are rotated counter-clockwise, which forms them into a single rotating bundle. This allows for straight-line motion. In the tumble phase, when the food gradient is weakening, flagella are rotated clockwise, randomly realigning the bacterium.
For a special case of Lévy Flight, a specific type of random walk called Brownian motion is realized (Lévy Flight is a generalized form of Brownian motion). Brownian motion has been used to model and explain the stochastic motion of small particles suspended in a fluid. The movement of the particles is the aggregate result of collisions with the fluid’s atoms and molecules. Mathematically, Brownian motion is described by the more general Wiener process.

Some work has suggested that a Lévy flight visits more distinct sites than Brownian motion in a certain amount of time, but only by a small margin [12]. However, when applied using a swarm, instead of a single agent, this difference can be amplified.

1.2.4 Lévy Foraging Hypothesis

A hypothesis within the field of biology has been that natural selection would have resulted in the survival of species with the most efficient foraging patterns. If Lévy flight
is the most efficient foraging technique, Lévy flight patterns should be able to be observed in nature [14]. More specifically, it is suggested that behavior transitions from Brownian motion to Lévy flight as food becomes scarcer. For this reason, much research has been concentrated on observing natural foraging patterns and assessing if they correspond well to Lévy flight. This was first done for albatrosses [12], and later done for marine predators [15], human hunter-gatherers [16], and even in ancient animals as evidenced by their fossil trails [17]. Work is continuing in identifying and developing tools to statistically match real movement data with Lévy models to test this theory [18][19][20]. As the analogy between search and area coverage has been established, these nature-inspired foraging techniques, proven over time to be efficient, can be applied to the area coverage problem.

1.3 Swarm Intelligence

The third critical topic area is swarm intelligence. The premise of swarm intelligence is to harness the efforts of individual members on a local level to produce some greater, emergent system-level behavior [21][22]. Swarm intelligence has been inspired by observations of nature, which presents a diverse portfolio of examples. The application of these techniques to robotics in known as swarm robotics [23].

The general topic of the formation of groups in nature has been investigated and suggested to enhance the detection of and response to important features in the environment, even in the face of uncertainty and distractions, by giving individuals access to higher-order, spatiotemporal information which allows individual cognitive limitations to be overcome [24]. Inspired by social insects, swarm intelligence algorithms are being applied to many scientific and engineering problems, and in addition to performing well in
static environments, have been shown to be both flexible and robust in a dynamic environment [25].

1.3.1 Return to Ant Model

The model of the ant colony, previously mentioned for its use of chemotaxis, elucidates the power of a swarm approach. Individual ants are neither particularly intelligent, nor influential, and yet can collectively forage for food, construct an elaborate nest, defend that nest from predators, and seamlessly switch back and forth between tasks. They do all this without any central leader directing their work. Examining the particular case of foraging for food, each ant follows only simple rules for behavior based on chemical pheromone concentrations. The emergent system-level behavior means that the colony exploits known food sources, simultaneously explores areas for new food sources, and can react to a changing, dynamic environment.

1.3.2 Other Examples of Swarm Intelligence in Nature

Other examples of a swarm approach in nature abound. Individual bees forage for nectar and pollen and perform a ‘waggle’ dance when they return to the hive. This dance communicates the direction and distance of the food source as well as the strength of the food source. Birds in flocks and fish in schools are not able to evaluate the movements of all members of the flock or school. Instead, they base their movements on only their immediate neighbors and the leading bird or fish. However, this leadership is only temporary and any member may perform the role of leader. The emergent group behavior based on only these local interactions can provide some benefit to the entire group. In the case of schools of fish, it allows them to more effectively detect and evade predators.
Observation of these occurrences of swarm intelligence in nature has led to the development of the next generation of optimization techniques such as ant colony optimization (ACO) [26], particle swarm optimization (PSO), Cuckoo Search [27], Flower Pollination Algorithm [28], and Firefly Algorithm [29]. These techniques have outperformed many of the more conventional optimization schemes and many of the most effective of these techniques, like Cuckoo Search, have utilized Lévy Flight.

1.3.3 Central vs. Decentral/Distributed Control

The most apparent difference between these swarm techniques and traditional centralized control is that swarm-based techniques are decentralized or distributed. This means that instead of one leader directing the individual actions of each member, each member itself decides on its own actions or works in conjunction with a local leader.

The terms decentral and distributed control are often used interchangeably, but they can be used to differentiate different modes of communication, as illustrated in Figure 1-4. Decentralized can be used to mean that although there is not a single, central leader, there

![Figure 1-4](image_url)

*Figure 1-4*: (Left) Centralized system—all communication between agents and central leader. (Middle) Decentralized system—all agents communicate with their respective decentralized leader and these leaders communicate among themselves. (Right) Distributed system—all agents communicate only locally with other agents.
are multiple local leaders, each responsible for a particular sub region of the overall region. Both centralized and decentralized can be understood as a multi-tiered system. In these types of systems, the leaders may have enhanced computational power, communication ability, or mobility, compared to the other basic units. Contrast the multi-tiered centralized and decentralized system with a single-tier distributed system. In a distributed system, only local interactions take place and no leader, not even a decentralized one, is present.

1.3.4 Advantages of Swarm Intelligence Approach

The advantages of a swarm approach, whether in the form of a decentralized or distributed system, in comparison to a traditional centralized approach, are scalability, lowered sophistication, and robustness. In a swarm, new members can easily be integrated into the group, enabling massive scalability. In a centralized approach, only a finite number of members could be added due to the limited processing and communication ability of the leader. Likewise, in a swarm individual members follow only simple rules which demand less sophistication and therefore, less cost. Finally, in the centralized approach, if the leader were to fail, the entire group would also fail. A swarm is resilient to the loss of one, or many, of its members and can continue to operate as it did prior to the failure.

1.4 Combining Area Coverage, Biased Walks, and Swarm Intelligence

The rationale for combining these three ideas is easy to convey. Better area coverage, measured through different metrics, can be more easily achieved using multiple agents rather than a single agent. To coordinate these multiple agents, swarm intelligence has been shown to have some advantages over traditional centralized approaches. Lastly, swarm intelligence uses simple, local control laws for which chemotactic, biased random walks are excellent candidates.
1.5 Literature Review and Research Gaps

1.5.1 Pheromone-based Swarm Research

There exists a rich collection of research dedicated to studying chemotactic, pheromone-based phenomena in nature. Much work has been done in exploring different methods of modeling ant systems and characterizing different aspects of ant’s behavior. The dynamic response to changes in an environment has been studied and some ant species have been shown to always choose the higher quality food source, despite existing trails, whereas other ant species behavior is more dependent on existing trails and less dependent on the relative strength of the food sources [30]. Later, it was shown how recruiting behavior and trail formation could be explained exclusively using the quality of the food source, for one of the species studied [31]. In another work, the behavior of a colony was compared for the case of having two equal food sources, as well as two unequal food sources, which could be generalized to model different species through changing model parameters [32]. It was also shown that the recruitment behavior, characterized by the number of scouts returning to recruit and the number of ants recruited by the scout, were related to both the type and size of food source that has been discovered [33].

Beyond foraging, pheromone trails have been shown to be created not only to aid recruitment once a food source has been discovered, but also to assist in exploration of an initially pheromone-free region, which illustrates another facet of ants’ self-organization capabilities [34]. Pheromone trails have also been shown to not only help direct ants to the highest quality food source, but also to regulate overcrowding by dynamically creating parallel trails, from which important lessons can be applied to designing transportation systems [35]. Furthermore, ants show some interesting navigational behaviors in finding
their way back to their nest, even without pheromone, as they are theorized not to use an exact vector summation method (as human navigators would do), but rather an approximate method where they remember the angle of turns they’ve made, and weighing the distance they’ve traveled for each segment, calculate a heading back to their nest [36].

The transition between disorganized foraging and pheromone-based, organized foraging behavior, described as a phase transition, has been studied and related to the ant population size for some species of ants, which supports the idea that a minimum number of members is needed for these societies to function effectively [37]. Refinements to the typical reaction diffusion model have been made that differentiate between ants searching for food and ants ferrying food, which have helped explain ants’ ability to react to a changing environment [38]. Another work exploring this transition takes the unique approach of characterizing the nest-bound and food-bound ants as waves with different phase velocities which show little correlation to ant speed and density [39].

A pheromone can be generally described as a diffusible substance that deteriorates over time. In more general work, active ‘walkers’ moving in response to a potential field which could alter their local potential by releasing this type of diffusible substance (comparable to a pheromone) were shown to exhibit clustering behavior [40]. The chemotactic sensitivity to this type of substance, has been related to a system ‘mass’, which influences the stability of the system as modeled by a typical Keller-Segel reaction diffusion system [41]. In more depth, chemotactic sensitivity and its effect on the formation or non-formation of pheromone trails has also been studied [42].

Some work has also focused on how these same chemotactic interactions could be emulated using robotics. Work has been performed to develop a control law that could be
applied to robots to emulate ant foraging behavior [43] and a completely different approach has used robots which have been programmed to forage for ‘food’ using a measure of colony ‘energy’ to drive decision making [44].

Now looking specifically at research that has been performed on using a stigmergic form of communication with virtual pheromones to enact different types of behavior in robots, Kuiper provided area coverage using a pheromone based model, but did not use evaporation or diffusion and simulated the system using a discrete grid where agents were only allowed to move into three of the eight neighboring grid cells that were directly in front of the agent [1]. Sauter did use evaporation and diffusion to provide surveillance and patrol, among other behaviors [5]. However, they did not investigate the effects of varying the diffusion and evaporation. Brueckner used a pheromone-based model to direct team behavior for search, imaging, and maintenance tasks [45]. Gaudiano provided area coverage using a pheromone-based model and investigates how adding additional agents enhanced performance, but there is no mention of evaporation or diffusion of pheromone [6]. Ranjbar-Sahraei used pheromone in a novel way to tessellate an area into individual circles of responsibility [7]. Here agents reduced the radius of their own circle when they detect pheromone outside of the circle and increase the radius of their circle when they detect pheromone inside the circle.

Although evaporation and diffusion had been used in the course of this existing research, no systematic investigation had been made into the relative role and influence of each and how they affect area coverage performance. This research gap was addressed in this work by performing a broad survey of different evaporation and diffusion parameters and studying the resultant performance.
The addition of noise, leading to a combined deterministic and stochastic system, has been shown to improve decision making in a dynamic environment for some species of ant [46] and Ramakrishnan used a pheromone-based model to replicate ant foraging behavior and investigated the role of noise on the performance of the system [43]. This type of investigation has not been performed for area coverage. In addition to evaporation and diffusion, different noise parameters were also investigated. An interesting line of research performed by Parunak examined how this addition of randomness, or noise, at a local level, can actually have the effect of creating self-organization on the system level [47].

1.5.2 Other Swarm Research

Smith investigated using a biased and unbiased random walk to provide area coverage where the walk could be biased to turn or biased to continue without turning [4]. Cheng studied the effects of limited communication and the possibility of faulty agents in a swarm [2]. Cooperative box-pushing by robots was achieved using only a combination of sensors, actuators, and a pre-programmed, prioritized list of behaviors [48]. Marjovi has studied how maps generated by multiple agents can be compiled using only local communication [49]. Indeed, another benefit of using multiple agents is that they can use collaborative localization to help improve their own position estimate as well as form a more accurate map of their environment [50][51]. This is a technique where each robot incorporates its neighbor’s measures of its own position as well as its own position estimate to form a better positional estimate.
1.5.3 Lévy Flight Swarm Research

Only a limited number of studies have looked at using Lévy flight in conjunction with swarms and to the author’s knowledge, none have used it for area coverage. Sutantyo found that using a step length from a Lévy distribution was more effective for search, but the improvement became less significant as more agents were added [10]. Nurzaman compared Lévy flight to gradient following for search [8]. It was found that a hybrid approach that used Lévy flight only when the gradient is small was the most effective. Scheutz used an approach where each agent is equipped with both an attractive and repulsive beacon which allows for continuous adaptive coverage to protect a target [9].

1.5.4 Other Multi-Agent Area Coverage Approaches

Many other approaches to providing area coverage have been proposed and several comprehensive literature reviews of this expansive topic have been performed [52][53].

In frontier-based coverage, the search area is seen as having a known area and unknown area. The boundary between these two areas is called the frontier, and this approach looks at different ways to shift the frontier from known space to unknown space [3][54]. A similar strategy, known as flooding, uses a depth first approach to explore an area [55].

Spanning trees, which connect all vertices of an undirected graph, are another proposed way to provide area coverage. Once these trees are formed, exhaustive coverage is provided when an agent navigates around the perimeter of these trees. Some authors have explored how to tessellate a space into multiple spanning trees to provide coverage using multiple robots [56][57][58]. An alternative strategy is to create a single spanning tree, and
allow multiple agents to follow it regular intervals to achieve a constant visitation frequency [59]. Other authors have worked to find algorithms that find optimal multiple spanning trees in polynomial time [60].

Another proposed method to provide area coverage is to tessellate an area and use a genetic algorithm to find optimal paths [61]. This method has been specifically proposed to help route microbots capable of delivering medication inside the human body [62]. Another method of tessellating a space into areas of responsibility uses the idea of Voronoi diagrams. Once tessellated, each agent would be responsible for covering their individual area [63][64]. The portion of space that is closer to an individual agent than all other agents would be its area of responsibility. A power diagram, which is a generalized version of a Voronoi diagram, has also been used to tessellate a space into areas of responsibility to route multiple vehicles [65]. Likewise, a method called Boustrophedon decomposition tessellates the area into cells and visits each cell using a sequence of back and forth paths. This has also been applied in the multi-agent case [66].

Low investigated a strategy for providing wide-ranging area coverage while still sampling some discrete number of hotspots, using a dynamic programming approach [67].

1.6 Objective and Organization of Thesis

Now that sufficient background has been given and the context in existing literature has been presented, the first objective of this thesis is to explore the quantitative and qualitative contributions of noise, diffusion, and evaporation on area coverage performance. The second objective is to measure the utility of using Lévy flight for the purpose of area coverage.
Chapter 2 gives a mathematical formulation of the problem and appropriate performance metrics. Chapter 3 gives details on the implementation and simulation of the model. The results are given in Chapter 4, followed by a discussion of the results in Chapter 5, and the conclusion in Chapter 6.
Chapter 2

Problem Formulation

2.1 Defining a Mathematical Model

The approach in modeling this problem is taken after Keller and Segel’s model of Chemotaxis [68]. The impetus for this approach is so that this work can be compared with the substantial amount of existing research in the area of chemotaxis. In this way, an artificial system that uses robots and virtual pheromone is akin to a biological system. For example, the Keller-Segel (KS) model has been used for modeling e. coli and can accurately replicate a variety of their spatial patterns that have been observed in nature.

The classical form of the KS model is given in (1) and (2). Here (1) gives the change in cell concentration, $a(r,t)$, and (2) gives the change in chemical concentration, $b(r,t)$ in time, $t$, and in space, $r \in R^2$. Robotic agents will take the place of cells in the KS model.

\[
\frac{\partial a(r,t)}{\partial t} = \nabla \cdot (D_a \nabla a(r,t) - \chi a(r,t) \nabla b(r,t)) + f \tag{1}
\]

\[
\frac{\partial b(r,t)}{\partial t} = D_b \Delta b(r,t) + g - h \tag{2}
\]

The terms $D_a$ and $D_b$ are the diffusion coefficients of the cells and chemical, respectively, and $\chi$ is the chemotactic sensitivity of the cells. If the pheromone is attractive, $\chi$ will be negative, and if the pheromone is repulsive, $\chi$ will be positive. Cell death/division
is represented by $f$, the production rate of pheromone is given by $g$, and the evaporation rate of pheromone is given by $h$. As it stands, these equations are still complex and would be difficult to solve and simulate behavior. Some simplifying assumptions are made to reduce the model to a more workable form.

First, it is assumed that the number of cells (robotic agents) remain constant such that the $f$ term is removed. Next, individual cells undergo a combination of random motion and gradient-following chemotaxis. It is assumed that pheromone is generated at a constant rate so that $g$ is only a function of $a$, and that the evaporation rate of pheromone is linearly dependent on the concentration so that $h$ is only a function of $b$, now with evaporation rate $\gamma$. Lastly, assume that the attractant (or repulsant) diffuses passively over the field. These assumptions applied to (1) and (2), result in (3) and (4).

$$\frac{\partial a(r,t)}{\partial t} = \nabla \cdot \left( D_a \nabla a(r,t) - \chi a(r,t) \nabla b(r,t) \right)$$ \hspace{1cm} (3)

$$\frac{\partial b(r,t)}{\partial t} = \nabla \cdot D_b \nabla b(r,t) + g(a(r,t)) - \gamma(b(r,t))$$ \hspace{1cm} (4)

The system is now reduced to a form that will be easier to simulate. This form is sometimes referred to as the Keller Segel Minimal Model [68]. These equations are still a set of coupled non-linear partial differential equations, and can’t be solved analytically.

These equations represent the system as a continuum and form an analytical basis for this work. Due to the large quantity of cells in biological applications, it is sensible to model the cells as continuum, rather than discrete agents. However, in this case it is desired to describe the agents in the discrete form of a Langevin equation so that this could be used as a control law for each individual robot. The next steps will relay how this Langevin equation is formed and how it can be related to the existing continuous description.
Assume that for a simple kinematic model, the inertial component may be neglected so that the velocity is a function of only the gradient following component and a noise term, as given in (5).

\[ \dot{R}_a = v_a = \chi \nabla b(r, t)|_{R_a} + \sigma dW \]  

Here \( R \) denotes the position of agent \( a \), and \( v \) the velocity of agent \( a \). Equation (5) is a summation of the gradient following component and the noise term, given by \( \sigma \). This noise is a constant magnitude and the direction is sampled from a uniform distribution. This partial differential equation is in a form that could be easily implemented for an actual robot. Finally, use (6) to relate the continuum description given in (3) and the discrete description given in (5).

\[ a(r, t) = \sum_{a=1}^{A} \delta(r - R_a(t)) \]  

The number of agents is given by \( A \), and \( \delta \) is the Dirac delta function.

### 2.2 Defining Performance Metrics

Three different metrics of assessing area coverage performance are used. These metrics are chosen because they closely correspond to the demands of different types of area coverage applications, as outlined in Chapter 1.

#### 2.2.1 Metric 1-Percent Area Coverage Integral

The Percent Area Coverage Integral metric includes elements of the rate of area coverage, as well as the exhaustivity of area coverage. For this metric, the percent of area that has been visited at least once is plotted as a curve with respect to time. A typical curve, which is designated \( m(t) \), is shown in Figure 2-1. The slope of the curve indicates the rate
of area coverage at any time, while the percent of area covered at any time is a measure of
the exhaustivity of the coverage. A single performance value, designated as $P_1$, incorporates both of these elements by taking the integral of the curve from $t=0$ to $t_{\text{final}}$, as
given in (7). Thus $P_1$ can be understood as the area under curve $m(t)$ where $m(t)$, as defined
by (8), can be found by integrating over the entire search space and $\text{visited}(r,t)$ is one if that
area has been visited at least once, and zero otherwise.

$$P_1 = \int_0^{t_{\text{final}}} m(t) dt \quad (7)$$

$$m(t) = \int_R \text{visited}(r,t) dR \text{ where } \begin{cases} 
\text{visited}(r,t) = 1 \text{ if visited once} \\
\text{visited}(r,t) = 0 \text{ if never visited}
\end{cases} \quad (8)$$

**Figure 2-1:** Typical curve of fraction of area covered over time. Area coverage
performance is quantified as the area under the curve.
As a performance measure, $P_1$ has some desirable characteristics. First, it provides a way of quantifying a performance curve, $m(t)$, and representing the curve in a single value. It is much easier to objectively compare scalar values from multiple runs than to compare the performance curves. Secondly, it uses percent of area covered as an input instead of the absolute value of area covered. This effectively normalizes the measure so that it is easier to compare swarm performance for larger or smaller search areas.

To achieve a high score, at the beginning a swarm must quickly cover new area without much coverage redundancy. In the latter stages, it must be able to find the continuously-shrinking portion of the environment that has never been covered.

For an environment that has been discretized, as will be the case for these simulations, (7) can be discretized as a summation from $t=0$ to $t_{final}$, as given in (9). Now $m_t$ is a discrete value at time $t$, given in (10) and found by summing over all the cells, where $visited_{ij}$ is one if that cell has been visited once, and zero otherwise.

\[
P_1 = \sum_{t=0}^{t_{final}} m_t \tag{9}
\]

\[
m_t = \sum_i \sum_j visited_{ij} \text{ where } \begin{cases} visited_{ij} = 1 & \text{if visited once} \\ visited_{ij} = 0 & \text{if never visited} \end{cases} \tag{10}
\]

### 2.2.2 Metric 2-Visit Entropy

Visit entropy is the second performance metric, which is a measure of the frequency of coverage. Entropy is a term used to mean the disorder of a system. In an ordered area coverage, each area of the search space would be visited with the same frequency. In a less ordered coverage, some areas may never be visited, while other areas are visited very frequently. Unless there is a driving factor for favoring one area over another, it is desirable
that visit frequency be similar across the entire search space and a lower entropy (meaning more ordered) is preferred.

Visit frequency, $\varphi(r, t)$, can be calculated as the number of times a location has been visited, $n_{visit}$, divided by the elapsed time, $t$, as given in (11). In general form, entropy can be calculated as given in (12).

$$\varphi(r, t) = \frac{n_{visit}}{t} \quad (11)$$

$$H(\varphi(r, t)) = - \int_k \text{Prob}(\varphi_k(t)) \log \text{Prob}(\varphi_k(t)) \, dk \quad (12)$$

Here $H$ is the differential entropy, which is an extension of entropy to continuous probability distributions, and $\varphi_k(t)$ is the $k$th visitation frequency at time $t$. Typically, a log base of 2 is used and $\text{Prob}(\varphi_k(t))$ is the probability of being in an area visited with $k$th frequency which can be calculated as given in (13).

$$\text{Prob}(\varphi_k(t)) = \frac{\int_r A_{\varphi_k(r, t)} \, dR}{A_{\text{total}}} \quad (13)$$

where

$$A_{\varphi_k(r, t)} = 1 \text{ if visited with frequency } \varphi_k$$

$$A_{\varphi_k(r, t)} = 0 \text{ if not visited with frequency } \varphi_k$$

$A_{\varphi_k(r, t)}$ is one where visited with $\varphi_k$ frequency, and zero otherwise, and $A_{\text{total}}$ is the total area. Because frequency is not a continuous value, as is the case in some entropy calculations, (12) may be simplified to the form given in (14). For these simulations which were performed using a discrete grid, (13) can be simplified to (15) where $A_{\varphi_{ijk}(r, t)}$ is one if that cell has been visited with $\varphi_k$ frequency, and zero otherwise.

$$H(\varphi(r, t)) = - \sum_k \text{Prob}(\varphi_k(t)) \log \text{Prob}(\varphi_k(t)) \quad (14)$$

$$\text{Prob}(\varphi_k(t)) = \frac{\sum_i A_{\varphi_{ijk}(t)}}{A_{\text{total}}} \quad (15)$$

where

$$A_{\varphi_{ijk}(r, t)} = 1 \text{ if visited at } \varphi_k$$

$$A_{\varphi_{ijk}(r, t)} = 0 \text{ if not visited at } \varphi_k$$

To give a representation of entropy of the system over the course of the entire simulation, similar to what was done for the first performance metric, the entropy from (14)
will be summed from $t=0$ to $t_{\text{final}}$, as given in (16). This is designated as $P_2$ as the second performance measure.

$$P_2 = \sum_{t=0}^{t_{\text{final}}} H(\varphi(r, t))$$

(16)

In another context, entropy is used as a measure of the dispersion of a distribution, and it is often used as an alternative or complement to standard deviation for this purpose. However, the two do not always behave in a similar way and it is useful to point out some of the differences. A good example of a case when the two would generate different results is a bimodal distribution. If both peaks were moved farther apart from one another, the standard deviation would increase while the entropy would remain the same. Another example is when $X$ is allowed to be a discrete random variable taking two values (-$c$, $c$) with equal probability. As $c$ increases, the standard deviation of the distribution increases but the entropy remains the same. A further example is when a set of values is multiplied by a constant. In this case the standard deviation will change, but the entropy will remain the same. For these reasons, entropy is taken to be a more robust and more indicative measure when the primary interest is in the relative disorder (or disorder) of a system.

Note that although the average visit frequency, $\bar{\varphi}$, would seem to be a useful measure of visit frequency, $\bar{\varphi}$ is only a function of the number of agents, $A$, the number of tessellations, $T$, and the average areas visited per unit time, $\rho$, as derived in (17) and (18). Here $n_{i,\text{visit}}$ is the number of times the $i$th location has been visited, and $t$ is the elapsed time. This means that it is invariant for most of the parameters that are being tested and not actually a good indication of visitation frequency.

$$\sum_{e=1}^{T} n_{p,\text{visit}} = A\rho t$$

(17)
\[ \bar{\varphi} = \frac{\sum_{e=1}^{T} \varphi_e}{T} = \frac{\sum_{e=1}^{T} n_{e,\text{visit}}}{Tt} = \frac{Ap_t}{Tt} = \frac{A\rho}{T} \]

2.2.3 Metric 3-Pop-Up Threat Detection

The third and final performance metric that is used is the percentage of pop-up threats detected, which is an indication of visitation frequency. Here pop-up threats, \( \mu \), are defined as temporary conditions that occur randomly throughout the search space and the percent detected, \( P_3 \), is calculated using (19). Here \( \mu_p \) is one if the \( p \)th threat was detected, and zero otherwise, and \( \mu_{\text{total}} \) is the total number of threats. A threat is considered to be detected if an agent comes within a certain sensing radius of the threat.

\[ P_3 = \frac{\sum_{p=1}^{\mu_{\text{total}}} \mu_p}{\mu_{\text{total}}} \text{ where } \begin{cases} \mu_p = 1 & \text{if detected} \\ \mu_p = 0 & \text{if not detected} \end{cases} \]

Like entropy, pop-up threat detection is indeed another indication of visitation frequency but it does provide more insight than entropy alone. For example, consider two scenarios where the entropy of the visit frequency would be the same for both. In the first half of the first scenario, agents favor the left side of the search space. About halfway through, the agents switch to favor the right side. In a second scenario the agents do not favor any particular area. It is possible that the two scenarios have the same entropy because the visit frequency of each area could be the same. However, if the pop-up threats occurred randomly in space, the threat of detection is different from one scenario to the other. Although this is an extreme example, it illustrates that the percentage of pop-up threats detected is not necessarily proportional to the disorder (or order) in the system.
2.2.4 Metric Shortcomings

The justification for each of these performance metrics was given, but for these metrics to be useful, their shortcomings must also be understood. One shortcoming is in the tessellation of a search space which will affect the percent area coverage integral and visit entropy metrics. It is assumed that once any part of an area is visited, the entire area can be considered to be visited. The validity of this assumption depends on the size of the tessellated area, and on the individual parameters of an application. Use the case of resource exploration as an example. If sampling for underground oil reserves, drilling a test well would provide a good indication about the presence of oil for a moderately sized area around the well. If seeking some mineral deposits that are highly localized, sampling in a particular area may not reveal any information about the presence of that mineral elsewhere.

The pop-up threat detection metric is also dependent on how it is implemented. Important parameters include sensor detection radius, duration of pop-up threats, and multiplicity of threats. Additionally, the pop-up threat may have the ability to sense if it would be immediately detected by a proximal robotic agent. In this case the threat could be endowed with the option of waiting until the robotic agent was out of sensor range or popping up in another location where an agent is not immediately present.
Chapter 3

Implementation

3.1 Summary of Cases

The three types of systems that were investigated are presented in Table 3.1.

Table 3.1: Overview of Cases being Examined

<table>
<thead>
<tr>
<th>Case</th>
<th>Random Walk Type</th>
<th>Step Size Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Biased</td>
<td>Constant</td>
<td>Gradient Following without Lévy Flight</td>
</tr>
<tr>
<td>2</td>
<td>Biased</td>
<td>Variable</td>
<td>Gradient Following with Lévy Flight</td>
</tr>
<tr>
<td>3</td>
<td>Unbiased</td>
<td>Variable</td>
<td>Pure Lévy Flight</td>
</tr>
</tbody>
</table>

For the biased walks, the mechanisms that generate and influence this bias are of interest. The gradient of the pheromone is what biases the motion, and the distribution of the pheromone is dependent on the evaporation and diffusion of the pheromone. The magnitude of the noise affects the influence of the bias.

For walks that used a variable step size, the mechanism for varying this step size is what is of interest. The individual details of the implementation of these cases are given below.
3.2 General Implementation Details

Simulations were conducted with 10 agents randomly initialized inside a 100 by 100 unit area with a simulation time of 3000 seconds. Agents were taken to move at a constant rate of one unit per second.

3.2.1 Biased Walk Implementation Details

The model given by (3) and (4) was simulated using a discrete grid in Cartesian space for the pheromone, while the agents moved in continuous space. The discretization of the diffusion term was accomplished using a forward time center space (FTCS) approach with Dirichlet boundary conditions given by (20) and (21). The Dirichlet condition fixes the value of pheromone at the search space’s boundaries at zero.

\[ b_{ij}^{t+1} = \beta_1 (b_{i+1,j}^t + b_{i-1,j}^t) + \beta_2 (b_{i,j+1}^t + b_{i,j-1}^t) + (1 - 2\beta_1 - 2\beta_2) b_{ij}^t \quad (20) \]

\[ \beta_1 = D_b \frac{\Delta t}{\Delta x^2}, \beta_2 = D_b \frac{\Delta t}{\Delta y^2} \quad (21) \]

Here \( b \) is now the discrete value of pheromone in a particular grid cell, where \( i \) and \( j \) are indices for cells in the \( x \) and \( y \) directions, respectively, and \( t \) is the time index. The grid sizes \( \Delta x \) and \( \Delta y \) specify the grid resolution in the \( x \) and \( y \) directions, respectively, and \( \Delta t \) denotes the time step size. The stability of the system is limited due to this FTCS scheme, which limits the size of the time step as given in (22).

\[ \Delta t \leq \frac{\Delta x^2 \Delta y^2}{2D_b (\Delta x^2 + \Delta y^2)} \quad (22) \]

A grid size of 0.25 was used for \( x \) and \( y \), and a time step of one second was used for initial simulations, which along with the values for the rate of diffusion of less than 0.15, meet the conditions for stability. This coarser time step was used for a broad survey.
of different noise, evaporation, and diffusion parameters but a finer time step of 0.1 s was used for more focused subsequent simulations, as explained below. Deposited pheromone was applied equally in all grid cells which were passed through during a particular time step. Thus if a time step of 0.1 seconds was used, and two cells were passed through during that time step, with one unit of pheromone deposited per second, each of these cells would receive 0.05 pheromone.

As given in (5), the desired velocity is a vector sum of the gradient following and noise components. The magnitude of this velocity was restricted to a constant value of one unit per second. To achieve this, the gradient following and noise components were summed and then normalized. Pheromone was deposited at a rate of one unit per second. Pheromone was chosen to be repulsive, so a value of one was assigned to $\chi$. This is intuitive because an agent depositing pheromone can mark where they have already searched and it is reasonable to want to avoid that area so that other areas can be explored. In contrast to moving toward areas of higher concentration, which is called positive chemotaxis, this behavior is called negative chemotaxis.

A typical pheromone map after some time has passed is shown together with the history of the positions of the agents in Figure 3-1. This helps to demonstrate the effects of evaporation and diffusion. Only the most recent portion of the pheromone trail is visible because the older portion of the trail has evaporated. Likewise, the width of portions of the trail indicates how diffusion has helped to spread the pheromone.

To facilitate a generalized comparison, no collision detection or avoidance was implemented.
The agents were set to a fixed velocity and the agents’ heading was evaluated once per second. Thus a cumulative path was formed of individual, one-unit-long segments. Note that although the segment length is constant, the path direction does vary with time. For this case of a constant path length, agents were allowed to move outside the search area. However, once outside they would cease to follow the control law given in (5) and would simply move back toward the center of the search area, until they were once again within the search boundaries.

To achieve a variable path length, the agents were still constrained to the same maximum velocity, but now the path length was chosen by sampling from a heavy-tailed power-law distribution. A representative distribution is shown in Figure 3-2. This type of distribution has a peak near a value of one, and a long, extended tail. The implication of sampling the path length from this type of distribution is that it is most probable that a
shorter path (~one unit long), is chosen. However, with less frequency a relative long path length is chosen from the extended tail of the distribution.

To sample from this distribution, first a uniformly distributed random variable, $u$, was generated in the range $[0, 1]$. This type of variable can be transformed into another distribution by using the inverse cumulative distribution function. For a cumulative distribution function, $Q$, which corresponds to a probability density $q$, (23) gives the distribution according to $Q$.

$$s = Q^{-1}(u) \quad (23)$$

The cumulative distribution function for the power-law distribution that was used is given by (24). Here $s_{\text{min}}$ is the minimum value that $x$ is allowed to take, which was set to a value of one. Then, the inverse of this cumulative distribution function is given by (25). The only parameter for this type of distribution to be varied is $\alpha$, which ranges from one to three. Increasing alpha has the effect of lowering the peak value and increasing the tail of the distribution, as shown in Figure 3-2.

**Figure 3-2:** (Left) Distribution generated using an alpha value of 2.0 and 1000 path lengths. (Right) Effect of various values of alpha on the shape of the power-law distribution. For lower values of alpha, a higher peak at one is observed. For higher values of alpha, a longer tail is observed.
\[ Q(s) = 1 - \left( \frac{s}{s_{\text{min}}} \right)^{-\alpha} \]  
\[ Q^{-1}(u) = s_{\text{min}}(1 - u)^{-1/\alpha} = (1 - u)^{-1/\alpha} \]

Now instead of evaluating the agents’ heading once every second, the heading was evaluated and the control law applied after each agent completed its current path, whether that path be relatively short or relatively long. Calling back to (5), for a constant-length path the new position will be given by (26), and for a variable-length path, the new position will be given by (27), where \( \|\hat{R}_a\| \) denotes the magnitude of the velocity vector.

\[ R_{a,\text{new}} = R_{a,\text{previous}} + \frac{\hat{R}_a}{\|\hat{R}_a\|} \]  
\[ R_{a,\text{new}} = R_{a,\text{previous}} + (1 - u)^{-1/\alpha} \frac{\hat{R}_a}{\|\hat{R}_a\|} \]

To illustrate the difference between these two different schemes, Figure 3-3 shows a path generated with a constant path length and a path generated with a variable path length. The difference in the two approaches is further amplified when implemented using multiple agents. Figure 3-4 shows the resultant path after the same amount of time as Figure, but using 100 agents.

Some other modifications where made to implement this variable path length. Recall that for the previous simulations with a constant path length, agents were able to move outside of the search area, and once outside the search area, ceased to follow the control law and simply moved toward the center of the search space until they were again within its boundaries. Using a variable path and now given the possibility of a very long...
Figure 3-3: (Left) Path generated using a variable path length with an initial position of (0,0), a fixed velocity of 1 unit/second, and an alpha value of 2.0 after 1000 seconds. (Right) Path generated using a constant path length with an initial position of (0,0), and a fixed velocity of 1 unit/second after 1000 seconds.

Figure 3-4: (Upper Left) Path generated using same parameters as for Figure (Left), but with 100 agents starting from (0,0). (Upper Right) Path generated using same parameters as for Figure (Right), but with 100 agents starting from (0,0). (Lower Left and Lower Right) Zoomed version of central portion of paths from Upper Left and Right, respectively, with common scale.
path, it was desired to prevent the agents from moving far outside the search space. Thus, if an agent was poised to exceed these boundaries in the next time step, a new heading and path length were chosen that would keep the agent inside the search space. This change was implemented for all cases, even for the constant path length case, to allow for a fair comparison.

### 3.3 Performance Metric Implementation

To implement the three performance metrics, a new grid was defined with grid resolution of one unit in the x and y directions. This is in addition to the grid used to discretize the pheromone field and the two separate grids were used so that independent grid resolutions could be chosen for each. As described earlier, once any part of a grid square is visited, it is assumed that the entire grid square can be considered to be visited. For each grid square, the quantity of visits was tracked continuously during the simulation.

Pop-up threats were set to occur one at a time, for a duration of thirty seconds, and were considered to be detected if an agent came within five units of the threat. Thus for a simulation time of 3000 seconds, one hundred threats would pop-up. Two types of pop-up threats were tested. The first type of threat, designated as a simple threat, popped up in a random location without any regard for the position of the agents. A second type of threat, designated as a smart threat, would not pop-up if it was inside the detection range of an agent. Instead, it would pop-up in another location. Both types of pop-ups were implemented within a single simulation to help eliminate any bias from agent starting locations.
3.4 Implementation Sequence

3.4.1 Comprehensive Survey of Evaporation, Brownian Diffusion, and Noise

The first series of simulations were done for the first case of biased walk with a constant path length (Case 1). This would serve as a baseline for further simulations and help to gain a broad understanding of how noise, diffusion, and evaporation affected such a system. A range of values of noise magnitude, rate of evaporation, and rate of diffusion were tested. A summary of the values tested for these variables is given in Table 3.2. All possible combinations of these variables were tested. This means that 175 different combinations of conditions were simulated. In addition to these runs which included both diffusion and evaporation (Case 1a), additional runs were performed without evaporation (Case 1b), and without diffusion (Case 1c) respectively. These additional configurations resulted in an additional 70 combinations of parameters. Thus 245 total combinations were simulated. Due to the vast number of combinations, and the need to run each combination multiple times to ascertain mean performance, a relatively coarse time step of one second was used.

<table>
<thead>
<tr>
<th>Case</th>
<th>Random Walk Type</th>
<th>Step Size Type</th>
<th>Noise Magnitude $\sigma$</th>
<th>Rate of Diffusion $D_b$</th>
<th>Rate of Evaporation $\gamma$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1a</td>
<td>Diffusion + Evaporation</td>
<td>Biased</td>
<td>Constant</td>
<td>0.01,0.05,0.1, 0.2,0.3,0.4,0.5</td>
<td>1E-2,1E-3, 1E-4,1E-6</td>
</tr>
<tr>
<td>1b</td>
<td>Diffusion Only</td>
<td>Biased</td>
<td>Constant</td>
<td>0.01,0.05,0.1, 0.2,0.3,0.4,0.5</td>
<td>1E-2,1E-3, 1E-4,1E-6</td>
</tr>
<tr>
<td>1c</td>
<td>Evaporation Only</td>
<td>Biased</td>
<td>Constant</td>
<td>0.01,0.05,0.1, 0.2,0.3,0.4,0.5</td>
<td>N/A</td>
</tr>
</tbody>
</table>
3.4.2 Introduction of Lévy Flight and Comparison of Cases

The second round of simulations now compared performance for the three cases of biased walk with constant path length (Case 1), biased walk with variable path length (Case 2), and unbiased walk with variable path length (Case 3). The details of each case are given below and are summarized in Table 3.3.

**Biased Walk with Constant Path Length** (Case 1) - The same type of simulation was performed as before, but now only the best performing values of noise magnitude, and rate of diffusion were used. To ascertain if evaporation had a significant effect on the results, runs were done both with evaporation (Case 1a), as well as without evaporation (Case 1b). In addition, a much smaller time step of 0.1 seconds was used. Because the path length is constant, the alpha parameter is not applicable to this case. This case can be described as gradient following without Lévy Flight.

**Biased Walk with Variable Path Length** (Case 2) - The best performing values of noise magnitude, and rate of diffusion were used. As before, to ascertain if evaporation had a significant effect on the results, multiple runs were performed with evaporation (Case 2a), as well as without evaporation (Case 2b). The value of alpha, which changes the shape of the power-law distribution, was varied from one to three, in increments of 0.5. This case is a combination of Lévy Flight and gradient following.

**Unbiased Walk with Variable Path Length** (Case 3) - The value of alpha was varied from one to three, in increments of 0.5. Note that noise, evaporation, and diffusion are no longer applicable to this case because it is an unbiased random walk. In other words, pheromone will have no influence and instead of a combination of gradient following and noise, only the noise influences the movement. This case is essentially pure Lévy Flight.
Table 3.3: Overview of Cases and Parameters to be Tested (2nd Round)

<table>
<thead>
<tr>
<th>Case</th>
<th>Random Walk Type</th>
<th>Step Size Type</th>
<th>Noise Magnitude $\sigma$</th>
<th>Rate of Diffusion $D_b$</th>
<th>Rate of Evaporation $\gamma$</th>
<th>Alpha</th>
</tr>
</thead>
<tbody>
<tr>
<td>1a</td>
<td>Diffusion + Evaporation</td>
<td>Biased</td>
<td>Constant</td>
<td>Fixed</td>
<td>Fixed</td>
<td>Fixed</td>
</tr>
<tr>
<td>1b</td>
<td>Diffusion Only</td>
<td>Biased</td>
<td>Constant</td>
<td>Fixed</td>
<td>Fixed</td>
<td>N/A</td>
</tr>
<tr>
<td>2a</td>
<td>Diffusion + Evaporation</td>
<td>Biased</td>
<td>Variable</td>
<td>Fixed</td>
<td>Fixed</td>
<td>Fixed</td>
</tr>
<tr>
<td>2b</td>
<td>Diffusion Only</td>
<td>Biased</td>
<td>Variable</td>
<td>Fixed</td>
<td>Fixed</td>
<td>N/A</td>
</tr>
<tr>
<td>3</td>
<td>N/A</td>
<td>Unbiased</td>
<td>Variable</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

3.5 Implementation in MATLAB

This program was implemented using MATLAB and the code is included in Appendix A. The most computationally expensive portion of the program is the FTCS scheme for pheromone diffusion which must loop through each cell. For this reason, Case 3, for which diffusion is not applicable, runs much more quickly. The largest factors which drive computation time are the time step, and the resolution of the pheromone grid. For reference, for a time step of 0.1 seconds, and pheromone grid resolution of 0.25, Case 1, 2, and 3 require 280, 261, and 5 seconds/run, respectively, without any graphical output. When the time step is backed off to 1 second, as was the case for the survey, Case 1 requires 29 seconds/run. These times are for MATLAB R2015b on a Windows 8 PC with Intel Core i7-4720HQ Processor, 16GB RAM, and a SSD.
Chapter 4

Results

4.1 Survey of Evaporation, Diffusion, and Noise

It is instructive to first examine the interactions between diffusion, evaporation, and noise simultaneously, and then to perform a more detailed inspection. Please refer back to Table 3.2 as needed for an overview of cases tested in this round of simulations and note that all results being presented were the mean over twenty-five runs. Performance for the case of diffusion and evaporation (Case 1a) is given in Figure 4-1. This shows how performance varied with both evaporation rate and diffusion rate for various noise values. Maximum performance was realized at a noise value of 0.05, rate of evaporation of 0.0001, and rate of diffusion of 0.0001. It can also be said that of all the noise values, for 0.05 noise, near maximum performance was achieved for any combination of evaporation and diffusion. The same is not true when choosing any single value of evaporation or diffusion where there are at least some combinations of the other two parameters that result in poor performance.

Figure 4-1 is instructive for observing some global trends, namely the sensitivity to noise, but it is difficult to make detailed observations. For a deeper analysis, Figure 4-2-Figure 4-5 present the results in a more detailed, methodical form.
Figure 4-1: Percent Area Coverage Integral for specific values of noise over a range of both evaporation and diffusion.

Figure and Figure detail how performance changed for a range of diffusion rates and present the same information in two different ways to aid analysis. Figure shows trends for specific evaporation rates, with different noise values in separate frames whereas Figure shows trends for specific noise values, with different evaporation rates in separate frames. These figures include data from the case of diffusion with evaporation (Case 1a), as well as diffusion only (Case 1b). From both figures, it is apparent that performance was maximum for a noise of 0.05, 0.1, and 0.2. For many of the better performing cases, and especially with weaker evaporation or no evaporation, performance peaked at a medium diffusion rate. Figure shows that without evaporation, the performance was still maximum, or near maximum, for each value of noise. Figure shows that the faster rates of evaporation consistently performed worse than slower rates of evaporation. In comparison to other values of noise, 0.01 had a much larger variation in performance.
Figure and Figure detail how performance changed for a range of evaporation rates, and as before, present the same information in two different ways to aid analysis. Figure shows trends for specific diffusion rates, with different noise values in separate frames whereas Figure shows trends for specific noise values, with different diffusion rates in separate frames. These include data from the case of diffusion and evaporation (Case 1a), as well as evaporation only (Case 1c). From Figure, for all noise values, not using diffusion became less and less competitive as evaporation became weaker. Noise values of 0.05 and 0.1 were the best performing and, as before, a noise value of 0.01 showed the most variation. From this figure, maximum performance for each evaporation rate was very comparable. From Figure, especially for strong noise values, strengthening evaporation decreased performance.

Post simulation statistical analysis was performed using ANOVA. It was confirmed that, with a 95% confidence level, the main effects of noise, diffusion, and evaporation on performance was statistically significant. In addition, all two-factor and three-factor cross-interactions between noise, diffusion, and evaporation were also statistically significant.
Figure 4-2: Percent area coverage integral for specific values of evaporation for a range of diffusion rates. Stronger evaporation is denoted by darker lines and the case of no evaporation is shown with a dashed line. Results for increasing values of noise are shown frame by frame from left to right.
Figure 4-3 Percent area coverage integral for specific values of noise for a range of diffusion rates. Weaker noise is denoted by darker lines. Results for increasing values of evaporation are shown frame by frame from left to right.
Figure 4-4: Percent area coverage integral for specific values of diffusion for a range of evaporation rates. Stronger diffusion is denoted by darker lines and the case of no diffusion is shown with a dashed line. Results for increasing values of noise are shown frame by frame from left to right.
Figure 4-5: Percent area coverage integral for specific values of diffusion for a range of noise values. Weaker noise is denoted by darker lines. Results for increasing values of diffusion are shown frame by frame from left to right.
4.2 Comparison of Three Cases

As before, all results presented were the mean over twenty-five runs. For all three metrics, three cases are shown. These cases are: gradient following without Lévy (Case 1a), gradient following with Lévy (Case 2a), and pure Lévy flight (Case 3). The gradient following control schemes used the best performing combination of values for noise, rate of diffusion, and rate of evaporation. Additionally, gradient following without Lévy and gradient following with Lévy were tested without evaporation (Cases 1b and 2b, respectively). Please refer back as needed to Table 3.3 for an overview of all cases tested in this round of simulations. The same values for noise and rate of diffusion were used for Case 1a and 1b, and for Case 2a and 2b. Note that because the alpha parameter is not applicable to Case 1, the performance is shown as constant for all alpha values. For all figures shown, the error bars designate the ninety fifth percent confidence interval.

Figure shows the first performance metric, percent area coverage integral, with the alpha parameter for the Lévy flight being varied. For all alpha values, gradient following performance was improved when using Lévy flight. Both for gradient following with Lévy (Case 2), and for pure Lévy flight (Case 3), performance deteriorated as the alpha parameter was increased, although this trend was much more significant for pure Lévy flight. The presence or absence of evaporation did not significantly affect gradient following performance.

Figure illustrates the second performance metric, entropy, as the alpha parameter varied. Recall that a lower entropic value is desirable because it implies a more ordered system. The pure Lévy flight (Case 3) had the highest entropy, followed by gradient
Figure 4-6: Percent Area Coverage Integral for all cases for a range of alpha values.

Figure 4-7: Entropy performance for all cases for a range of alpha values.
following with Lévy (Case 2), and finally, gradient following without Lévy (Case 1). This ranking was consistent for all values of alpha. This means that gradient following without Lévy flight is the most ordered control scheme, with respect to visitation frequency. As alpha increased, the entropy of the system decreased for pure Lévy flight whereas entropy peaked for gradient following with Lévy at an alpha value of 1.5. Again for this metric, the presence or absence of evaporation did not significantly affect the results.

Now compare all three cases using the final metric, which is pop-up threats detected. Figure and Figure show simple and smart threats detected, respectively, over a range of alpha parameter values. For both simple and smart threats, maximum performance was achieved using gradient following with Lévy, with an alpha value of 1.0 and 1.5, respectively. For simple threats, gradient following with Lévy was the best for all values of alpha, but for smart threats, performance decreased as the alpha parameter increased. There was no consistent trend of better or worse performance when using or not using evaporation; evaporation was shown to help for some values of alpha. Evaporation was consistently helpful for detecting smart threats using gradient following with Lévy.
Figure 4-8: Percent of simple pop-up threats detected for all cases for a range of alpha values. Dashed lines indicate that no evaporation was used.

Figure 4-9: Percent of smart pop-up threats detected for all cases for a range of alpha values. Dashed lines indicate that no evaporation was used.
Chapter 5

Discussion

5.1 Survey of Evaporation, Diffusion and Noise

In the survey of evaporation, diffusion, and noise, the primary performance metric was the percent area coverage integral.

5.1.1 Impact of Noise

By and large, the most impactful parameter was the magnitude of noise. From Figure-Figure, it can be seen how much impact noise had on area coverage performance, independent of rate of diffusion and evaporation. It was also evident how too much or too little noise can negatively impact performance. For global maximum performance, a noise value of 0.05 was used and very competitive performance was observed for this noise value for any combination of diffusion and evaporation.

Understanding why too little noise is as bad as too much noise requires understanding the contribution of noise. Noise is helpful because it can help drive an agent to move through an area that has been covered to reach an area that needs to be covered. Without noise, an agent would not move through this already-covered area, and could in fact become trapped. However, too much noise also has a negative impact because it marginalizes the pheromone gradient that is present. As mentioned earlier, pheromone is
essentially information, and in most cases it is useful to heed this information when making
decisions about where to move.

5.1.2 Impact of Diffusion

The addition of some amount of diffusion can improve overall performance. For
many cases, including the best performing examples, it is clear from Figure and Figure
that maximum performance was achieved at a medium value of diffusion. Performance
also appears to be linked to rate of evaporation as diffusion does not appear to have as large
of an impact when evaporation was very high. This result is expected because if pheromone
is allowed to evaporate very quickly, pheromone cannot be diffused fast enough to have
any significant impact, regardless of diffusion rate.

It is intuitive that too much or too little diffusion could detrimentally affect
performance. If pheromone is not diffused enough, agents must get very close before being
able to sense any gradient. If they had detected pheromone earlier, they would also be more
likely to move away from the pheromone earlier, presumably in a more promising
direction. If pheromone is diffused too quickly however, not enough pheromone remains
at the original point of deposition, meaning an agent cannot actually differentiate between
covered and non-covered areas.

5.1.3 Impact of Evaporation

Unlike diffusion and noise, the addition of evaporation did not appear to
significantly improve performance from a global perspective. This is evident in Figure,
where performance appears to almost universally improve as evaporation decreases to zero.
This is expected when considering how this metric measures performance. Based on this
metric, revisiting a cell does not contribute to performance. Thus, any amount of
evaporation would make returning to a cell more likely, and could only decrease performance. The benefit of evaporation is therefore less clear unless performance is measured in a different way, whereby revisiting a cell may be valuable. An example of a scenario of this type would be detecting pop-up threats.

Even though from a global perspective the benefit of evaporation may not be clear, some limited cases are visible in Figure where performance indeed increased as evaporation increased. Thus, the influence of evaporation may not be as straightforward as initially described. One of the cases where evaporation seemed to help is when both diffusion and noise were low. In this case, diffusion does not quickly spread pheromone so it remains near to the original deposition point. Unless pheromone evaporates quickly enough, it becomes difficult to cross over an existing pheromone trail (which may lead to an unexplored area). Because the influence of noise is also low, crossing over an existing trail becomes even more unlikely. This is supported by the fact that when noise was stronger, it transitioned back to better performance with less evaporation.

Contrast this specific scenario with low diffusion and high evaporation. The pheromone still remains near the original deposition point but now it evaporates before it can be used to guide other agents.

5.1.4 General Survey Results

Noise, evaporation, and diffusion interact in a complex and elegant way. From these simple mechanisms, arises a self-organized system-level behavior. It is evident that performance was a function of all three parameters and that maximum performance cannot be realized without using an ideal combination of all three. It is also evident that for a given fixed value of one of these parameters, there was not a universal answer for the ideal values
of the other parameters. This survey is the first important contribution of this work as it quantified how noise, evaporation, and diffusion interact to affect area coverage behavior and performance, which to this point had not been studied in the context of area coverage.

The metric used to quantify performance during this survey was the percent area coverage integral. A typical percent area coverage curve is shown in Figure which illustrates the differences between the different stages of area coverage and discovery. This metric is most appropriate for assessing performance in the initial stages of area coverage, but its usefulness will decrease as time elapses. At some point, most of the portions of the area will have been covered at least once, and thus there will not be much differentiation between different control schemes. When this occurs, a performance metric that focuses on visitation frequency becomes more useful.

![Typical area coverage curve showing fraction of area covered along y-axis during different stages of area coverage](image)

Figure 5-1: Typical area coverage curve showing fraction of area covered along y-axis during different stages of area coverage
5.2 Comparison of Three Cases

Now when comparing the three cases of gradient following without Lévy (Case 1), gradient following with Lévy (Case 2), and pure Lévy flight (Case 3), all three performance metrics were employed. Please note that for Case 3 there is no Brownian motion involved, so the only source of noise in motion is Lévy flight.

5.2.1 Area Coverage

The results from Figure show that using a gradient following scheme with Lévy flight offered superior performance. The chance of choosing a longer segment path helps the agent to proceed further in a particular direction than if that agent had been strictly using smaller path lengths. This can be helpful because it could allow an agent to more easily pass through a visited area to a potentially uncovered area. It can also be helpful because when using many short segments, the agent is given many opportunities to divert from its current heading. This can be detrimental if its initial heading, which was influenced by the contemporaneous gradient, is leading in a promising direction.

However, as the alpha value increased, and as the chance of choosing a longer path also increased, performance decreased. Hence, choosing longer path segments more frequently actually harmed performance and choosing a longer path sometimes results in revisiting a covered area. Also, a higher alpha value resulted not only in long paths being chosen more frequently, but the paths chosen were even longer than those for lower alpha values (see Figure). This is because the tail of the distribution becomes more extended as the alpha value increases. Thus, even when a longer-than-normal path may be advantageous, a much longer path may hurt performance. The best performance is obtained
using a balance of normal segment lengths, which leverages only local information, and occasional longer segment lengths.

In another sense, invoking Lévy flight can be seen as an infrequent call to ignore local information (pheromone gradient), with the underlying assumption that always following only local information is not optimal. Because the paths are longer, local information is effectively ignored because the agent does not stop as frequently to assess the gradient. The same observation can be made for the addition of a noise to a system. Both adding noise and varying the segment lengths decrease the influence of local information, and it has been shown that this can lead to improved global performance, when appropriately applied.

5.2.2 Visit Entropy

When measuring the entropy of visitation frequency, gradient following without Lévy was the best control scheme for all values of alpha, as shown in Figure. However, for high values of alpha, gradient following with Lévy was very competitive. In general, the fact that gradient following with Lévy had slightly more entropy than without Lévy is not surprising. Changing from constant length to variable length paths is essentially adding additional disorder to the system, which should result in some additional entropy.

It is interesting that entropy decreased (desirable) as the alpha parameter increased for gradient following with Lévy. A higher alpha value resulted in more frequent longer-path segments, and decreased performance as measured by the percent area coverage integral. Given that this is not the case for entropy, perhaps using more frequent long-path segments, although resulting in revisiting cells, also resulted in more cells being revisited.
at similar frequencies. This trend of decreasing entropy (desirable) was also observed in the case of pure Lévy flight, which supports this argument.

The use of pheromone, characterized by the gradient following schemes, is a tool for self-organization of the system. Qualitatively, it is easy to comprehend how adding pheromone allows for a more organized, and less entropic system. Pheromone is a way of embedding information about the environment and having information allows for better decisions. However, how to quantitatively measure the contribution of pheromone in ordering the system is not as obvious. This is one way in which the value of entropy as a performance metric is realized.

Another useful quality of the entropy metric is that it describes the variance of a distribution with a single quantity. One concern is that in doing so it may mask differences in the shape of the visitation frequency distribution, which could also indicate the desirability of a distribution. To ensure that the shapes of the distributions are accurately reflected in the entropy value, Figure 5-2 shows the visitation frequency and final entropy

![Figure 5-2: Distribution of visit frequencies for all cases. Frequency indicated by number of visits/total seconds](image)

<table>
<thead>
<tr>
<th>Visit Frequency</th>
<th>Total Cells</th>
</tr>
</thead>
<tbody>
<tr>
<td>0/3000</td>
<td>3000</td>
</tr>
<tr>
<td>1/3000</td>
<td>2500</td>
</tr>
<tr>
<td>2/3000</td>
<td>2000</td>
</tr>
<tr>
<td>3/3000</td>
<td>1500</td>
</tr>
<tr>
<td>4/3000</td>
<td>1000</td>
</tr>
<tr>
<td>5/3000</td>
<td>500</td>
</tr>
<tr>
<td>6/3000</td>
<td>0</td>
</tr>
</tbody>
</table>

Gradient Following without Levy, Entropy=2.73
Gradient Following w/Levy, Entropy=2.85
Pure Levy Flight, Entropy=3.43
(Levy Flight results truncated at 18/3000 for clarity, results extended to 37/3000)
value for a single instance of all three cases, each using its best performing values. The least desirable distribution created using pure Lévy flight had the highest entropy, and the most desirable distribution created using gradient following without Lévy had the lowest entropy. Here desirability is being judged subjectively, where it is generally more desirable to have the distribution concentrated at the lower frequencies (but of course not at a frequency of zero). Thus, the single entropy value does appear to accurately reflect the rank of entropy values. The distribution does give much more detail than just a single value, but it is also more complex to analyze and compare cases.

Now, compare the results of one instance of gradient following with Lévy for an alpha value of one and alpha value of three, as given by Figure 5-3. Again, it appears that the entropy value is a good indicator of the visitation frequency distribution. Although both runs have a similar number of unexplored cells, using an alpha value of three resulted in fewer cells being visited at a relatively high frequency (which would be to the detriment to visitation frequency of other cells).

![Figure 5-3: Distribution of visit frequencies for two cases of gradient following with different alpha values. Frequency indicated by number of visits/total seconds.](image-url)
Together, these results help to justify the use of entropy as a good, single-value indicator of visitation frequency.

### 5.2.3 Pop-Up Threat Detection

One surprising result is that ‘smart’ threats were detected at a very similar rate as simple threats. This can be partially explained by the relative sparseness of the threats. Consider that each agent has a detection radius of 5 units. For ten agents without any overlapping detection area, this accounts for 785 square units. The entire search space is 10,000 square units so that the probability that a threat pops-up within an agent’s detection range is only 7.85% which decreases even further once it is considered that an agent may be near the edge of the search space, or near another agent, and a portion of its detection range could be impotent or redundant.

Gradient following with Lévy performed the best for detecting both simple and smart pop-up threats. It is interesting that although this scheme had a slightly higher (worse) entropy than gradient following without Lévy for most alpha values, it was able to detect more pop-up threats. Both of these metrics are supposed to be measures of visitation frequency, but they are presenting some contrasting results. This is initially disappointing because entropy is a much more general performance measure than pop-up threat detection and it would be convenient if entropy performance could be directly extended to predict pop-up threat detection. This is because pop-up threat detection depends on many application specific factors including threat duration, detection range, threat multiplicity, and threat quantity and size.

From another perspective, the result is encouraging because gradient following with Lévy had superior performance for both the area coverage integral, and pop-up threat
detection. To further understand why more pop-up threats were able to be detected using gradient following with Lévy, in comparison to without Lévy, the pheromone field of each at the end of a single instance of the simulation is shown in Figure . Here, when Lévy is not used, larger voids in the pheromone field were present which could explain why more pop-up threats were not detected. When Lévy is added, these large voids were not present, which would explain why more pop-up threats were detected.

![Final pheromone field for gradient following without Lévy flight (Left) and with Lévy flight (right)](image)

**Figure 5-4:** Final pheromone field for gradient following without Lévy flight (Left) and with Lévy flight (right)

It is useful at this point to examine under what circumstances entropy may indicate an ordered system that would still perform poorly at detecting threats. Figure illustrates how two different scenarios with the same total number of cells and the same total number of visits have different entropy values. Scenario 2 has a lower entropy value (normally desirable), but 90% of the cells have never been visited in this scenario. Scenario 1 has a higher entropy value (normally not desirable), but every cell has been visited once or twice. This is an extreme example which was constructed to exaggerate one possible negative aspect of using entropy. Of course by viewing the mode of the visitation frequency, this
type of poor distribution could be detected. Based on the distributions shown in Figure and Figure, the situation exemplified in Figure does not appear to be a concern.

5.2.4 General Comparison Results

The most significant result and second important contribution of this work is that gradient following utilizing Lévy flight was shown to perform very well for the percent area coverage integral and for pop-up threats detected. Recall that these metrics are intended to convey the rate of coverage, exhaustivity of coverage, and visitation frequency. This means that this control scheme could be applied to a diverse range of practical applications for which these metrics are prime indicators of performance.
A secondary result is the introduction of entropy as a performance metric and an understanding of what it does and does not convey about system behavior. It is an excellent metric for quantifying the orderedness of a system, which is not-necessarily an intuitive concept. However, when using entropy, it must be considered that it may not exactly correlate with performance for specific visitation frequency dependent tasks, such as pop-up threat detection.

5.2.5 Contrasting Control Law and Lévy Foraging Hypothesis

The way the gradient following control law was constructed, noise had more influence when the gradient is very low (not much local information), and less influence when the gradient is very high (strong local information). This is intuitive because it makes more sense to use noise when there is not a strong indication of what direction to go.

When gradient following was then implemented with Lévy flight, Lévy flight was used whether the contemporaneous gradient was weak or strong. Contrast this with the Lévy foraging hypothesis which states that foraging behavior transitions from Brownian motion to Lévy flight as resources become scarcer. In the case of this pheromone-driven control scheme, the scarceness of resources can be analogized to the relative strength of the pheromone gradient. It would be interesting to implement a control scheme where not only would noise be more influential as the gradient decreased, but there would also be a shift from Brownian motion to Lévy flight. This control scheme would more closely correspond to the Lévy foraging theory and could lead to improved area coverage.

5.3 Limitations and Open Research Areas

There are a number of areas that this work left open for future examination. This work did not address the sparseness of agents, a measure of the number of agents per search
area. In this case the sparseness could be quantified as one agent per one thousand square units of area. Other works have examined the performance gained/lost by adding/removing agents and the resultant efficiency per agent [6], which is an important parameter when weighing the cost benefit of various swarm sizes.

The shape of the search space will influence performance, as will the behavior near the border of search space. A square search space would be appropriate for many practical applications and that is why it was chosen. However, the corners may present a challenge to the swarm and holding the edges at a constant pheromone value of zero will also affect diffusion patterns and consequent agent behavior. Some researchers have attempted to negate these edge and corner effects by simulating the space as a three dimensional torus [4]. This is a viable alternative for comparing different theoretical control schemes while eliminating bias due to edge effects, but it does not represent a realistic search space for practical applications.

The search space in this work was tessellated into small grid squares and the size of these squares will affect the validity of the results. Indeed, when initially a very coarse grid resolution of one unit was used (instead of finer 0.25-unit resolution used for simulations), agents were observed often moving in direction parallel to the tessellation axes. This was mitigated by increasing the resolution until these effects were no longer observed, but this resolution will still have an impact on the results. A second, coarser grid was used to track visitation frequency where a grid square was considered known if any part of that square was visited. The type of application will dictate an appropriate resolution for this second grid, and will affect the performance.
This work did not consider any communication requirements between the agents. If implemented using a real swarm, communication could be accomplished via local communication, as illustrated in Figure 1-4. This would typically require agents to remain within some communication distance of at least one other agent. The collaborative processing power of all agents could in this way be used to maintain an up-to-date virtual pheromone map of the search area. Alternatively, each agent could communicate with another central unit which would maintain this virtual pheromone map. As another alternative, some research has also been performed on using physical chemicals that diffuse and evaporate, just as ant pheromones do [69].

It was discussed how the sparseness of pop-up threats, threat duration, and detection capability of agents will influence pop-up threat detection. As these parameters are application specific, variation of these parameters was not investigated in more detail.

This work considered a simplified kinematic model where the agents could move freely in any direction from time step to temp step, which is how very small particles which undergo random walks have been observed to move. A more realistic model will have to account for inertia and the specific kinematic model of the robot. Additionally, other works have examined how area coverage performance varies when agents are allowed to move in only three directions, slight left, straight ahead, and slight right [4]. The effect of altering the probability of moving in each direction was also studied. This is more indicative of a realistic kinematic model, but it also would be interesting to study if limiting movement options in this way could improve performance. It was recognized that due to noise, when many small constant-length steps are taken, the agent has many opportunities to divert its course from a promising direction, to a less-promising direction. An explanation given for
why Lévy flight performs better is that it allows an agent to proceed further in a single step, hopefully in a promising direction. Limiting an agent’s available headings could have a similar effect and also is a way for the previous headings of an agent to influence its instantaneous heading. Essentially, this is a form of physical memory.

This work concentrated on the initial stages of area coverage up until the area is between 90 and 100% covered, as indicated in Figure 1. After this initial period, exploration-focused behavior transitions into a patrolling behavior. The behavior and performance of the different control schemes during patrolling could be studied in the same way but the results may be different than the results from this work.
Chapter 6

Conclusion

This work represented the confluence of the three distinct areas and ideas: (1) area coverage and measures of performance, (2) biased random walks, and (3) swarm intelligence. To provide area coverage, each individual member of a swarm used a type of biased random walk as the basis for a simple control law. To arrive at this control law, the well-established Keller-Segel reaction/diffusion equations were used as an analytical basis. Agents deposited a virtual pheromone as they moved through space which marked areas that had already been covered, and this pheromone diffused and evaporated over time. In this sense, pheromone can be thought of as information where diffusion helps to disseminate information and evaporation helps the older information to be forgotten. Other agents could detect this pheromone and their movement was a function of the gradient and random noise. From these simple rules for local interactions, system-level area coverage emerged.

Two main rounds of Matlab simulations were performed where in the first round, agents moved using constant-length path segments. The purpose of this round of simulations was to establish the relationship between the magnitude of noise, rate of evaporation, and rate of diffusion on performance and to find optimal values for these
parameter, which was an open research area. Ranges for each of these parameters were established and all possible combinations were tested, including cases without evaporation, and without diffusion. Performance in the first round was quantified using a single-value performance metric which included measures of rate and exhaustivity of coverage.

The second round of simulations introduced variable-length path segments using a different type of random walk called Lévy flight, where the path length was taken from a power-law distribution. A power-law distribution is “heavy-tailed”, meaning that most frequently a short path will be chosen, but occasionally a longer-path will be chosen from the “extended tail”. Using Lévy flight was a novel approach, because it had been proposed for search, but never implemented to provide area coverage. Three types of control schemes were tested and compared. The first scheme used constant-length paths and the best-performing values for noise, diffusion, and evaporation from the first round of simulations. This is the same scheme that was evaluated during the first round of simulations and can be labeled as “gradient following without Lévy flight”. The second scheme used those same optimal parameter values, but instead used variable-length paths. This can be thought of as “gradient following with Lévy flight”. The third scheme omitted any gradient following component, and movement was a non-biased Lévy flight. Hence, for this third scheme, noise, diffusion, and evaporation were not applicable. Results were compared using the same performance metric from the first round, as well as two additional metrics which were two different measures of visitation frequency. For the schemes that used Lévy flight, the alpha value which varies the shape of the power-law distribution, was also varied.
The first of two major contributions of this work was a better understanding of the interdependence of noise, evaporation, and diffusion for a gradient-following control scheme as achieved by a broad survey of these parameters during the first round of simulations. Specifically, noise was seen to be the parameter that had the largest impact on area coverage performance. The use of diffusion was shown to also improve performance, albeit without the impact of noise, and evaporation was shown to improve performance for some specific cases. The very best performing cases used a moderate amount of noise, diffusion, and evaporation which can be intuitively explained. Some amount of noise helps to escape local minima, whereas too much noise dilutes the influence of the pheromone. Some diffusion helps to spread information so that agents can detect a visited area at a greater distance, but too much diffusion removes all of the pheromone from the point of deposition, which makes visited and unvisited areas undiscernible. Finally, some evaporation helps an agent to eventually return to an area that has been already visited, whereas too much evaporation can, like too much diffusion, make visited and unvisited areas undiscernible. Subsequent statistical analysis confirmed not only that the main parameters had a statistically significant effect on performance, but also that cross-interactions between all the parameters were statistically significant.

The second major contribution of this work was the proposal of a gradient following control scheme incorporating Lévy flight which was compared to other control schemes in the second round of simulations. This control scheme had not before been applied to achieve area coverage and was shown to achieve superior performance. From all the control schemes tested, it provided the most exhaustive coverage, the highest rate of coverage, and the best ability at detecting pop-up threats. The best performance occurred
using small values of alpha, where a smaller value of alpha leads to choosing shorter path
segments more frequently than when using a higher value of alpha. It is believed that Lévy
flight improves performance by occasionally ignoring local information (pheromone),
which allows it to move through a covered area to an uncovered area. Qualitative analysis
of the pheromone field generated by this scheme show fewer large gaps, which explains
how more pop-up threats could be detected.

Further contributions of this work include three performance metrics which were
devised to give a comprehensive assessment of area coverage performance by providing
single-value measures of the coverage exhaustivity, rate of coverage, and visitation
frequency. The first metric, labeled as “percent area coverage integral”, is most useful
during the initial phases or area coverage, but becomes less useful as behavior switches
from exploration to patrolling, because it places no value on visiting an area more than one
time. The second performance metric was the entropy of visitation frequency, which is
very attractive because of its generality. In comparison, the third metric of pop-up threat
detection also provides a measure of visitation frequency, but it requires many additional
application specific parameters to be specified (i.e. detection range, threat duration, threat
multiplicity). Visitation entropy and pop-up threat detection were not found to correspond
exactly with one another, but it is believed that entropy provides a good estimate for more
application specific threat-detection ability. Although two-types of pop-up threats were
considered, one of which could avoid popping up the detection range of an agent,
performance was similar due to the relative sparsity of agents.

One of the most promising areas for future work is to design an area coverage
scheme that transitions from Brownian motion to Levy flight depending on the strength of
the pheromone gradient whereas this work used Levy regardless of the gradient. The idea that this may offer improved performance stems from the Levy foraging hypothesis which proposes that animals switch from Brownian motion to Levy flight as food becomes scarcer.
References


[22] X.-S. Yang, Recent Advances in Swarm Intelligence and Evolutionary


M. Muller and R. Wehner, “Path Integration in Desert Ants, Cataglyphis fortis,”


Appendix A

Program Code Written in MATLAB

tic
clc
clear
clear plot

% display settings
update1=0; %IF one, update agent position plot in real time,
updateevery=500; %update position every so many time steps
displayoutputs=0; %IF one, output plots will be displayed
printoutputs=0; %IF one, output plots will be exported as gIF
exportcount=1; %IF one, cells visited at each time step and frequency will be exported to excel
exportfrequency=1; %sets how often data will be sampled (must be limited due to excel column limit)
filename = 'testtime.xlsx'; %only needed IF exportcount=1

% mode settings
mode=0; %0=gradient following with constant path length, 1=gradient following with Lévy flight, 2=pure Lévy flight
constantlength=1; %only needed FOR mode=0
Lévyalpha=1.5; %only needed FOR mode=1 or mode=2 Lévy flight parameter

% general settings
deltat=1; %time step
time_resolution=1*deltat;
gamma=0.0001; %evaporation rate=percent pheromone/second
deposit=1; %deposition rate = units pheromone/second
travel=1; %travel speed = units distance/second
noise=0.05; %magnitude of noise, IF gradient strong, noise will be negligible, IF weak, noise will be significant

gamma_effective=gamma*time_resolution; %adjusted to time step
deposit_effective=deposit*time_resolution; %adjusted to time step
travel_effective=travel*time_resolution; %adjusted to time step

coverage=1.0; %program stops when this percent has been visited at least once.
time=3000; %maximum time steps allowed
agents=10; %number of randomly initialized agents
% diffusion settings
D=0.0001; %diffusion rate
resolution=4; %units per unit length
deltax=1/resolution;
deltay=1/resolution;

% animation settings
loops=time/deltat;
F(loops) = struct('cdata',[],'colormap',[]);

% check IF FTCS diffusion will be stable
IF D>deltax*deltax/(4*deltat);
    disp('diffusion is unstable due to deltax,deltat,diffusion constant')
    pause
END

% search area boundaries
xmin=0;
xmax=100;
ymin=0;
ymax=100;

% Pop Up Threat settings
popduration=30; %duration of popup threat
popdetection=5; %detection range of agent FOR pop-up threat
popquantity=round(time/popduration); %number of pop-up occurrences

% create grid for pheromone, better resolution possible
xgv=linspace(xmin,xmax,(xmax-xmin)/deltax); %assume grid cell size of 1 unit
ygv=linspace(ymin,ymax,(ymax-ymin)/deltay); %assume grid cell size of 1 unit
[X,Y] = meshgrid(xgv,ygv); %create grid

% create grid FOR area coverage, resolution of one unit
xvv=linspace(xmin,xmax,(xmax-xmin)); %assume grid cell size of 1 unit
yvv=linspace(ymin,ymax,(ymax-ymin)); %assume grid cell size of 1 unit
[XV,YV] = meshgrid(xvv,yvv); %create grid

q=1; %multiple set index
qmax=1;
WHILE q<=qmax
    %D=0.01; %only needed because testing multiple sets
    %gamma=0.1; %only needed because testing multiple sets

    r=1; %multiple run index
    rmax=1; %maximum number of runs
    WHILE r<=rmax
        s=1; %statistical run, no variation
        smax=1;
        WHILE s<=smax;
            smartpopdetected=zeros(1,popquantity); %will track if detected, if one, means detected
            smartpoplocationx=xmax.*rand(1,popquantity);
            smartpoplocationy=ymax.*rand(1,popquantity);
            simplepopdetected=zeros(1,popquantity); %will track if detected, if one, means detected
            simplepoplocationx=xmax.*rand(1,popquantity);
        END
    END
END
simplepoplocationy=ymax.*rand(1,popquantity);

% these diffusion constants inside of calc due to varying D value
alpha=D*deltat/deltax^2;
beta=D*deltat/deltay^2;

% initiate UAV
Nx=100.*rand(1,agents); % initialize population randomly
Ny=100.*rand(1,agents); % initialize population randomly

pathremaining=zeros(1,agents); % will give remaining path length
agentheading=zeros(1,agents); % will track agent headings
pheromone=zeros(numel(xgv),numel(ygv)); % initial matrix of known/unknown ... cells, FOR testing assume all known
deltapheromone=zeros(numel(xgv),numel(ygv)); % will be used FOR changes to ... pheromone in each time step
visited=zeros(numel(xvv),numel(yvv)); % once a cell has been visited once, will turn to one
frequency=zeros(numel(xvv),numel(yvv)); % will track how many times each cell has been visited

% monitors cells that have not been visited
FOR k=1:numel(Nx)
    currentvisitcellx=floor(Nx(1,k))+1; % finds current cell x
    currentvisitcelly=floor(Ny(1,k))+1; % finds current cell y
    visited(currentvisitcellx,currentvisitcelly)=1;
    frequency(currentvisitcellx,currentvisitcelly)=1;
END
t=0; % time index
% UAV position graphing
IF displayoutputs==1;
c = linspace(1,10,numel(Nx));
clf(figure(2))
clf(figure(3))
clf(figure(4))
figure(2); % display UAV position
scatter(Nx(1,:),Ny(1,:),[]); % will track when to export data
xlim([xmin xmax])
ylim([ymin ymax])
hold on;
END

u=1; % update index
loop=1; % loop index
stray=0; % stray counter
export=1;

count=ones(1,time/exportfrequency);
entropy=zeros(1,time/exportfrequency);
deviation=zeros(1,time/exportfrequency);
WHILE t<=time

% now find gradient
Nx_prev=Nx; % will store previous positions
Ny_prev=Ny; % will store previous positions
FOR k=1:numel(Nx); % now update agent positions
    currentcellx=floor(Nx(1,k)/deltax)+1; % finds current cell x gives column number
currentcelly=floor(Ny(1,k)/deltay)+1; %finds current cell y %gives row number %should never be outside of grid
IF Nx(1,k)<=xmin||Nx(1,k)>xmax||Ny(1,k)<=ymin||Ny(1,k)>ymax %if …
    outside of grid, move back toward center of grid
    agentheading(k)=atan2(-(Ny(1,k)-50),-(Nx(1,k)-50)); %heading is …
        summation of noise and reverse gradient
    pathremaining(k)=0;
    stray=stray+1;
ELSE %if not outside of grid
    IF pathremaining(k)<=0 %IF a new path needs calculated
        IF mode~=2 %IF not pure Lévy flight, calculate gradient
            up=currentcelly+1; %gives row number
            down=currentcelly-1; %gives row number
            left=currentcellx-1; %gives column number
            right=currentcellx+1; %gives column number
                %all set to one just to track for special circumstances
            phero1=1;
            phero3=1;
            phero5=1;
            phero7=1;
            %look for special conditions at edge of grid
            IF up==ymax/deltay+1;
                phero1=0;
            END
            IF down==ymin/deltay;
                phero5=0;
            END
            IF right==xmax/deltax+1;
                phero3=0;
            END
            IF left==xmin/deltax;
                phero7=0;
            END
            END
            %now set pheromone levels if no special conditions apply
            IF phero1==0;
                phero1=pheromone(up,currentcellx);
            END
            IF phero5==0;
                phero5=pheromone(down,currentcellx);
            END
            IF phero3==0;
                phero3=pheromone(currentcelly,right);
            END
            IF phero7==0;
                phero7=pheromone(currentcelly,left);
            END
            %these are reverse gradient values (pheromone repulsive)
            gradientx=(phero7-phero3)/(2*deltax);
            gradienty=(phero5-phero1)/(2*deltay);
ELSE %for pure Lévy flight
    gradientx=0;
    gradienty=0;
END

% these are the noise values
noisearngle=2*pi*rand;
noisex=noise*cos(noisearngle);
noisey=noise*sin(noisearngle);
agentheading(k)=atan2(gradienty+noisey,gradientx+noisex);
    %heading is summation of noise and reverse gradient
IF mode==0 %non-Lévy flight
    pathremaining(k)=constantlength-travel_effective;
ELSE %Lévy flight
    pathremaining(k)=rand^((1/Lévyalpha)-travel_effective);
END

% IF choices cause path to stray outside of grid, recalculate
WHILE Nx(1,k)+travel_effective*cos(agentheading(k))<xmin
    \|Nx(1,k)+travel_effective*cos(agentheading(k))=xmax…
    \|Ny(1,k)+travel_effective*sin(agentheading(k))<ymin…
    \|Ny(1,k)+travel_effective*sin(agentheading(k))>ymax
    noiseangle=2*pi*rand;
    noisex=noise*cos(noisearngle);
    noisey=noise*sin(noisearngle);
    agentheading(k)=atan2(noisey,noisex);
    %heading is summation of noise and reverse gradient
    IF mode==0 %non-Lévy flight
        pathremaining(k)=constantlength-travel_effective;
    ELSE %Lévy flight
        pathremaining(k)=rand^((1/Lévyalpha)-travel_effective);
    END
END %END WHILE new path generated moves outside grid

ELSE %IF a new path doesn't need calculated
    IF Nx(1,k)+travel_effective*cos(agentheading(k))<xmin…
        \|Nx(1,k)+travel_effective*cos(agentheading(k))=xmax…
        \|Ny(1,k)+travel_effective*sin(agentheading(k))<ymin…
        \|Ny(1,k)+travel_effective*sin(agentheading(k))>ymax
        WHILE Nx(1,k)+travel_effective*cos(agentheading(k))<xmin\|Nx(1,k)…
            +travel_effective*cos(agentheading(k))=xmax\|Ny(1,k)+travel_effective…
            *sin(agentheading(k))<ymin\|Ny(1,k)+travel_effective…
            *sin(agentheading(k))>ymax
            noiseangle=2*pi*rand;
            noisex=noise*cos(noisearngle);
            noisey=noise*sin(noisearngle);
            agentheading(k)=atan2(noisey,noisex);
            %heading is …
            summation of noise and reverse gradient
    IF mode==0 %non-Lévy flight
        pathremaining(k)=constantlength-travel_effective;
    ELSE %Lévy flight
        pathremaining(k)=rand^((1/Lévyalpha)-travel_effective);
    END %END mode determination
END %END WHILE
ELSE %continue on path
\begin{equation}
\begin{aligned}
\text{Nx}(1,k) &= \text{Nx}(1,k) + \text{travel\_effective} \times \cos(\text{agentheading}(k)) \\
\text{Ny}(1,k) &= \text{Ny}(1,k) + \text{travel\_effective} \times \sin(\text{agentheading}(k)) \\
\text{pathremaining}(k) &= \text{pathremaining}(k) - \text{travel\_effective} \\
\end{aligned}
\end{equation}
\%decrease length of path remaining
\end{verbatim}
END %IF current path will move out of grid
\end{verbatim}
END %IF a new path needs created
\end{verbatim}
END %IF outside of grid (shouldn't be needed now)
\end{verbatim}
END %END FOR agent loop

\text{deltapheromone} = \text{zeros(numel(xgv),numel(ygv))};
% monitors cells that have not been visited and will also add pheromone
\text{FOR } k=1:\text{numel(Nx)} %put breakpoint here IF needed
% this for pheromone
\text{ENDcellx} = \text{floor}(\text{Nx}(1,k)/\text{deltax}) + 1; %finds END cell x %gives column
\text{ENDcelly} = \text{floor}(\text{Ny}(1,k)/\text{deltay}) + 1; %finds END cell y %gives row
\text{startcellx} = \text{floor}(\text{Nx\_prev}(1,k)/\text{deltax}) + 1; %finds start cell x %gives column
\text{startcelly} = \text{floor}(\text{Ny\_prev}(1,k)/\text{deltay}) + 1; %finds start cell y %gives row
\text{currentcellx} = \text{startcellx}; %will increment through entire path %gives column
currentcelly = \text{startcelly}; %will increment through entire path %gives row
currentx = \text{Nx\_prev}(1,k); %will increment through entire path
currenty = \text{Ny\_prev}(1,k); %will increment through entire path
\text{visitedcells} = \text{abs}(\text{ENDcellx} - \text{startcellx}) + \text{abs}(\text{ENDcelly} - \text{startcelly})
\text{temp0} = 0; %will track how many visited cells have been found
\text{slope} = (\text{Ny}(1,k) - \text{Ny\_prev}(1,k)) / (\text{Nx}(1,k) - \text{Nx\_prev}(1,k));
\text{heading} = \text{atan2}(\text{Ny}(1,k) - \text{Ny\_prev}(1,k), \text{Nx}(1,k) - \text{Nx\_prev}(1,k));
% this for visitation
\text{ENDvisitcellx} = \text{floor}(\text{Nx}(1,k)) + 1; %finds END cell x %gives column
\text{ENDvisitcelly} = \text{floor}(\text{Ny}(1,k)) + 1; %finds END cell y %gives row
\text{startvisitcellx} = \text{floor}(\text{Nx\_prev}(1,k)) + 1; %finds start cell x %gives column
\text{startvisitcelly} = \text{floor}(\text{Ny\_prev}(1,k)) + 1; %finds start cell y %gives row
\text{currentvisitcellx} = \text{startvisitcellx}; %will increment through entire path %gives column
currentvisitcelly = \text{startvisitcelly}; %will increment through entire path %gives row
currentvisitx = \text{Nx\_prev}(1,k); %will increment through entire path
currentvisity = \text{Ny\_prev}(1,k); %will increment through entire path
\text{visitedvisitcells} = \text{abs}(\text{ENDvisitcellx} - \text{startvisitcellx}) + \text{abs}(\text{ENDvisitcelly} - \text{startvisitcelly})
\text{temp1} = 0; %will track how many visited cells have been found
% this for pheromone
\text{WHILE } \text{temp0}<\text{visitedcells} \&\& \text{mode} = 2;
\text{IF } \text{heading} >= 0 \&\& \text{heading} <= \text{pi}/2 %first quadrant, check right boundary, slope +
\text{BR} = (\text{currentcellx} - 1) \times \text{deltax}; %bottom right corner
\text{TR} = \text{BR} + \text{deltay}; %top right corner
\text{IR} = (\text{currentcellx} \times \text{deltax} - \text{currentx}) \times \text{slope} + \text{currenty}; %y = \text{intercept} = \text{run} \times \text{slope} (\text{which equals rise}) + \text{current y}
\text{IF } \text{IR} > \text{BR} \&\& \text{IR} < \text{TR} %advance right
\text{currentcellx} = \text{currentcellx} + 1; %x changes but y stays same
\text{currentx} = (\text{currentcellx} - 1) \times \text{deltax};
\text{currenty} = \text{IR};
\text{ELSE } %advance up
\text{currentcelly} = \text{currentcelly} + 1; %y changes but x stays same
\text{END}
\text{ELSE IF } \text{heading} > \text{pi}/2 %quadrant 2, check left boundary, slope -
BL=(currentcellx-1)*deltax; %bottom left corner
TL=(currentcelly-1)*deltay; %top right corner
IL=((currentcellx-1)*deltax-currentx)*slope+currenty; %y intercept = run*slope (which equals rise) + current y
IF IL>=BL&&IL<=TL %advance left
    currentcellx=currentcellx-1; %x changes but y stays same
    currentx=(currentcellx)*deltax;
    currenty=IL;
ELSE %advance up
    currentcelly=currentcelly+1; %y changes but x stays same
END
ELSE IF heading<=-pi/2 %check quadrant 3, left boundary, slope -
    BL=(currentcelly-1)*deltay; %bottom right corner
    TL=BL+deltay; %top right corner
    IL=((currentcellx-1)*deltax-currentx)*slope+currenty;
    %y intercept = run*slope (which equals rise) + current y
    IF IL>=BL&&IL<=TL %advance left
        currentcellx=currentcellx-1; %x changes but y stays same
        currentx=(currentcellx)*deltax;
        currenty=IL;
    ELSE %advance down
        currentcellx=currentcellx+1; %y changes but x stays same
    END
ELSE %must be quadrant 4, right boundary, slope +
    BR=(currentcelly-1)*deltay; %bottom right corner
    TR=BR+deltay; %top right corner
    IR=(currentcellx*deltax-currentx)*slope+currenty; %y intercept = run*slope (which equals rise) + current y
    IF IR>=BR&&IR<=TR %advance right
        currentcellx=currentcellx+1; %x changes but y stays same
        currentx=(currentcellx)-1)*deltax;
        currenty=IR;
    ELSE %advance down
        currentcellx=currentcellx-1; %y changes but x stays same
    END
END% END quadrant 3 and 4
END %END quadrant 2
END %END quadrant 1
IF currentcellx*deltax<=xmin||currentcellx*deltax>xmax
    ||currentcellx*deltay<=ymin||currentcellx*deltay>ymax
ELSE
deltapheromone(currentcellx,currentcellx)=deltapheromone(currentcellx,currentcellx)+deposit_effective/visitedcells; %add pheromone to visited cell
END

temp0=temp0+1;
END %END WHILE
% this for visit
WHILE temp1<visitedvisitcells;
    IF heading>=0&&heading<=pi/2 %first quadrant, check right boundary, slope +
        BR=(currentvisitcellx-1); %bottom right corner
        etc.
    ELSE
        etc.
    END
    currentcellx=
    currentcellx=
    etc.
TR=BR+1; %top right corner
IR=(currentvisitcellx-currentvisitx)*slope+currentvisity; %y…
    intercept = run*slope (which equals rise) + current y
IF IR>=BR&&IR<=TR %advance right
currentvisitcellx=currentvisitcellx+1; %x changes but y stays same
currentvisix=(currentvisitcellx-1);
currentvisity=IR;
ELSE %advance up
currentvisitcellv=currentvisitcellv+1; %y changes but x stays same
END

ELSE IF heading>pi/2 %quadrant 2, check left boundary, slope -
    BL=(currentvisitcellv-1); %bottom right corner
    TL=BL+1; %top right corner
    IL=((currentvisitcellx-1)-currentvisitx)*slope+currentvisity; %y intercept =…
        run*slope (which equals rise) + current y
    IF IL>=BL&&IL<=TL %advance left
currentvisitcellx=currentvisitcellx-1;%x changes but y stays same
currentvisx=(currentvisitcellx);
currentvisity=IL;
ELSE %advance up
currentvisitcellv=currentvisitcellv+1;%y changes but x stays same
END

ELSE IF heading<=-pi/2 %check quadrant 3, left boundary, slope +
    BL=(currentvisitcellv-1); %bottom right corner
    TL=BL+1; %top right corner
    IL=((currentvisitcellx-1)-currentvisitx)*slope+currentvisity; %y intercept =…
        run*slope (which equals rise) + current y
    IF IL>=BL&&IL<=TL %advance left
currentvisitcellx=currentvisitcellx-1;%x changes but y stays same
currentvisx=(currentvisitcellx);
currentvisity=IL;
ELSE %advance down
currentvisitcellv=currentvisitcellv-1;%y changes but x stays same
END

ELSE %must be quadrant 4, right boundary, slope -
    BR=(currentvisitcellv-1); %bottom right corner
    TR=BR+1; %top right corner
    IR=(currentvisitcellx- currentvisitx)*slope+currentvisity; %y intercept =…
        run*slope (which equals rise) + current y
    IF IR>BR&&IR<=TR %advance right
currentvisitcellx=currentvisitcellx+1;%x changes but y stays same
currentvisx=(currentvisitcellx-1);
currentvisity=IR;
ELSE %advance down
currentvisitcellv=currentvisitcellv-1;%y changes but x stays same
END
END% END quadrant 3 and 4
END %END quadrant 2
END %END quadrant 1

IF currentvisitcellx<=xmin|currentvisitcellx>xmax…
currentvisitcelly <= y_min || currentvisitcelly > y_max
ELSE
   visited(currentvisitcelly, currentvisitcellx) = 1;
   IF currentvisitcellx != startvisitcellx || currentvisitcelly != startvisitcelly
      % only adds to frequency IF not already in cell
      frequency(currentvisitcelly, currentvisitcellx)...
      = frequency(currentvisitcelly, currentvisitcellx) + 1;
   END
END

END %end while

temp1 = temp1 + 1;
END %end for loop

t = t + deltat; % moved from below
IF export == export_frequency/deltat
   count(1, loop) = sum(sum(visited));
   e = numel(unique(frequency));
   clear freqvalues;
   freqvalues = unique(frequency);
   temp3 = 0;
   temp4 = 0;
   avgfreq = 1/(numel(xvv)*numel(yvv))*numel(Nx);
   FOR i = 1:e
      temp1 = sum(sum(frequency == freqvalues(i, 1))); % number of cells with that many visits
      temp2 = temp1/(numel(xvv)*numel(yvv)); % this is probability of this frequency
      temp3 = temp3 + temp2*log2(temp2); % this finds the entropy component
      temp4 = temp4 + temp1*(freqvalues(i, 1)/t-avgfreq)^2; % this finds the sum of all deviations
   END
   entropy(1, loop) = -temp3;
   deviation(1, loop) = temp4/(numel(xvv)*numel(yvv));
   export = 0; % reset export timer
   loop = loop + 1;
END
export = export + 1; % increment export timer
IF mode==2; % not needed FOR pure Lévy
deltapheromone=deltapheromone+pheromone.*(-gamma_effective); % should be negative because evaporation, trying to increase speed
pheromonenew=zeros(numel(xgv),numel(ygv));
FOR i=2:numel(xgv)-1 % adds diffusing pheromone to deltapheromone matrix
  FOR j=2:numel(ygv)-1
    pheromonenew(i,j)=alpha*(pheromone(i+1,j)+pheromone(i-1,j))
    + beta*(pheromone(i,j+1)+pheromone(i,j-1))+(1-2*alpha-2*beta)*pheromone(i,j);
  END
END
FOR i=1 % solve FOR special case of left hand boundary
  FOR j=2:numel(ygv)-1
    pheromonenew(i,j)=alpha*(pheromone(i+1,j)+0)+beta*(pheromone(i,j+1)+
    +pheromone(i,j-1)+(1-2*alpha-2*beta)*pheromone(i,j);
  END
END
FOR i=xmax/deltax % solve FOR special case of right hand boundary
  FOR j=2:numel(ygv)-1
    pheromonenew(i,j)=alpha*(0+pheromone(i-1,j))+beta*(pheromone(i,j+1)+
    +pheromone(i,j-1)+(1-2*alpha-2*beta)*pheromone(i,j);
  END
END
FOR j=1 % solve FOR special case of lower boundary
  FOR i=2:numel(xgv)-1
    pheromonenew(i,j)=alpha*(pheromone(i+1,j)+pheromone(i-1,j))
    + beta*(pheromone(i,j+1)+0)+(1-2*alpha-2*beta)*pheromone(i,j);
  END
END
FOR j=ymax/deltay % solve FOR special case of upper boundary
  FOR i=2:numel(xgv)-1
    pheromonenew(i,j)=alpha*(pheromone(i+1,j)+pheromone(i-1,j))
    + beta*(0+pheromone(i,j-1)+(1-2*alpha-2*beta)*pheromone(i,j);
  END
END
% solve FOR special case of corners
pheromonenew(1,1)=alpha*(pheromone(2,1)+0)+beta*(pheromone(1,2)+0)+(1-2*alpha-2*beta)*pheromone(1,1);
pheromonenew(1,ymax/deltay)=alpha*(pheromone(2,ymax/deltay)+0)+..
  beta*(0+pheromone(1,ymax/deltay-1)+(1-2*alpha-2*beta)*pheromone(1,ymax/deltay);
pheromonenew(xmax/deltax,ymax/deltay)=alpha*(0+pheromone(xmax/deltax-1,...
  ymax/deltay)+beta*(0+pheromone(xmax/deltax,ymax/deltay-1)+(1-2*alpha-2*beta)*pheromone(xmax/deltax,ymax/deltay);
pheromonenew(xmax/deltax,1)=alpha*(0+pheromone(xmax/deltax-1,1))
  + beta*(0+pheromone(xmax/deltax,1+1)+0)+(1-2*alpha2*beta)*pheromone(xmax/deltax,1);
pheromone=pheromonenew;
pheromone=pheromone+deltapheromone;
END

IF displayoutputs==1
  figure(2); % display UAV position
scatter(Nx(1,:),Ny(1,:),[],c);
drawnow
F(loop) = getframe(gcf);
IF update1==1|u*deltat==updateevery;
    h(1)=scatter(Nx(1,:),Ny(1,:),[],'MarkerEdgeColor',[0 0 0],'MarkerFaceColor',...
[0 0 0],'SizeData',75);
    pheromonemap=pheromone(1:numel(xgv),1:numel(ygv));
    figure(4) %display pheromone distribution, replace figure 1
    contourf(X,Y,pheromonemap);
    colorbar;
    IF exportcount==1
        fig2name=strcat('Position',num2str(t));
        fig4name=strcat('Pheromone',num2str(t));
        print(figure(2),fig2name,'-dpng');
        print(figure(4),fig4name,'-dpng');
    END
    delete(h(1));
    u=0;
END
END

u=u+1;
END

%plot naming FOR export
fig2name=strcat('Position,q-',num2str(q),',r-',num2str(r),',s-',num2str(s));
fig3name=strcat('Coverage,q-',num2str(q),',r-',num2str(r),',s-',num2str(s));
fig4name=strcat('Pheromone,q-',num2str(q),',r-',num2str(r),',s-',num2str(s));
rowname=strcat('A',num2str(s));
sheetname1=strcat('cover',',q-',num2str(q),',r-',num2str(r));
sheetname2=strcat('entropy',',q-',num2str(q),',r-',num2str(r));
sheetname3=strcat('dev.',',q-',num2str(q),',r-',num2str(r));
sheetname4=strcat('sipop',',q-',num2str(q),',r-',num2str(r));
sheetname5=strcat('smpop',',q-',num2str(q),',r-',num2str(r));

IF displayoutputs==1
    figure(2); %display UAV position
    scatter(Nx(1,:),Ny(1,:),[],'MarkerEdgeColor',[0 0 0],'MarkerFaceColor',[0 0 0]);
xlim([xmin xmax])
ylim([ymin ymax])
hold on;

    IF exportcount==1
        figure(3); %display percent of area explored
        plot(1:loop-1,(count(:)/(numel(xvv)*numel(yvv))),'b-');
xlabel('time,t')
ylabel(' (% of area explored')
END

pheromonemap=pheromone(1:numel(xgv),1:numel(ygv));
figure(4) %display pheromone distribution, replace figure 1
    contourf(X,Y,pheromonemap);
    colorbar;
    hold on;
END

IF printoutputs==1&&displayoutputs==1;
  print(figure(2),fig2name,'-dpng');
  IF exportcount==1
    print(figure(3),fig3name,'-dpng');
  END
  print(figure(4),fig4name,'-dpng');
END

IF exportcount==1;
  warning('off','MATLAB:xlswrite:AddSheet')
  xlswrite(filename,count,sheetname1,rownname)
  xlswrite(filename,entropy,sheetname2,rownname)
  xlswrite(filename,deviation,sheetname3,rownname)
  xlswrite(filename,simplepopdetected,sheetname4,rownname)
  xlswrite(filename,smartpopdetected,sheetname5,rownname)
END

s=s+1
END %WHILE statistical run

r=r+1
%D=D/10;
%gamma=gamma/10;
%gamma_effective=gamma*time_resolution; %adjusted to time step
Lévyalpha=Lévyalpha+0.5;
END %WHILE multiple run
%noise=noise+0.1;
q=q+1;
END %WHILE set

toc