A Swarm Intelligence Approach to Distributed Mobile Surveillance

Michael Brian Marshall

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
In
Computer Engineering

September 9, 2005

Bradley Department of Electrical and Computer Engineering
Blacksburg, Virginia

Keywords: Swarm Intelligence, Distributed Surveillance, Mobile Robotics, Emergence

A Swarm Intelligence Approach to Distributed Mobile Surveillance

Michael Brian Marshall

Abstract

In the post-9/11 world, new and improved surveillance and information-gathering technologies have become a high-priority problem to be solved. Surveillance systems are often needed in areas too hostile or dangerous for a direct human presence. The field of robotics is being looked to for an autonomous mobile surveillance system. One major problem is the control and coordination of multiple cooperating robots. Researchers have looked to the distributed control strategies found in nature in the form of social insects as an inspiration for new control schemes. Swarm intelligence research centers around the interactions of such systems and how they can be applied to contemporary problems. In this thesis, a surveillance system of mobile autonomous robots based on the principles of swarm intelligence is presented.

Dedication

To my family for always encouraging me to be better than ever

Acknowledgements

First of all, I want to thank my advisor Dr. Peter Athanas for his support and guidance throughout my graduate studies. His patience and insight were key to the completion of this project. Dr. Scott Harper and Dr. Jae Park also provided invaluable guidance and criticism throughout my work. I would also like to thank my committee members Dr. Cameron Patterson and Dr. Mark Jones.

For sharing in the joys and frustrations of ER1 robotic hardware, I would like to thank Chris Einsmann, Ben Muzal, Matt Blanton, and James Goodwin. Their input and assistance added greatly to the knowledge created by this project.

I want to thank my parents for giving me the life-long love and support allowing me to take on this challenge

Table of Contents

Acknowledgements iv

Introduction 1

1.1 Thesis Statement 2

1.2 Thesis Objectives 2

1.3 Thesis Organization 3

Background 4

2.1 Intelligence 4

2.2 Terminology 5

2.3 Collective Robotics 5

2.3.1 Advantages 5

2.3.2 Autonomy 6

2.3.3 Control Architectures 7

2.3.4 Communication 8

2.3.5 Collective Tasks 9

2.4 Swarm Intelligence 10

2.4.1 Biological Inspirations 10

2.4.2 Emergence and Self-organization 11

2.4.3 Scientific Interests 11

2.4.4 Advantages to Robotic Control 12

2.4.5 Other Applications of Swarm Intelligence 13

Surveillance System Design Overview 14

3.1 Design Criteria 14

3.2 Ant Foraging Strategy 15

3.2.1 Communication weakness 16
3.2.2 Communication Feedback Threshold 17
3.3 Final Design 18
Simulation 22
4.1 The Need for Simulation 22
4.2 StarLogo 22
4.3 Implementation in Simulation 23
Physical Robotics 26
5.1 The need for Physical Implementation 26
5.2 ER1 Personal Robotic System 27
5.2.1 Drive System 27
5.2.2 Image Recognition 27
5.2.3 IR Sensors 29
5.2.4 Battery Power 29
5.3 ERSP 29
5.4 Implementation details 30
Results and Analysis 33
6.1 Results Comparison 33
6.2 Result Calculations 33
6.3 Simulation Results 34
6.3.1 Simulation Test Procedures 34
6.3.2 Large Scale Results 35
6.3.3 Physical Imitation Results 36

6.4 Physical ER1 Results 38

6.4.1 Physical Test Procedure 38

6.4.2 Positional Certainty 40

6.4.3 Number of Sightings 40

6.5 Analysis 41

6.5.1 Overview 41

6.5.2 Simulation 41

6.5.3 Physical Testing 42

6.5.4 Simulation versus Physical Results 43

Conclusions 44

7.1 Summary 44

7.2 Future Work 45

List of Figures

Figure 1	6
Figure 2	11
Figure 3	16
Figure 4	18
Figure 5	20
Figure 6	23
Figure 7	24
Figure 8	27
Figure 9	28
Figure 10	28
Figure 11	29
Figure 12	32
Figure 13	39

Chapter 1

Introduction

Swarm Intelligence is a relatively new area of study, related to the field of distributed control. Swarm intelligence seeks to use the behavior and control mechanisms found in nature especially in social insects to control multiple robots and to solve problems. The central idea behind swarms is that local interactions between simple agents can create complex global behavior. The control is completely decentralized with every agent responsible for its individual actions. Being a young field of research, the applications and problems being solved with swarm intelligence are easier than those of the more matured areas of distributed control. Most research involving swarms has focused on resource gathering or pattern formation through self-organization [46] [42].

Surveillance is a natural extension of the previous work done in this field. Instead of gathering physical resources such as landmines [45], information about specific targets is collected. In the post-9/11 world, new and improved surveillance and information-gathering technologies has become a high-priority problem to be solved. According to [24], DARPA has increased its funding for surveillance and security related technologies by approximately $24 million between 2004 and 2005. Likewise, the Department of Homeland Security increased its 2005 budget for Border Patrol Surveillance and Sensor Technology to $64.1 million. The military is also adopting a swarm-like organization [5] for the purpose of trying to create smaller, more technologically equipped units. As the number of soldiers shrinks, more advanced autonomous solutions will be required.

In this thesis, a surveillance system of mobile autonomous robots based on the principles of swarm intelligence is presented. The autonomous agents mimic the behavior of ants when foraging for food. In this case they are foraging for recognizable targets. The system implements a reaction threshold, which is lacking in all previous attempts at a swarming system. This threshold improves system performance by controlling swarm size and reducing the possibility of a deadlock state.

1.1 Thesis Statement

The objective of this thesis is to illustrate, isolate, and resolve a vulnerability in contemporary swarm intelligence systems that have no communication feedback. By blindly trusting and reacting to any signal from a perceived agent, the system has the possibility of falling into a deadlock state. This thesis also proposes a solution to safeguard against this situation and improve efficiency by instituting a reaction threshold that must be crossed before an agent will react to communication.

To accomplish this objective, a surveillance system of mobile autonomous agents was designed using a swarm intelligence control scheme based on the foraging strategy of ants. This system was used as the test bed for the proposed communication mechanisms mentioned above. Modeling of the system was done both in simulation and in the laboratory using physical robots. The simulation package called StarLogo [47] allowed for scenarios involving hundreds of robotic agents, which would be infeasible in the physical testing given the university's finite supply of money and space. The physical testing used the ER1 robotic hardware manufactured by Evolution Robotics [22]; this will be discussed thoroughly in Chapter 5.

1.2 Thesis Objectives

The specific goal of this thesis work is to investigate the tolerance of swarm intelligent systems to errors in communication. Because this is the first attempt at research in this area at Virginia Tech, building a framework for this and future research was a fundamental and necessary goal.

The objectives of this research are:

- To demonstrate a communication weakness in many of the current swarm intelligent systems and propose a viable solution to safeguard against the problem and improve performance.
- To develop a surveillance system of multiple autonomous robots using a swarm intelligence approach as a test bed for the experiments.
- To create reusable code structures, hardware, and research methodologies providing the foundations for further studies into swarm intelligence at Virginia Tech.

1.3 Thesis Organization

Chapter 2 provides background information on swarm intelligence and distributed robotic control systems. The system design is discussed in Chapter 3. Chapters 4 and 5 present the software simulation and hardware testing respectively. These chapters give a detailed description of the technologies used to accomplish the work and the results obtained. Chapter 6 is a critical analysis of the proposed system and the results gathered. Chapter 7 summarizes the research project, results produced, and the author's contributions.

Chapter 2

Background

This chapter provides background information regarding collective robotics and swarm intelligence. Where possible, it points out links to other related fields such as artificial intelligence, distributed control, and biological societies.

2.1 Intelligence

What defines intelligence and when can a computer system be classified as intelligent? Webster's Dictionary defines intelligence as "the capacity to know or understand" [50], which is rather unhelpful since every computer has the ability to "know" pieces of data store in memory. Alan Turing suggested that we may begin to consider a robot intelligent when it could fool humans into thinking it was a human [37].

According to Bonabeau [9], an entity must satisfy its "viability" conditions while interacting in its environment. An intelligent individual must maintain an "identity" throughout its existence. For example, a "mobile" robot must navigate its surroundings while avoiding obstacles and never enter a deadlock state. Extending this to a team of multiple robots, the team has collective intelligence if the viability of the group as a whole is required for each individual to sustain viability. This is the definition of intelligence used for the remainder of this thesis. A full discussion of the philosophical nature of intelligence is beyond the scope of this thesis, but this brief foray into the metaphysical shows why it is difficult to classify intelligence let alone create an intelligent system.

2.2 Terminology

Without a shared language, communication cannot occur. This section introduces the common terminology used in the remainder of this document.

- *Agent* – any being(bird, ant, robot, or otherwise) participating in a swarm intelligent system.
- *Robot* – a mechanical device capable of responding to environmental stimuli by some predetermined set of behaviors
- *Sensor* – any device a robot may use to extract information from its environment
- *Tracking* – estimating the position of a selected object in space and time

2.3 Collective Robotics

Collective robotics extends the study of robotics to include interaction and communication between multiple robotic agents. Studies in collective robotics focus on the control strategies and communication paradigms to allow the cooperation of multiple robots.

2.3.1 Advantages

There are many advantages to using many robots versus a single robot. First, no matter how sophisticated, fast, strong, computationally powerful a single robot may be, its abilities are still finite. There is always a problem that exceeds capabilities of a single robot. A number of robots can be used to cooperatively accomplish tasks that are beyond the reach of a single robot. For example, multiple robots working together can transport a heavy object that no one could carry alone.

Economics also motivates the study and implementation of collective robotics. By using many cooperating less capable robots instead of more sophisticated individual robots, the overall cost of the system may be decreased. Once production reaches a point to realize economies of scale, the average cost per robot will drop significantly making robotics more attractive in more situations.

Cooperation between robots can also increase the efficiency and performance of multiple robots in certain tasks. A number of robots working together can perform a task more quickly or return better results than the same number of robots working independently. The level of performance enhancement is highly dependent upon the task being performed as some tasks are highly sequential and are not easily or are not capable of being done in parallel.

2.3.2 Autonomy

To be considered autonomous, the ability to determine the future course of actions must be at the robot level; the robot cannot simply execute a predefined sequence of operations. Mobile robots face many more challenges as they encounter an unknown and often dynamic environment as compared to stationary robotic manipulators, such as those on a factory assembly line. Mobile robots currently used for real world applications [48] have at least some form of human-aided control. A human operator can take over control and keep the robots from entering a possibly deadlocking state, such as getting stuck in a corner or crashing into a mountain. As shown in the representative graph below, truly autonomous, mobile robots have yet to accomplish any task beyond simple "toy tasks" such as two-dimensional pattern formation [42] and resource gathering [31], [32], [45].

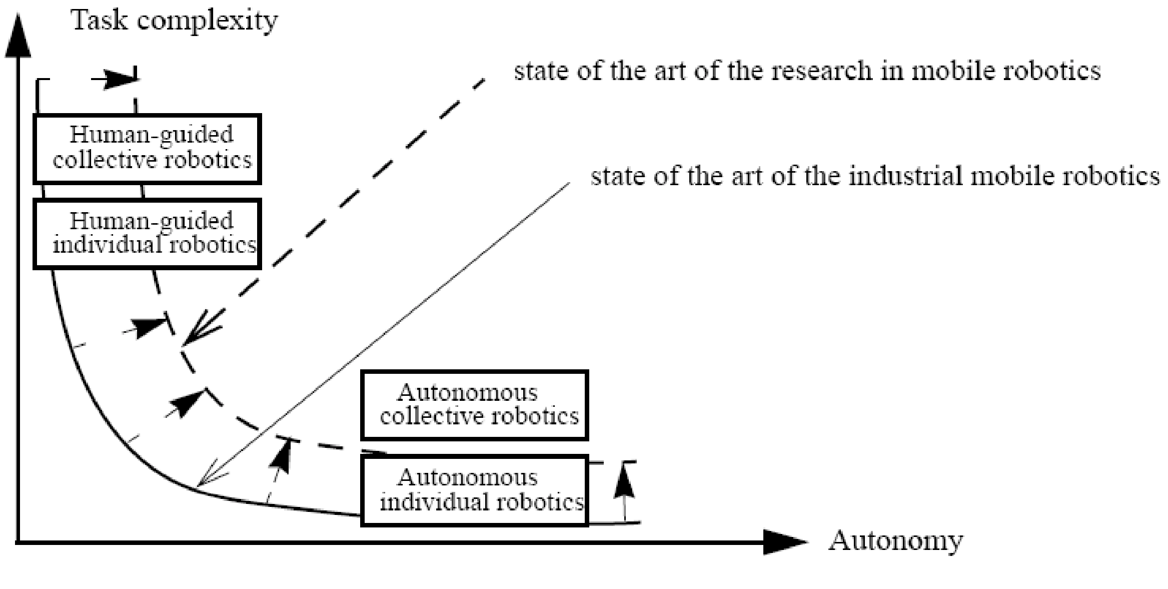

Task complexity
state of the art of the research in mobile robotics
Human-guided collective robotics
state of the art of the industrial mobile robotics
Human-guided individual robotics
Autonomous collective robotics
Autonomous individual robotics
Autonomy

Figure 1: truly autonomous robots lacking any human intervention have yet to perform tasks beyond the most simple

To cope with the challenges of remaining autonomous under these conditions, mobile robots have a dramatically increased number of sensors for tasks such as obstacle avoidance, image recognition, friend / foe detection, or global positioning. Adding multiple robots only adds to this problem as each robot must accomplish its goals without interfering with the goals of another. As the environment becomes more challenging and the sensory input increases, determining the optimal next action becomes more difficult. Much research [49] has been done in the low level optimization of robot movement and sensory response to ensure safe movement and increase searching efficiency. The aims of this thesis are to investigate control architectures and communication paradigms that are at a level above the basic control and to utilize the research already done in that area.

2.3.3 Control Architectures

There have been many novel approaches to the control of collective robotics. In this section the main categories of such architectures are summarized.

The first category group is the subsumption architecture developed by Brooks [11]. These are behavior oriented control strategies where Simple behaviors performed by Augmented Finite State Machines, are hierarchically organized so that more complex behaviors emerge. It uses low-level behaviors like responses to sensors that are called upon and build on each other. As the situation changes, the robot switches between behaviors. The ALLIANCE system [38] is an example of this type of architecture.

The motor-schema architecture proposed by Arkin [4] uses primitive behaviors (called motor schema here) as the subsumption architecture. The main difference is multiple behaviors can be active simultaneously and combined cooperatively. Multiplying the vector response of each motor schema by a gain, then summing and normalizing the result produces the higher-level behavior.

Action selection architectures use competence modules, primitive behaviors, and feature detectors, which deliver information about the environment and present state of the robot. Only one competence module is active at a time. The activation signal is spread to all competence modules based on the feature detectors' outputs and the state of the preconditioning competence module. Once the activation level of a competence module reaches a certain level, it becomes active and its activation level is reset to zero.

Many other original architectures and hybrids of the ones mentioned above have been developed. They are too numerous to include in this thesis, but it is worthwhile to have an overview of the main, prevalent architectures utilized today before detailing the one designed for this thesis.

2.3.4 Communication

In order for robots to work together, they must communicate. This section discusses the possible methods and their physical implementations by which the autonomous robots communicate. There are two main types of communication: implicit and explicit; the later is divided into two broad categories global and local.

Implicit communication occurs by altering the environment and reading information from it. For example, when an agent finds some location of interest it deposits a chemical signal, which draws other agents to it. This type of communication is also called *stigmergic*. The word "stigmergy" derives from the words: "stigma" which means exterior sign and "ergon" which means work. Together the meaning is "incitement to work by the products of work". By using this method the environment itself becomes a shared memory.

With explicit communication, messages are sent directly to a target robot, supervisor, to the entire group via a broadcast message. This method requires a dedicated physical channel, such as a radio or infrared link, through which to send these messages. The explicit messages often include more information than implicit communication such as sender and receiver information.

During global explicit communication, the physical link with limited bandwidth can become a major bottleneck as all agents require access to the one shared resource. Many research systems use global communication [3], [43], and have developed robot transmission protocols to try and overcome the bottleneck problem. Local explicit communication schemes have also been developed as an alternative to deal with the bottleneck while still retaining the increased information exchange of explicit message passing. Research has been done to develop novel hardware implementations and local communication protocols [29] such as creating communication chains between adjacent team members to convey message to the entire group [51].

2.3.5 Collective Tasks

As mentioned previously, not all tasks are appropriate for a system of mobile robots. Amdahl's Law [18] extends beyond computer architecture; there are some tasks that must be done sequentially and cannot benefit from parallel work. Tasks, which are well suited to groups of robots, are now listed. Many of the state-of-the-art research systems designed so far have sought an original approach to one of the following problems:

- *Exploration and Mapping* – Exploring and navigating an unknown area to obtain sensory information or an area map translates well to groups of robots. These applications do not usually require any modifications to the environments and do not require a high level of synchronization between the robots. Several collective control strategies have been employed to solve this problem including flocking [34], formation maintenance [20], relative position and uncertainty correction [41], and motion planning [2]. Swarm intelligence was used by Ichikawa and Hara [30] as well as Cohen [16] to implement environmental mapping systems. Both projects used a similar local communication scheme with other robots acting as beacons.
- *Resource Gathering* – Gathering resources is a natural extension of exploration. As the area is navigated, a targeted resource is collected and returned to a drop point. A pertinent real-world application of this problem is the discovery and removal or diffusion of landmines. Ant foraging strategies [8], [33], [46] have generally been used to control these gathering systems.
- *Collective Transportation* - Collective transportation is divided into two subcategories: strict cooperation where the heavy cannot be carried by one robot alone and loose cooperation where coordinated activity is not required but will accelerate progress. Werger and Matric [51] used chain formation mechanisms to study loosely cooperative transportation. While investigating strict cooperation, Boehringer [7] demonstrated that it was possible to push an object in a coordinated fashion without explicit communication among the robots. Each robot measured the torque over their wheels and extrapolated the reaction force that was being exerted on them and changed their behavior accordingly.

2.4 Swarm Intelligence

Swarm intelligence refers to systems which accomplish complex global tasks through the simple local interactions of autonomous agents. The control is completely distributed among the individual agents with no leader coordinating any of the activities. Swarm intelligence [33] is "a property of systems of non-intelligent robots exhibiting collectively intelligent behavior".

According to Beni [6], there is a quantitative definition of swarm intelligence. He claims swarm intelligent behavior occurs among *N* interacting agents only when *N* reaches some critical number N_c. *W(N),* the work achieved by *N* interacting agents, can be described by a function with a critical point at N_c. $W_o(N)$, the work achieved by *N* independent agents, is 0 for all values of *N*. Sugawara and Watanbe [46] generalized this definition to say that swarm intelligence occurs whenever $W(N) > W_o(N)$ or *N* interacting agents perform more work than an equivalent number of N independent agents. Performance gains through swarming occurs when a critical mass of agents come together and enter a positive feedback loop. For example, the defensive capabilities of a single bee are insignificant, but an entire swarm can protect the hive from most animal attackers.

2.4.1 Biological Inspirations

Ant colonies efficiently forage for food and build intricate nests with no single, controlling queen. Birds fly in formation without centralized leadership or explicit communication. Herd animals like cows do not follow one leader of the pack. These simple biological agents coordinate extremely complex behavior without any help from a global perspective or centralized controller.

How do birds flock and fly in a choreographed like fashion diving and swooping in unison? Trying to answer this question, Reynolds [40] created a "boid" model to explore the swarming relationship of birds. All boids follow three simple rules. First a boid cannot get too close to any over boid to prevent mid-are collisions. Second, a boid must copy the rest of the flock by averaging the other local boids' velocities and directions. Third, the boid must try to minimize exposure to the outside of the flock by trying to fly toward the perceived center. The simulation results closely resembled the behavior of real birds in flight. This experiment was one of the first to show how simple interactions could create a global control pattern.

It is worth noting the difference between biological inspiration and trying to emulate the behavior of biological systems. The point of this research is to glean the best aspects of biological societies and collectives and apply them to our technological problems. Creating a system that exactly mimics the behavior of biological collectives is an excellent simulator for biological research but not the best form of robotic control. The goal is to exploit the millions of years of natural evolution that has produced solutions to problems very similar to the technological ones currently faced by science.

2.4.2 Emergence and Self-organization

Swarm intelligence relies upon the emergent properties of its components to manifest itself. Emergence [33] is the process by which complex patterns form out of the interaction of simpler rules. To label an event as being emergent it should be hard to predict from a description of the lower level components. For example, it would be difficult to imagine the resulting flocking from seeing only a single boid and its rules. There has been much research into the fields of social science [23], physics , biology, and engineering [33], [9] concerning emergent properties.

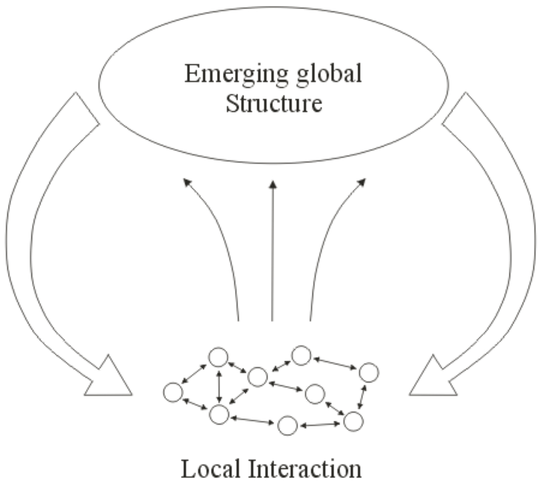


Figure 2: Global behaviors arise from local interactions

Self-organization is another term used to describe systems composed of discrete, individual components that create a global action through interaction. In chemistry, the term is used to describe reaction-diffusion system and autocatalytic networks [1]. It has been applied to sensor networks [14] and evolutionary computation [45].

2.4.3 Scientific Interests

There are many scientific motivations encouraging a number of varying fields to study swarm intelligence. First, biologists seek a deeper understanding of social insects and other animal societies. By studying the interactions that replicate such colonial behavior, they gain insight into the laws governing the natural world. Some have even conjectured that the human body [27] and brain [33] are swarm intelligent systems composed of simple interacting agents.

Second, swarm intelligence provides an interesting challenge from a complex systems point of view. Precisely and deterministically programmed robots interact with a noisy, dynamic environment. Can the robot – robot interactions or robot – environment interactions be predicted? Can they be controlled?

In relation to this thesis, swarm intelligence is being studied to advance the control of autonomous robots. It presents many new challenges and possible benefits to the field of robotics.

2.4.4 Advantages to Robotic Control

Swarming strategies offer a number of advantages to the control of multiple autonomous robots. Creating complex global behaviors with minimal communication and simple, individual intelligence is very attractive to system designers.

Because there is no centralized control, the decision making power is completed distributed among the robots. Destroying any one robot will not cause the system to stop functioning. Agent loss may decrease performance, but the system's goals can still be achieved. This makes swarming well suited for military operations and other harsh environments too dangerous for human beings where the probability of unit destruction is high. Also a leaderless system is highly scalable as the team members are not waiting on instructions from a single source.

It is hoped, that the swarming robots will be more reliable and repairable due to the simplicity and modularity of the design. This simple design can also lead to lower manufacturing costs as the parts are easier to mass produce and the sensors do not have to be of the same high level of quality. Also, because the individual robot control is simpler, the cost and time to design the systems could be lowered. The true impact of these monetary benefits is still a theoretical issue as no real world implementation of swarming robots currently exists.

In military applications, the stochastic non-deterministic movement of the robots lends some protection, as it makes them much harder for an enemy to target [5]. It is easy to observe the overall motion of a flock or swarm, but an individual's movement is seemingly random. The path of a single robot is not pre-determined, and thus its next destination is hard to determine.

2.4.5 Other Applications of Swarm Intelligence

The mechanisms behind swarming behavior are not limited to robotic control. They can be applied to many distributed computational problems. Ant Colony Optimization (ACO) research draws inspiration from ant colonies and is used to solve discrete optimization problems. APO has been used to solve the traveling salesman problem, the sequential ordering problem, and the quadratic assignment problem [21], [25], [17]. Swarming algorithms utilizing the ant behaviors are being used to more quickly establish routing schemes in ad hoc wireless networks [44].

Chapter 3

Surveillance System Design Overview

This chapter introduces the surveillance system designed to take advantage of swarm intelligent properties. It presents the design criteria of the system and ant foraging strategies that inspired the swarming design. The communication weakness present in current foraging-based systems is explained and a solution is proposed. Finally, the final system design is detailed.

3.1 Design Criteria

The goal of mobile surveillance system is to rapidly track the position of a known mobile target in a previously unknown area. Because the area under surveillance is unknown and possibly dangerous, manually installing stationary sensors is not feasible and the sensors must be mobile. The area is also assumed to be too large for a single agent's sensors to find a target effectively.

Another design consideration is the high probability of the loss of an agent. This system is targeted toward military applications where a hostile force will be actively trying to disrupt it or harsh environments too dangerous for humans where environmental hazards may cause agent loss. The system cannot depend upon the presence of any one robot to be a communication hub or leader; therefore, the control must be completely distributed.

The communication scheme is another important design criterion. To accelerate the searching, the robots must cooperate and somehow communicate. The system must be extremely scalable as the size of the area can be large. A centralized communication system would eventually create a performance bottleneck limiting the number of robots and creating a single point of attack.

Note that the system is constrained to situations with specific conditions. As mentioned earlier, only some tasks are appropriate for collective robots and only a subset of those are appropriate for swarm intelligent systems. The central purpose of this research is to study swarm intelligence and thus a task was chosen that could most benefit from a swarm intelligent system.

3.2 Ant Foraging Strategy

Ants are an excellent example of a natural swarm intelligent system. Contrary to popular belief, there is no centralized queen ant controlling all of the worker ants. The ants are all individuals responding to their own sensory information and pheromone signals. Pheromones are chemical deposited by ants, which other are capable of detecting and reacting to; they are the basis for ant communication.

One of the main tasks of ants is foraging, locating food and transporting it back to the nest. Ants demonstrate the following behavioral rules: They wander randomly from the nest in search of food. If they find a piece of food, an attractant pheromone is deposited and a trail of pheromone is left back to the nest. If another ant senses the pheromone, it follows the pheromone up its concentration gradient to the source. As more ants find the food, the pheromone concentration at the source is raised higher, attracting more ants. This positive feedback loop produces a swarm of ants to quickly transport the food source.

The ant foraging strategy is inspirational for a distributed control scheme to collect information about a given target. Many research projects [46], [36], [34], [32], [31] have used ant foraging as the basis for control of resource gathering systems. Surveillance information and sensory data can easily be substituted for a food source. Instead of excreting a pheromone to use the physical world as shared memory, the robots emit electromagnetic signals, using the electromagnetic field as their shared memory. The local communication and lack of a centralized controller makes this scheme ideal for the constraints listed in the previous section.

3.2.1 Communication weakness

The efficiency of the ant foraging strategy is unparalleled in the natural world. Ants' resiliency to single agent destruction makes them the perfect house pest, but many years ago, humans discovered a way to turn the ants' own emergent positive feedback against themselves. Attractant poisons release a similar pheromone to that which is released when ants find a piece of food. Because the ants blindly follow this excitatory signal, the ants are drawn toward the poison as though it were another ant signaling a food source. The ants efficiently poison themselves and their numbers drop below the critical number proposed by Beni [6] needed to reach their swarming potential.

Figure 3: Poison disrupts ant foraging.
*many ants were harmed in the writing of this thesis

The introduction of the poison created a new emergent behavior, which caused the entire system to enter a deadlock state through near-total agent destruction. In the case of the system designed, an erroneous or malicious signal would lead to all robots uselessly congregating in a single area. The uncontrolled, positive feedback cycle that lead to the accelerated resource gathering also caused the system to collapse when an unexpected condition arose. Without a way to determine if a signal is trustworthy, a local intrusion can cause global disruptions. All of the research reviewed assumed that all communication was trustworthy and that no agent was lying.

This type of untrustworthy communication is possible in the hostile and dangerous environments, which the system designed in this thesis is intended for. The miscommunication could occur by two means. First, a malicious agent, possibly a military enemy trying to conceal the search target, may spread false signals to attract the agents to a location of little interest. The communication might be encrypted, but the agent loss rate is expected to be high, so the enemy may have enough hardware to compromise the security measures or simply capture and use a robot. Also, the communication is expected to be simple: radio signal or light flash that local agents can detect and move toward. Secondly, a robot may malfunction once it is damaged in the harsh environment, sending false data along the secured communication channel. Without some sort of safeguard against this problem, the surveillance system runs the risk of entering a deadlocking or extremely inefficient state.

3.2.2 Communication Feedback Threshold

To solve the problem presented in the previous section, a communication threshold scheme is proposed and implemented in the test system. The cooperative behavior is limited in the following two ways:

- A robot will only respond to an external signal, supposedly from another robot, if it passes a threshold value. This threshold is based on the distance from the source and the time since a target was last spotted. If the signal is too far away or if a target was just spotted locally, the robot will not follow the signal
- If a robot follows a signal for a set amount of time and finds nothing, it will assume there has been a communication disturbance and enter an isolation state for a time. In the isolation state, the robot acts just like an independent agent not responding to external signals.

The second limit prevents the system from falling into a deadlock state by not allowing the robots to all stay grouped at the signaling point. If all of the robots reach the false signal and enter the isolation state, the worst that can happen is the system performs at the level of independent agents. The first limit is a preventative measure to try and stop all the robots from following a signal. If all robots follow a signal, part of the area will be unmonitored and a target could move undetected.

Not allowing part of the robots to participate in the flocking behavior can increase efficiency as well as prevent deadlock. It is proposed that there is an optimal robot density for gathering data. Below that density, not enough data will be gathered and chances to measure the target will be lost. Above that optimal density, the robots begin to crowd one another, effectively creating traffic jams. Previous work has been done to use the external communication, based on the chirping of frogs, to regulate flock density, but without an internal threshold that system is susceptible to the miscommunication deadlocks already mentioned [28]. Chapter 6 presents the data pertaining to efficiency changes due to the introduction of the threshold.

3.3 Final Design

The surveillance system design draws upon the previous work of ant-foraging-inspired systems [46], [36], [34], [32], [31] and adds a communication threshold to prevent a deadlocking state and improve efficiency. This creates a mobile surveillance network with scalable communication and decentralized control, which fits the criteria as stated in Section 3.1. Figure 4 presents a state diagram that summarizes the individual robot behavior. A description of each state is as follows:

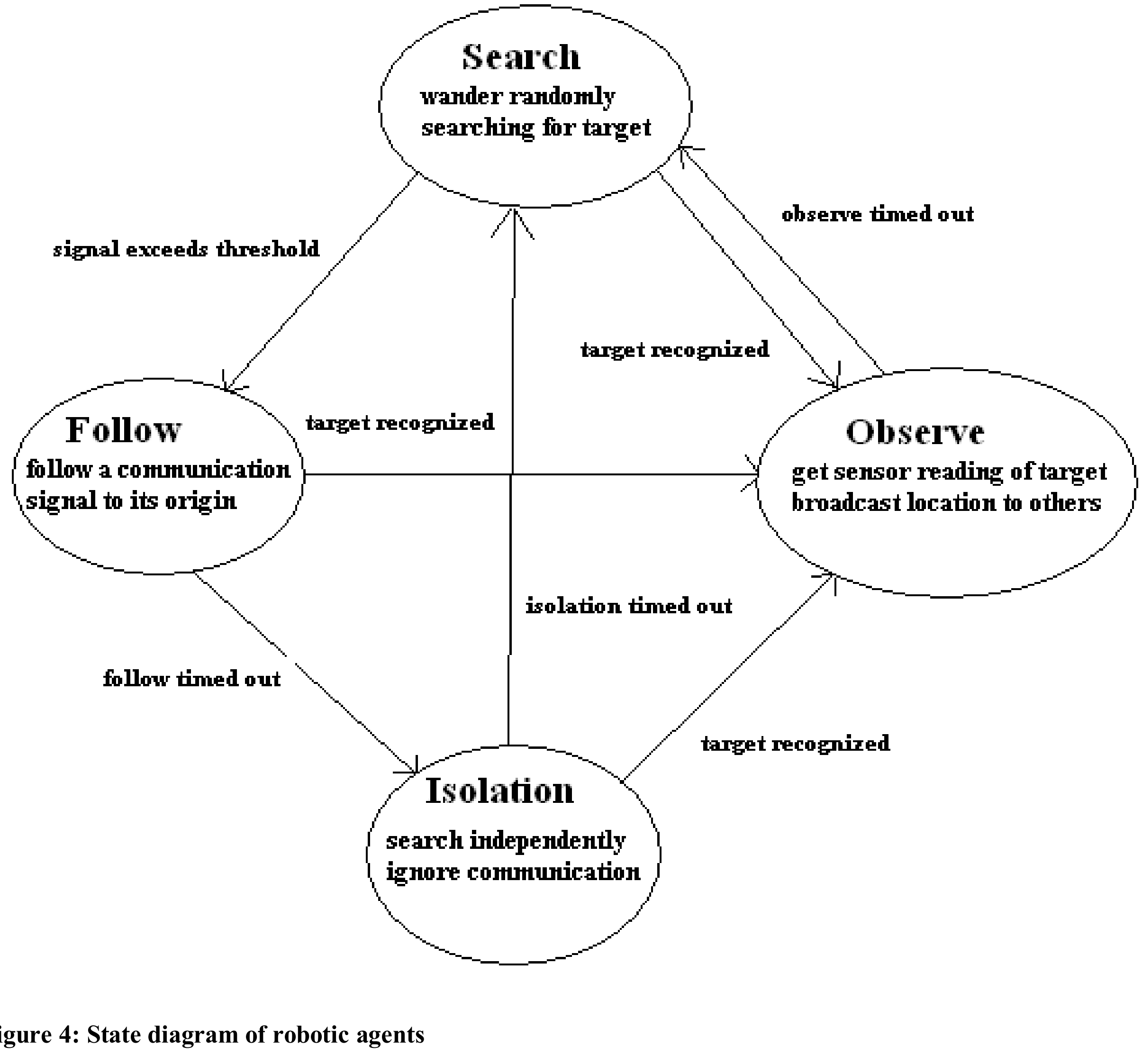


Figure 4: State diagram of robotic agents

- *Search State* – While searching, the robot moves forward then makes a random rotation between –45 degrees and 45 degrees. It is looking for the target and trying to cover as much ground as possible
- *Observe State* – Once a target has been sited, the robot stops and slowly scans the area trying keep the target within its field of vision. A signal is also sent to all nearby robots to attract them to that location in hopes of increasing the accuracy of the readings.
- *Follow State* – If an external communication signal reaches the threshold as outlined in the previous section, the robot will follow that signal to its source. If the target is not found, the communication is assumed to be either false or outdated, and the robot operates in an independent mode.
- *Isolation State* – After having followed a signal with no results, a robot will behave as thought it were independent and no longer respond to external communication for a period of time. If a target is sighted, it will enter the observing state and again cooperate with the other robots.

The following next-state equations describe the robot behavior and are derived from the state diagram:

$$S' = -aS + \frac{1}{x} O + \frac{1}{y} I - t\, S{*}O$$

$$O' = aS - \frac{1}{x} O + g\frac{v}{d} F + aI$$

$$F' = t\, S{*}O - \frac{v}{d} F$$

$$I' = (1 - g) \frac{v}{d} F - \frac{1}{y} I - aI$$

Where *S*: the number of robots search, *O*: the number of robots observing, *F*: the number of robots following a signal, *I* the number of robots in isolation mode. The parameters in the above equations are defined as follows:

a: probability of finding a target independently,

x: observe state timeout,

y: isolation state timeout,

g: probability of finding a target by following a signal,

v: average velocity of a robot,

d: average distance to signal source,

t: the probability that a signal will cross the communication threshold.

Each line in the state transition graph becomes a term in the above equations as robots enter and leave states. If a term, such as *aS*, is subtracted from one next state equation, it must be added to another next state because the total number of robots is conserved.

The four-state design above was an improvement upon the general three state design used in previous research [46]. Figure 5 shows a design without an isolation state to halt the negative effects of miscommunication.

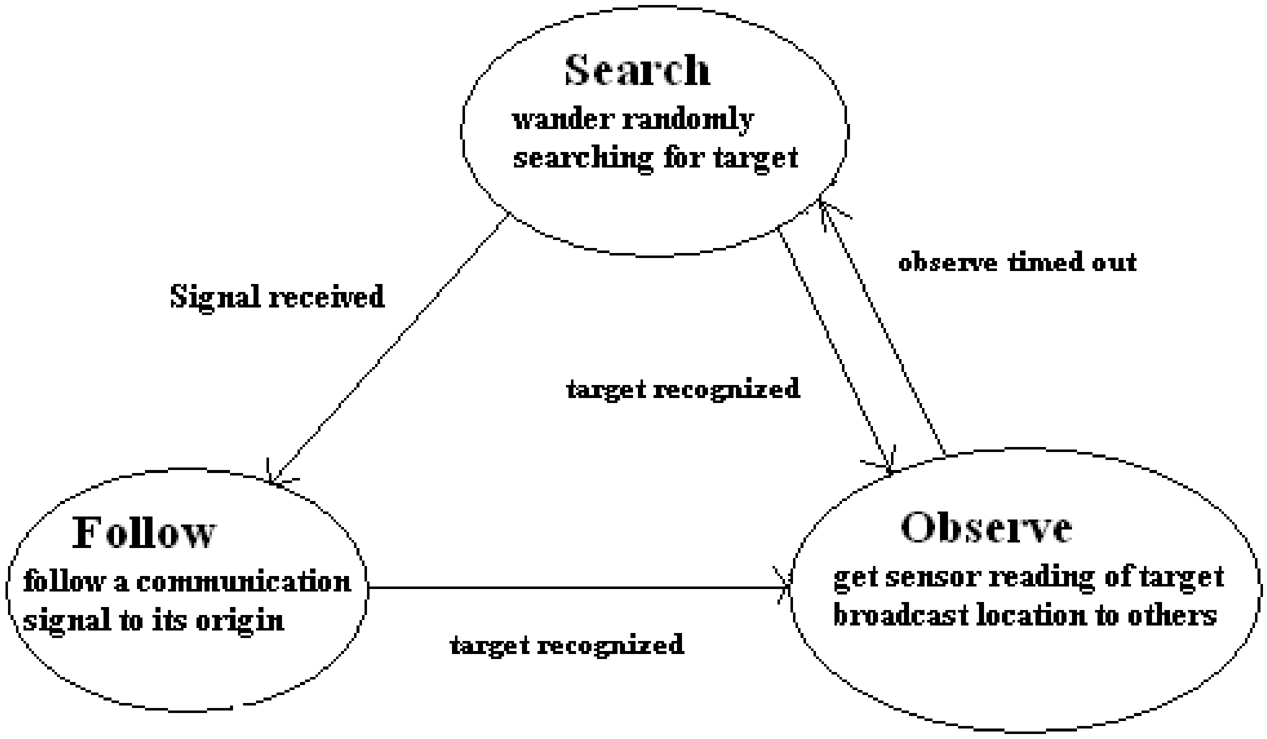

Search
wander randomly searching for target
Signal received
observe timed out
target recognized
Follow
follow a communication signal to its origin
Observe
get sensor reading of target broadcast location to others
target recognized

Figure 5: the three state design that is the basis of previous collective, swarming systems.

This state transition diagram produces the following equations using the same variables as defined above.

$$S' = -aS + \frac{1}{x}\, O - t\, S^*O$$

$$O' = aS - \frac{1}{x}\, O + g\,\frac{v}{d}\, F$$

$$F' = t\, S{*}O - \frac{v}{d}\, F$$

Using these equations, the communication vulnerability of previous systems can be shown mathematically as follows: When a false communication emitting an attractant signal to a location containing no targets enters the system, it can be modeled by adding one to *O*. Now *O* can be no less than one. Also let us assume this communication is strong enough to be received by all robots; t, the probability of detecting a communication signal, becomes one. The term *t S*O* becomes *S*, and the above equations become

$$S' = 1/x\, O - S$$
$$O' = aS - 1/x\, O + g\, v/d\, F$$
$$F' = (1-a)S - v/d\, F$$

Now, all searching robots that do not encounter a target will enter the following state. As soon as a robot enters the search state, it will begin following a signal that leads to nothing. With no robots searching, eventually only the area at the source of the miscommunication will be under surveillance.

In contrast, the four simple states shown in Figure 4 and the rules governing their actions lead to an emergent system capable of cooperatively tracking mobile targets without entering a deadlock state. The isolation state is what allows this system to avoid the deadlocks possible in previous designs [46], [36], [34], [32], [31]. The next two chapters detail its implementation in simulation and physical robotic hardware.

Chapter 4

Simulation

This chapter describes the system simulator and its usage to obtain the simulation results. The free software package is discussed along with how the system designed was implemented.

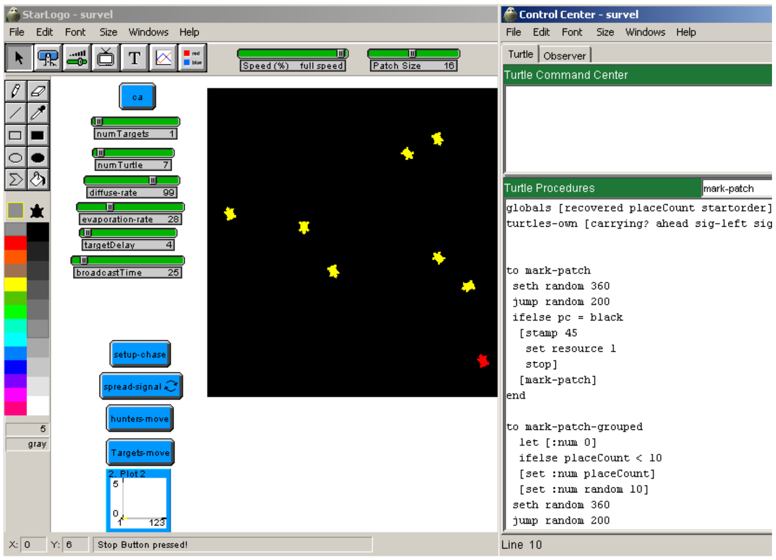
4.1 The Need for Simulation

In most projects, simulation is simply a tool to expedite hardware or process design. In this thesis, simulation allows for modeling situations that are currently beyond the resources available. Virginia Tech has six robots and large rooms to run experiments. This system is designed to scale up to hundreds if not thousands of robots over a much greater area. Also the logistics of maintaining and testing a fleet of robots that numerous exceed the capabilities of the limited number of people working on this project. Due to economic and personnel limitations, simulation is the only currently feasible option to do testing on this scale.

4.2 StarLogo

StarLogo was used as the simulation package for this thesis. StarLogo is a simulation environment designed for studies in decentralized systems, “systems that are organized without an organizer, coordinated without a coordinator” [47]. The simulator is free software produced by the Media Laboratory and Teacher Education Program at MIT. The software is designed to foster new ways of thinking about decentralized control.

StarLogo is a descendent of the classic Logo programming language. With traditional versions of Logo, the user gives commands to graphical “turtles” on the display to create drawings and animations. StarLogo extends this basic system by controlling the behavior of thousands of turtles in parallel. In addition, StarLogo makes the turtles' world reactive to turtle actions and influence turtle behavior: the user can write programs for thousands of "patches" that make up the turtles' environment. Turtles and patches can interact with one another -- For example, the user can program the turtles to be sensitive to the qualities of the patches around them, and alter their behaviors based on what they sense in the patches below. StarLogo is designed with research in Artificial Life projects in mind.



```
globals [recovered placeCount startorder]
turtles-own [carrying? ahead sig-left sig


to mark-patch
 seth random 360
 jump random 200
 ifelse pc = black
  [stamp 45
   set resource 1
   stop]
  [mark-patch]
end

to mark-patch-grouped
  let [:num 0]
  ifelse placeCount < 10
  [set :num placeCount]
  [set :num random 10]
 seth random 360
 jump random 200
```

Figure 6: StarLogo has an integrated development environment for creating simulations

The MIT group used Java and YoYo, a language they created, to implement the simulation environment. YoYo is a variant of the Logo running on the Java Virtual Machine that allows the user to create Logo/Java hybrid programs for distribution and use on the web.

4.3 Implementation in Simulation

StarLogo was designed specifically for investigating swarm intelligent systems similar to the one presented in this thesis. The StarLogo implementation was fairly straight forward using its assembly-language-like instruction set to model the robots movement and behavior.

As mentioned in the previous section, the environment is composed of connected patches, which are programmed separately from the individual agents. These patches are used to implement the communication between agents. Each patch has a "signal" real number value associated with it. An agent can read or alter the signal value of the patch that it is currently on. A single thread is created for each patch and the signal value diffuses from higher to lower concentrations to the surrounding patches. This mimics the spread of a biochemical or electromagnetic source. The patches become the shared memory from which all the agents read communication information from one another.

```
;function – follow-signal
;set heading in direction with greatest value of patch
;variable signal
;;;;;;;;;;;;;;;;;;;;;
to follow-signal
 setahead next-signal
 rt 45 ;measure to the right 45 degrees
 setsig-right next-signal
 lt 90 ;measure signal to the left 45 degrees
 setsig-left next-signal
 rt 45;return to initial heading
;if the signal is stronger to the right turn right
 if (sig-right > ahead) and not (sig-right < sig-left)
   [rt 45 stop]
;if the signal is stronger to the left turn left
 if (sig-left > ahead) and not (sig-left < sig-right)
   [lt 45]
end

;function used by follow-signal to the determine the
;value of the patch variable signal in front of the
;turtle
to next-signal
 output signal-at dx dy
end
```

Figure 7: Sample StarLogo code – This snippet of code is what causes a searcher to head toward areas of greater signal strength

To implement the state machine described in the previous section, the agents continuously run functions that correspond to the states. For example, while in the search state an individual turtle loops through a search function that breaks into an observe function when a target is sighted. The implementation of each state is detailed below:

- *Searching*: A simple user-created function called "wiggle" provided the main random movement for the searching agents. This function caused an agent to turn randomly between –50 degrees and 50 degrees and then move forward one patch. While searching and a target is in front of the agent within two patches, an agent will enter the observe state. If the value of the attractant signal variable, a unit-less real number representing signal strength, is greater than .1 in the current patch, the agent enters the following state.

- *Observing*: While in the *observing* state, an agent sets the attractant "signal" in that patch to 100 to attract other agents. The agent stops moving briefly and tries turning to keep the target in front of it.
- *Following*: Once in the *following* state, the agent triangulates the source of signal by measuring the signal at the patches ahead, to the right, and to the right. It then heads forward toward the direction of highest signal for one patch. This process is repeated until the target is sensed within two patches ahead of the agent or the *following* state threshold times out.
- *Isolation*: If the threshold times out, the agent is in the *following* state without finding a target for a specified amount of time, the *isolation* state is entered. The isolation uses the same wiggle function as described in the search state, but it does not respond to the signal variable.

Chapter 5

Physical Robotics

This chapter describes the physical implementation of the prototype surveillance system designed. The hardware and software from Evolution Robotics used in the implementation, [22] are detailed. Also, the specific details of implementing the system using these products are covered.

5.1 The need for Physical Implementation

The previous chapter explained the need for simulation because of the limitations expected in laboratory experiments due to a limited number of robots. A physical implementation consisting of only a few robots is still necessary to demonstrate that the system is indeed feasible using current technology. Also, robot density, the number of robots compared to the area size, is a more accurate description of the system size. Because few robots are used in a smaller area and the system is designed to be highly scalable, tests on a small scale may be extrapolated.

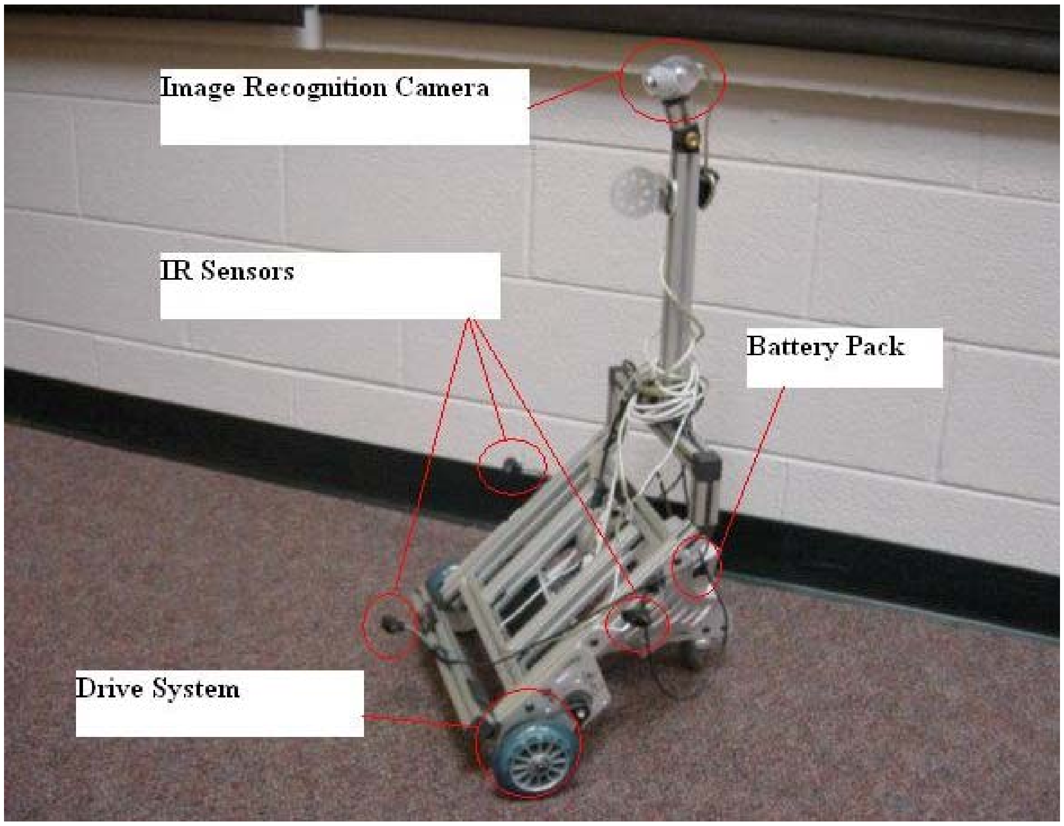


Figure 8: The main components of the ER1

5.2 ER1 Personal Robotic System

The ER1 is a low-cost, entry-level robotic system featuring sensory hardware backed up by professional grade software for image recognition and robot control. Any consumer laptop is used as the controlling "brain" to control a mobile, configurable chassis carrying IR and vision sensors. By using a standard laptop as the control platform, the user is able to develop more complex programs without being limited to a smaller, less powerful custom processor. The ER1's cost and flexibility make it ideal for a research setting.

5.2.1 Drive System

The ER1's drive system consists of two independently driven wheels, allowing it to rotate and change direction in place. A third wheel is un-motorized and simply provides balance. The drive system is rated to carry 20 lbs. of payload. There are sensors that track the rotations of the motorized wheels to provide positional information. On rough surfaces, the wheels may slip and lead to errors in the perceived position over time.

5.2.2 Image Recognition

The ER1 uses a 640 X 480 USB camera for image recognition. The vision software is quite good and can detect an object with many features between ten and fifty inches away about 80% of the time. The robot's angular velocity had to be nearly zero to reliably

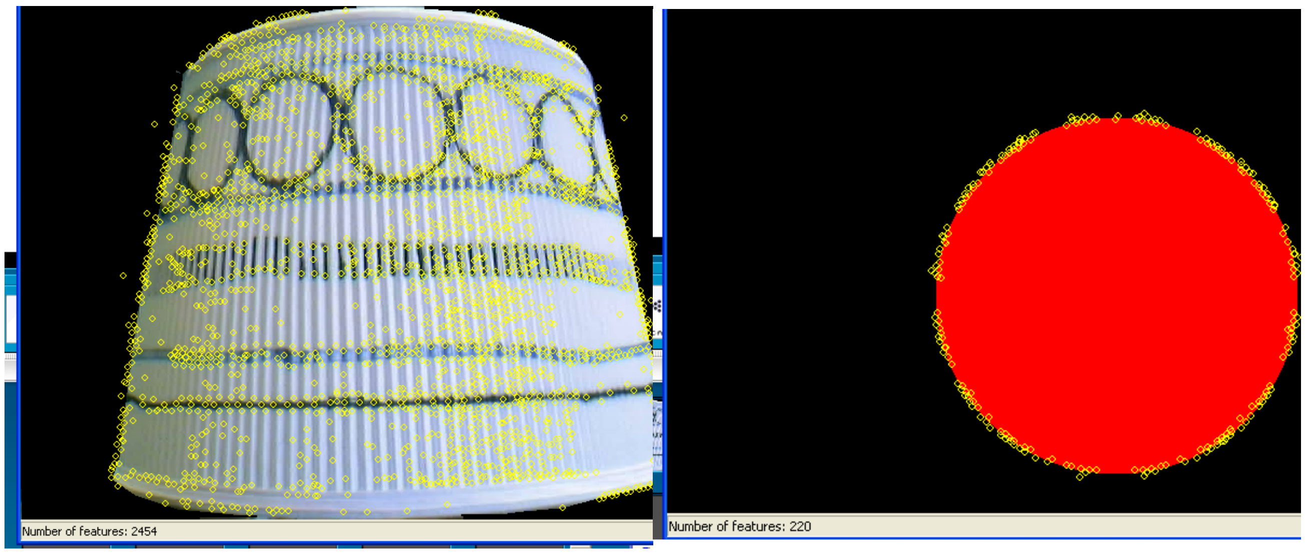


Figure 9: The red circle may be more recognizable to the human eye, but the repeating patterns on the lampshade provide 2454 recognizable feature where as the bright red circle only has 220 features.

recognize an object as lateral movement tended to blur the image and distort the object.

To be effectively recognized, an object must have many features the vision tool can recognize. For example, it is much easier to detect a dollar bill with many unique, intricate patterns than a red balloon that is just a circle to the vision tool. In this experiment, a lampshade with repeating symmetrical patterns around it was used as the target that all other robots were searching for. The patterns provided additional features for recognition and the vertical symmetry of the

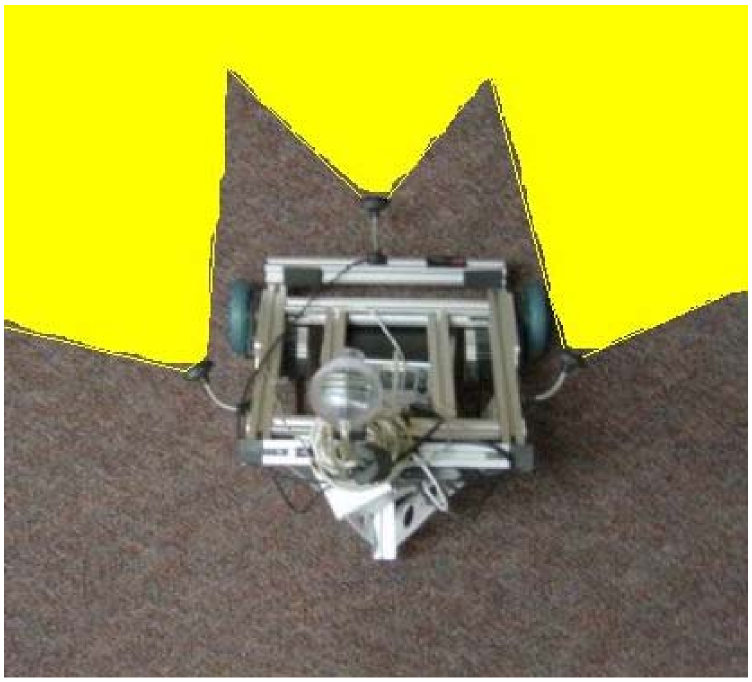


Figure 10: Estimated range of IR sensors. The ER1's back is left blind

shade allowed it to be recognized from any angle of approach.

5.2.3 IR Sensors

There are three infrared sensors used for obstacle avoidance. They are very precise and can detect flat surfaces such as walls reliably, although black surfaces tend to be sensed less reliably as they are not as reflective. Unfortunately, the other ER1 robots are composed mainly of empty space and do not register as well on each others' IR sensors. This leads to frequent robot collisions in close quarters. Also, each robot only has three sensors, one for the front and each side. This leaves the back unmonitored, which is especially problematic when the robots have to back up blindly during obstacle avoidance.

5.2.4 Battery Power

The battery pack allows the ER1 to operate autonomously without any tethers. A full charge allows for approximately 1.5 hours of operation.

5.3 ERSP

ERSP 3.0 is a robotic development platform also developed by Evolution Robotics [22] for use with any robotic product, although it comes with setup files for the ER1. ERSP speeds and simplifies development by providing critical infrastructure, core capabilities, and tools to designers. The software comes with a number of pre-built functions and APIs to facilitate vision, navigation, and environmental interaction. The package also includes GUI-based tools for "drag and drop" development of concepts and applications. ERSP is hardware and OS independent allowing developers the flexibility to choose components that are best for the situation.

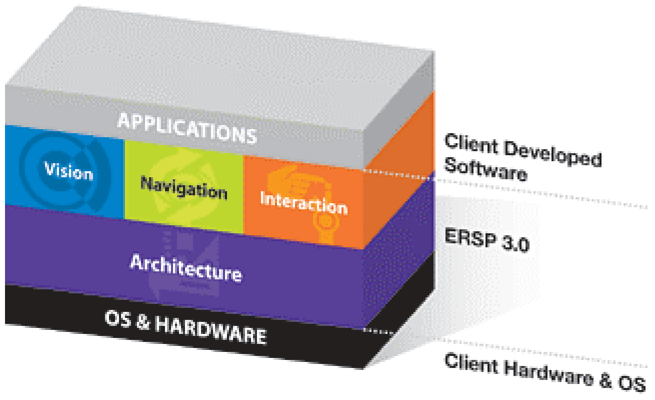


Figure 11: ERSP Application Framework (from [22])

The system architecture of ERSP is modularized into three main layers with well-defined interfaces between them. The lowest level is for hardware management, the middle level is for behaviors used by more complex applications, and the highest layer is for goal-oriented tasks. The three layers are:

- *Hardware Abstraction Layer (HAL)* – The HAL removes hardware and operating system dependencies between the robot and controlling application. It communicates directly with the physical hardware using XML configuration files. This layer provides portability of programs so that the same code can run on future generations of robot hardware.
- *Behavior Execution Layer (BEL)* – The BEL abstracts all the hardware details into modules used for sensing, decision-making, and autonomous action. A behavior can be thought of as constantly processing some input data and producing an output based on that data. For example, continuously reading image data from the camera and outputting any recognized objects is a behavior. Using the GUI Behavior Composer, behavior networks, composed of connected behaviors, can be created that continuously process information and respond to stimuli.
- *Task Execution Layer (TEL)* - The TEL provides goal-oriented functions to be used at the highest application level of programming. Commanding a robot to navigate through an area, locate a soda can, and pick it up would be an example of a task. This level is designed to be easy for programmers with experience writing in the standard procedural style.

These layers form a hierarchy from low to high levels of abstraction. There can also be communication between these levels and higher layer can be built from components of a lower level. Behaviors are wrappers around specific hardware function. *Primitive tasks* can be created by embedding a behavior network inside of a task. Tasks can communicate with one another by passing *events* amongst themselves. By using server and client behaviors derived from the malleable behavior class, events and other data can be passed across a network connection to a task on a separate computer.

5.4 Implementation details

This section describes how the surveillance system was implemented using the ERSP software and ER1 hardware. The ER1 hardware provides the mobility and sensors needed to implement the design. The IR sensors are used for obstacle and collision avoidance. The image recognition system is used to identify and track targets.

To implement the communication system, a TCP network connection is shared between all the robots. When a robots identifies a target and enters the observe state, it sends a message containing its position to all robots. Ideally we would want a simple signal like an electromagnetic pulse that attracts robots. Unfortunately implementing a signal emitter and detector was beyond the resource and time constraints of this project. To simulate a limited signal, robots only respond to the message if the [X, Y] coordinates of the message's sender are within a specified distance. Normally this shared connection would create a bottleneck as discussed in Chapter 2, but this implementation only has five robotic agents at a maximum and the connection can handle the traffic without noticeable delays.

The built-in functions provided with ERSP handled the lower-level interfacing with the hardware such as motor control and reading from the IR sensors. This allowed for much high-level design. The greatest challenge with implementing the system was creating the state machine as described in Chapter 3. ERSP excels at creating robots that respond to environmental stimuli, but changing that response behavior based on an overall system state was not initially apparent.

After much trial and error, the following scheme was developed. The network communication, image recognition, and movement control are all placed in their own continuous tasks and run in parallel. Creating a new task is synonymous with creating a new thread. In this way the external sensors and movement can be changed during state changes without interfering with each other. A single main control task takes information from all the other tasks and based upon that data and its current state decides the next state, which in turn affects how the other tasks behave. Figure 11 depicts a diagram of an overview of the continuous tasks and the information passed between them. Having a single control task consolidated the control logic and allowed all state information to be stored in a single location. The inter-task communication is accomplished using events, which signal state changes or major sensory events such as recognizing the target.

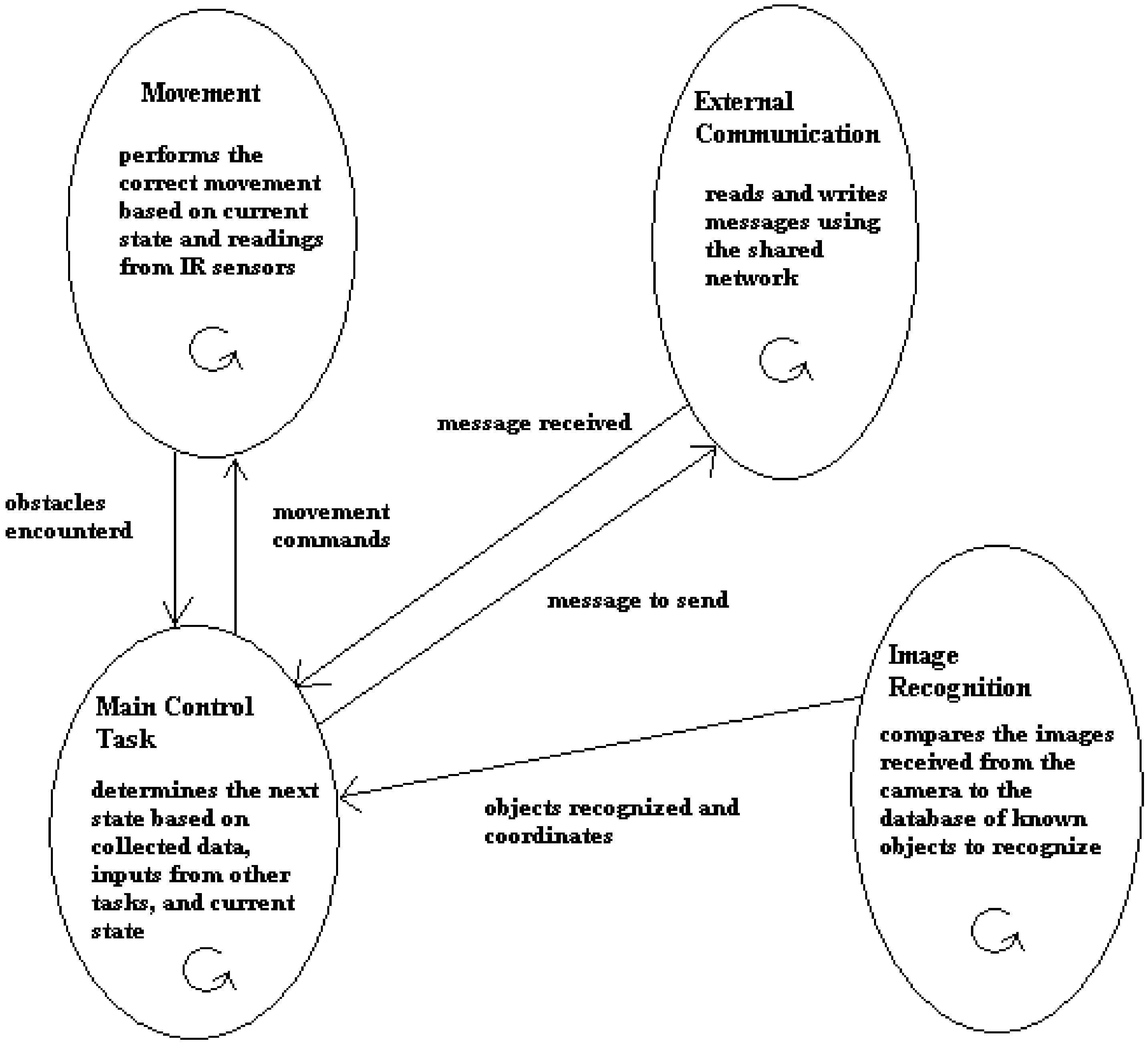

Movement
performs the correct movement based on current state and readings from IR sensors
External Communication
reads and writes messages using the shared network
message received
obstacles encounterd
movement commands
message to send
Image Recognition
compares the images received from the camera to the database of known objects to recognize
Main Control Task
determines the next state based on collected data, inputs from other tasks, and current state
objects recognized and coordinates

Figure 12: Task Overview for controlling the ER1

Chapter 6

Results and Analysis

This chapter presents the results obtained in both simulation and physical hardware. The results are analyzed individually and collectively.

6.1 Results Comparison

Because the study of swarm intelligence is a relatively young field, comparing results with past research was difficult. There is no standard benchmark or problem that all systems use for evaluation. Also, many of the previously designed systems use custom hardware, which changes the dynamics of the tests because the robots may be faster or more agile. All the past research surveyed used independent robots as the baseline and compared the efficiency and/or speed gains from cooperation. In this way, it is not the raw speed and tracking results of a cooperative system that are of interest but instead the amount of improvement over the same system using a comparable number of independent agents.

As mentioned in Chapter 5, the sensor technology, IR sensors and image recognition, used here is fairly rudimentary. Comparing our physical implementation to a current surveillance system with more advanced sensing and detection capabilities would be more of a comparison of the sensor quality than of the collective architecture. Also this system is designed for situations where current, stationary and centrally controlled systems are not feasible, so comparisons against them would be inappropriate.

6.2 Result Calculations

To gauge the effectiveness of the system, the positional certainty of the target's location is measured. To do this, the target's position is recorded at 1 second intervals during operation. Each time the target is spotted, its perceived location is recorded along with the time of the spotting. At every time step, the distance between the perceived and actual position is calculated using the following formula:

$$Distance = \sqrt{((X_{actual} - X_{preceived})^{2} + (Y_{actual} - Y_{preceived})^{2})}$$

Because the angle between the perceived location and actual location is still unknown, this distance represents the radius of a circle of location certainty. The area of the certainty region is calculated with the following formula:

$$Certainty\ Area = \Pi * Distance^{2}$$

To obtain the final measure of effectiveness, the certainty measure is compared to the total area as follows:

$$Percentage\ of\ area\ known\ to\ contain\ target = Certainty\ Area\ /\ Total\ Area * 100$$

This measure allows for comparisons across multiple area configurations as the area size is accounted for.

6.3 Simulation Results

This section presents the results obtained from simulation using StarLogo. All simulations were run on a 1.8 GHz Pentium 4 processor with 512 MB of RAM. Two sets of tests are run. The first uses numbers of robots ranging from 25 to 250, which is currently infeasible in hardware. The second test mimics the physical tests by only testing one to three searching robots in an area proportional to that used in the laboratory.

6.3.1 Simulation Test Procedures

The following procedures were used to make the simulate results more accurate and easily repeatable.

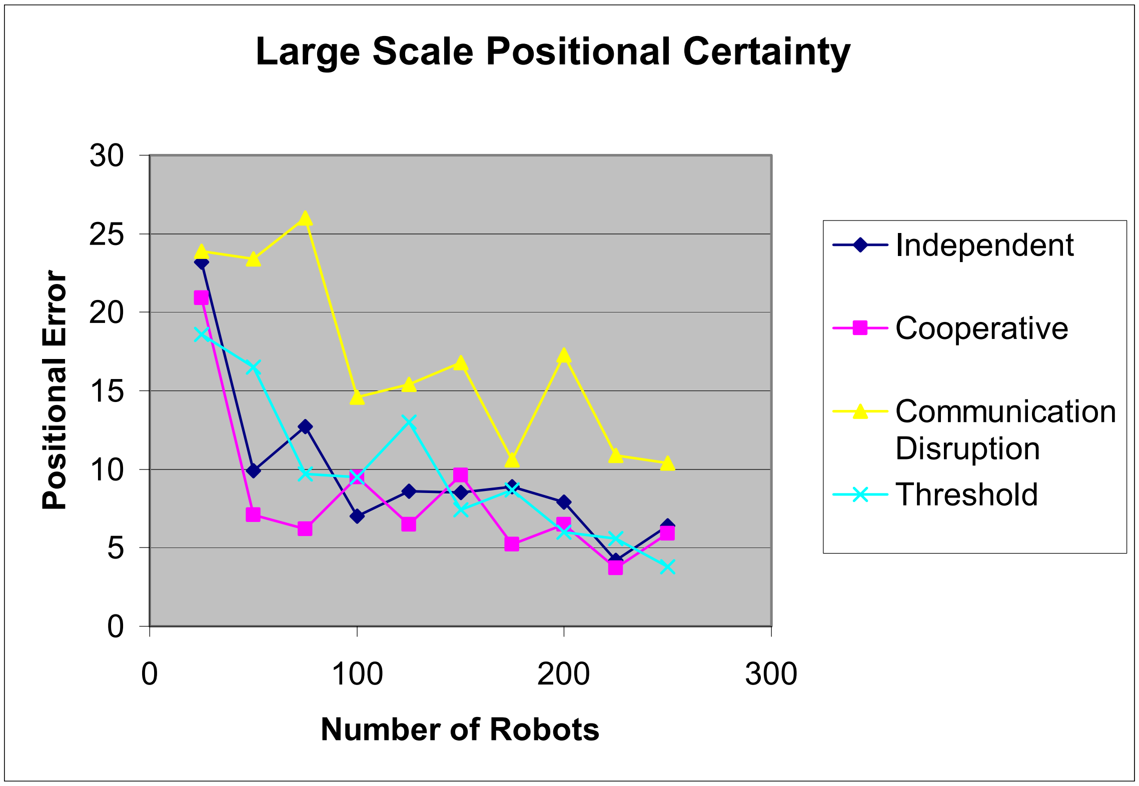
- Time – Because the running time of StarLogo is dependent on the processor making the computations and the current load on that processor, real time was not a suitable way to measure test duration. Each test allows the target to make 450 movements; the same amount of work is done regardless of the system configuration running the tests. This movement limit corresponds to the 300 second time limit in the physical tests because at their current speeds an ER1 can travel approximately 1.5 times its length in a second.

- Collisions – To model the disadvantages of robot-to-robot collisions, delays are added to the robot movement. When two or more robots occupy the same patch, they wait for .1 seconds and then move past one another.
- Area – A 12 patch by 12 patch area was used to mimic the physical hardware. Each table used as a barrier in the hardware testing was approximately four robot lengths wide. The simulation scale is one patch to one robot length.

6.3.2 Large Scale Results

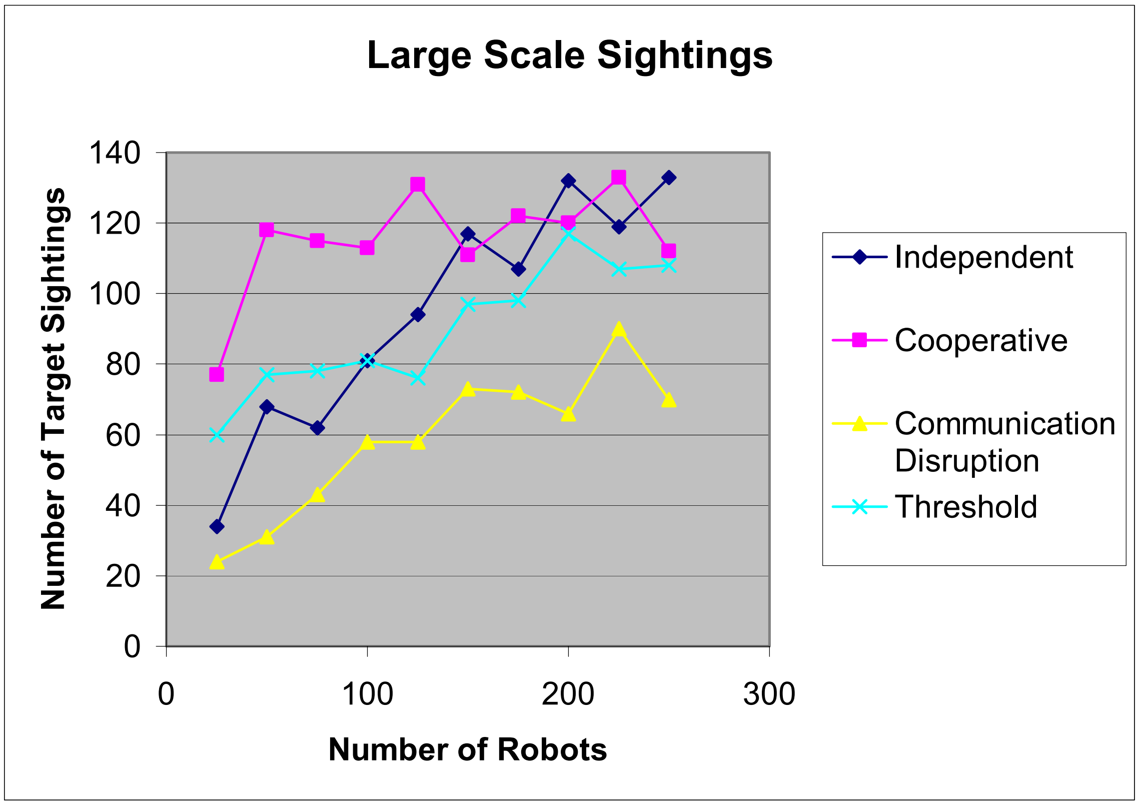
These simulations model systems with robots numbering far beyond the current limitations of hardware. The results show the potential of the designed system as the available robotic resources increase. Positional certainty and the number of sightings are reported.

Positional Certainty – Positional certainty is the percentage of the area, which the target's position is narrowed down to. Lower percentages represent a more accurate placement of the target.



As seen in the graph above, the positional certainty of the tests involving a communication disruption is significantly higher than the other three tests. The cooperative behavior without any disruption yielded slightly more accurate results than the other tests.

Number of Sightings – The number of sightings reports the number of chances a robot had to take a measurement of the target's position.

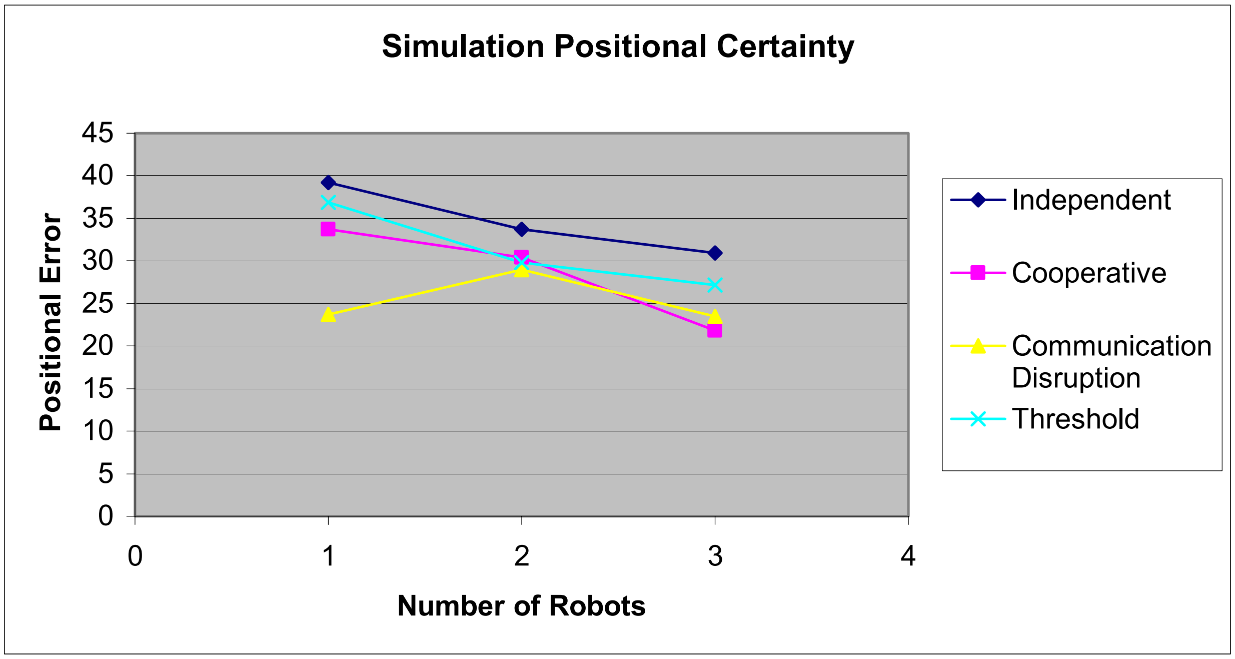


As before, the communication disruption produces results which are significantly separated from the other three tests. In all cases, as the number of robots increases, the number of sightings reaches a peak where adding more robots does not substantially increase the number of sightings.

6.3.3 Physical Imitation Results

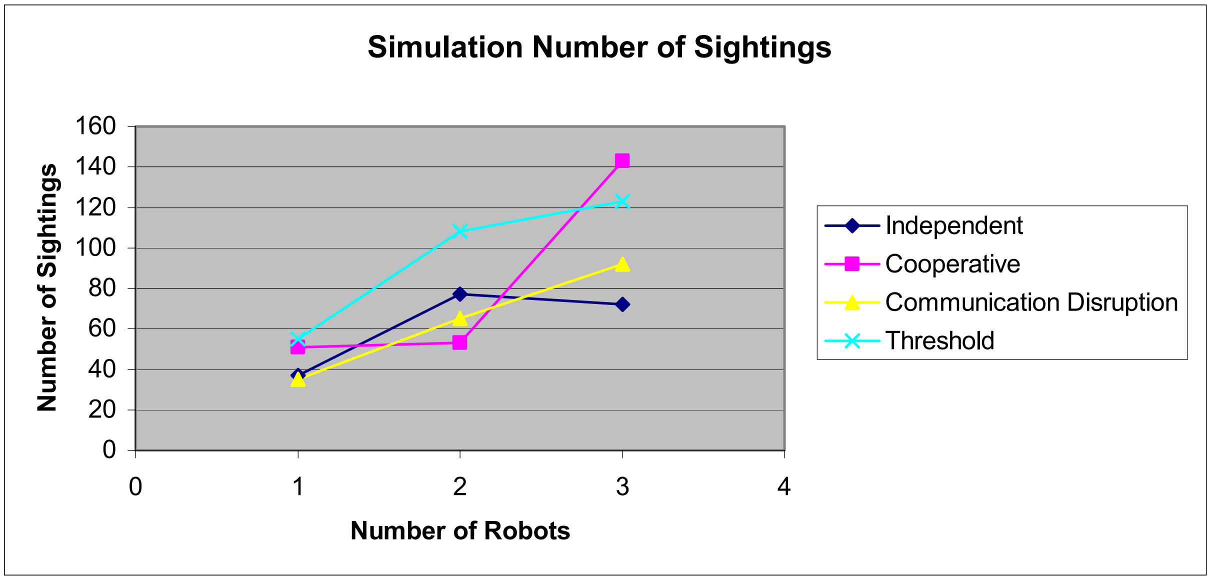
These simulations are designed to model the behavior expected in the physical testing. The tests use the same system as the trials before, but only one to three robots are tested, as in the case in the physical testing.

Positional Certainty – Positional certainty is the percentage of the area to which the target's position is narrowed down. Lower percentages represent a more accurate placement of the target.



As expected the cooperative tests and those using a communication threshold produces slightly more accurate results than the independent robots. Unexpectedly, the communication disruption produces the most accurate positioning of the target. This discrepancy is discussed in Section 6.5.

Number of Sightings – The number of sightings reports the number of chances a robot had to take a measurement of the target's position.



The number of sightings produced results closer to the expected outcome but still there were no clear trends as in the large-scale testing. The communication disruption was on the lower end and the cooperative produced the maximum number of sightings.

6.4 Physical ER1 Results

This section describes the test methodology and the results obtained from the ER1 hardware testing. Due to space constraints in the lab, the testing occurred in a 20-foot by 20-foot classroom on the Virginia Tech campus. This room provided a large open space with ample electrical outlets to recharge the laptops and ER1 batteries. A general-purpose, consumer wireless router was used to create the communication network. Having a dedicated network kept the network delay constant during the testing as there were no outside users creating sporadic increased data transfer. Also, placing the wireless router in such close proximity to the robots allowed for maximum signal strength throughout the testing room.

6.4.1 Physical Test Procedure

The following testing procedures were created to expedite the testing and may help to standardize future testing at Virginia Tech.

- *Test Area* – The testing area is formed by flipping twelve tables on their side to form a three table by three table square arena as shown in Figure 13. Each table is 152 cm long, which is approximately four robot lengths, making the total area approximately 112 m^2. The tables made exceptional barriers because the flat reflective tabletop provided excellent feedback for the IR sensors. Also, the arena is easily reconstructed without tedious and error-inducing measurements keeping the size constant.
- *Time* – Each test lasted five minutes. This time limit was programmed into the robots using the C `getTime()` function, so they stop moving at the timeout to negate any errors incurred from human timing. Five minutes appeared to be ample time for the system to settle out into a steady state and achieve consistent results. Also the longer the system operates, the positional error increases as robot collisions occur. Each test was repeated three times and the results averaged to account for statistical deviations.

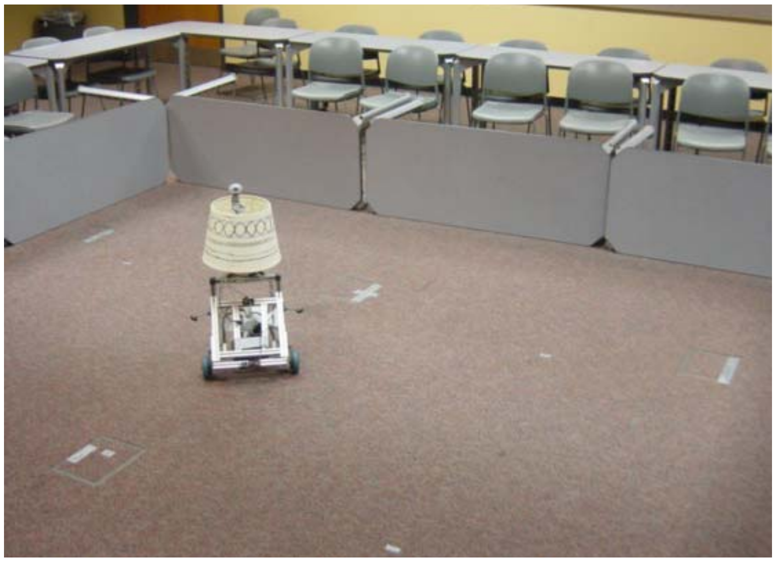
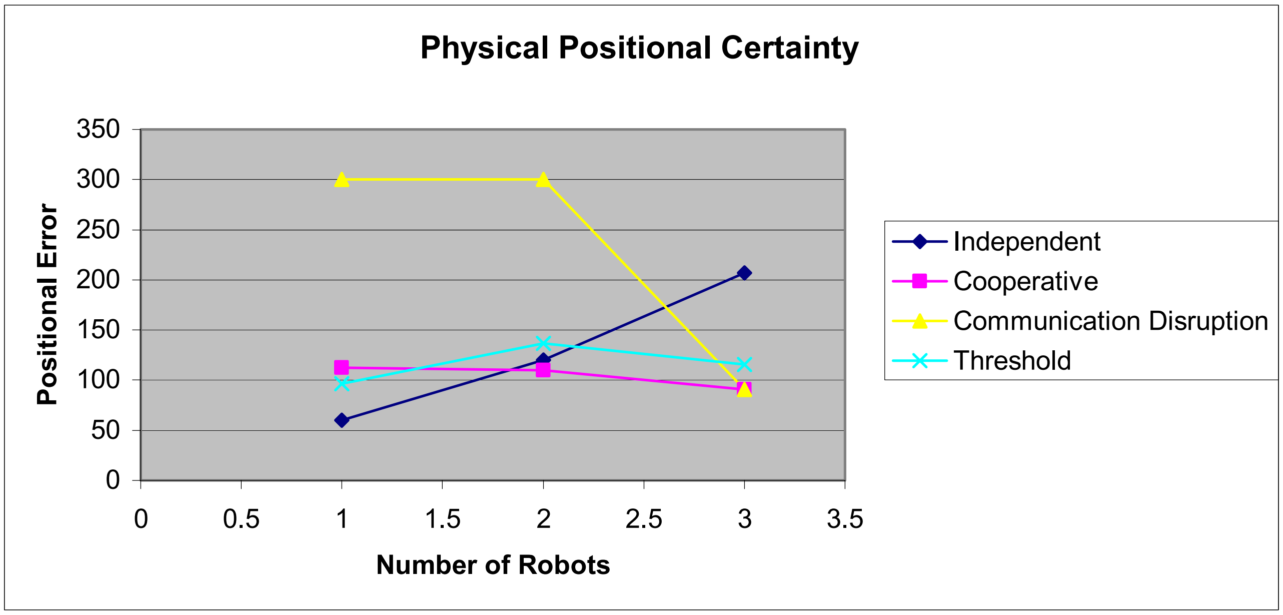


Figure 13: The area used for the robotic testing.

- *Data* – To test the accuracy of the system we record the space-time position of the target and the system's perceived position.
- *Target* – A ER1 with a symmetrically decorated lampshade attached above it served as the target for the other tracking ER1s. It ran the same search state code to move randomly as the rest of the search team except it did not change states or respond to external stimuli. The target also moved at half the speed of the searchers to allow them to catch up to it.

6.4.2 Positional Certainty

When benchmarking surveillance or tracking systems, comparing the system's perceived position in space and time to the actual position of the target is the truest measure of the system's usefulness.

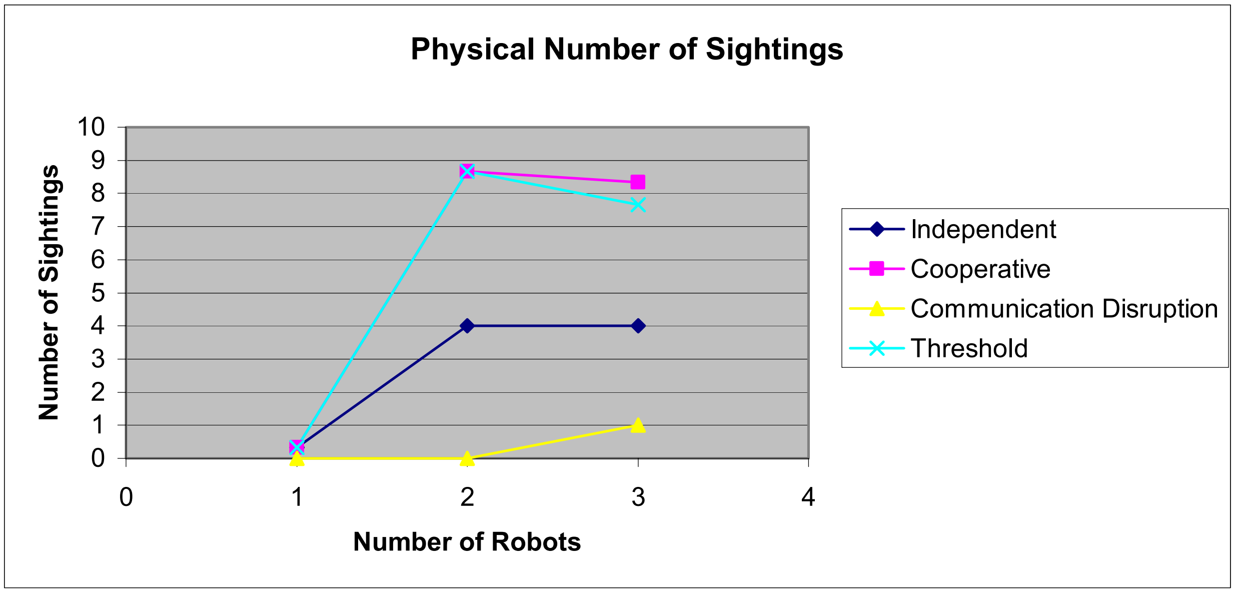


*In the first two tests with one and two robots, no sightings were made and no data recorded when communication was disrupted. These trials were given a positional certainty of 300%

Communication disruption produced dramatically different results compared with the other tests because in two instances the robots never found the target. Instead, all of the robots huddled in a corner of the area and collided with one another. All the tests converged to a similar accuracy when the number of robots was increased to three.

6.4.3 Number of Sightings

Due to the positional error incurred by robot-to-robot collisions, measuring the number of sightings by the robotic system should be considered as a performance indicator. Improved sensor technology will improve positional certainty but the robots must still initially be in place to take the measurement. Also as the number of robots is increased, the number of collisions increases offsetting the accuracy gains of taking more readings.



This data is most like the expected results. The communication disruption prevented the system from recording almost any sightings. The cooperative behavior was the most effective closely followed by the cooperation with a communication threshold.

6.5 Analysis

This section analyzes the results gathered in both simulation and hardware.

6.5.1 Overview

Although there is some fluctuation, the overall trend is as expected. In general, cooperative robots show lower positional error and higher numbers of target sightings than an equivalent number of independent robots. Communication disturbance and miscommunication deteriorated system performance as all robots became grouped in one area. It did not cause total system failure as expected because the target would on occasion wander into the area that all the robots were being called to. The communication threshold allowed for the system to maintain performance near that of the cooperative robots while communication disturbances were in place.

6.5.2 Simulation

The large-scale simulations provided the data that was consistent with this thesis's theories regarding swarm intelligent communication. In both the positional certainty and sightings tests, the communication disturbance results are well below the performance of the other system scenarios. The communication threshold allowed the system to overcome the miscommunication and return results comparable to the cooperative system. Also, the data shows that as the number of robots increase, the performance reaches a plateau where the independent, cooperative, and communication threshold systems meet. For example, the cooperative system using 50 robots sights almost twice the number of robots as the independent robots, but when the numbers are increased to 250 robots, the performance gains are diminished.

The simulations results mimicking the physical tests were not as persuasive evidence. In the physical simulation, the communication disruption actually had the least positional error. The cooperative system sighted the target the most times, but again the communication disruption did not significantly impair the system and its performance was on par with the independent robots. This is due to the smaller size of the area to which the target was constrained. The probability was much greater that the target would wander into the area to which all the robots were being lead.

6.5.3 Physical Testing

The positional certainty results of the physical implementation were dismally inaccurate. Although the tracking performance was poor, the comparison showed a distinct disadvantage of the communication disruption as compared to the other systems. The positional error was sometimes over 100% and, contrary to all the simulation data, increased as the number of robots increased in some cases. This positional error is attributed to robot-to-robot collisions. The increased number of robots increased the collisions, which further skewed the coordinate systems of individual robots. The error rose above 100% when the robots' position became so skewed the target was reported as being outside the test area.

The number of sightings provided more consistent data. The communication disruption allowed almost no sightings of the target as all the robots were crashing into one another in one corner of the test area. The cooperation and communication threshold nearly doubled the sightings of the independent system. As the number of robots increased to three, the number of sightings fell slightly. This is attributed to an increase in collisions as in the simulation results.

6.5.4 Simulation versus Physical Results

The simulation and physical results correspond well to one another. Aside from the positional certainty differences in the physical simulation, the simulation results show the same trends as the physical results. The simulator seemed to produce more consistent results with larger numbers of robots. StarLogo was designed to simulate the patterns of interaction of hundreds of agents, and simulations where the robots number in the single digits may require more trials to normalize the increased variance of singular entities. The trends of the large scale simulations' sightings were accurately reflected in the physical trials. In conclusion, the simulator is a worthwhile tool for determining overall system behavior, but its results are less interesting as the number of interacting agents is decreased.

Chapter 7

Conclusions

This chapter summarizes the thesis work and contributions. Possible future research projects and improvements are also discussed.

7.1 Summary

This thesis presented work done in the field of swarm intelligence at Virginia Tech. It details the procedure, hardware, and results obtained from experiments investigating the effects of miscommunication amongst swarming agents. All of the author's goals stated in Chapter 1 were achieved.

It was shown in both simulation and physical hardware that current system utilizing swarm intelligent control schemes have a major weakness involving false communication. If a malicious force infiltrates the communication of a robot malfunctions, system that blindly follows external communication could enter a terminal deadlock state. To remedy this situation, a communication threshold scheme was implemented and tested. This threshold prevented the robots from responding to external communication that did not meet certain levels of criteria as specified in Chapter 3. Also, if following a signal did not lead to anything of interest, the robot entered an isolation state for a period of time where it did not respond to any external communication.

Finally, a surveillance system using swarm intelligence control scheme was developed as a proof of concept and a test bed for the previously mentioned experiments. A group of mobile autonomous vehicles track a known target in space and time. The decentralized control and limited machine complexity needed creates a system that is tolerant to single agent destruction. This resiliency to physical attack makes the system of interest to military and defense contractors pursuing information gathering research.

This research has created a foundation for further research at Virginia Tech. The hardware acquired and assembled is configurable allowing it to adapt to new systems and research projects. The research procedures and methodologies presented will allow for faster results gathering in future projects. The ERSP state machine implementation methodology, as described in Chapter 5, will help propel any state design to implementation much faster.

7.2 Future Work

This thesis project has laid the foundation for a future research into swarm intelligence at Virginia Tech. Being an emerging field, there are many possible projects that can build upon this research. First, the threshold can be expanded to be more of a performance enhancement than just a safety check. In theory, the robots could use even more local information to create a better estimated global picture to decide if they should follow a signal. For example, a robot might log its encounters with other robots and obstacles to get an estimate of how complex the area is and how long it may take to traverse a distance.

Every collision with an object or another robot creates error in a robot's perceived position because the wheel slip and turn but the robot is not moving. A more reliable method of determining a robot's spatial positioning would improve results accuracy and make the cooperative behavior more effective as a truer location could be shared. This could be done with a global positioning system or an inertial sensor system. Work is currently being done at Virginia Tech to integrate the Phoenix/AX inertial/GPS board into the ER1 system.

To create a system capable of completing real world tasks, the sensory equipment, infrared and image recognition, must be upgraded. As mentioned in Chapter 5, the sensory equipment is geared more toward research and education than actual implementation. As the sensors are improved, comparisons to this system will not be valid in terms of distributed architectures. If the sensors are updated, all baseline testing must be repeated using the improved equipment to generate a new baseline with which to test the architecture.

The next large, useful improvement to the system is to expand the movement to three dimensions. Currently the system is designed for land-based vehicles, but the same concepts of signaling and following would be applicable to a flight-based system. Flight introduces an increased chance of robot destruction from obstacle or robot-to-robot collisions. A British research project using miniature helicopters [39] has recently been started to build swarming robots capable of flight.

Another interesting project would be to replace the robot control with an FPGA. The control is not extremely complex and the input/output process of the behavior networks is very similar to design in an HDL language. Also, the Configurable Computing Machine Laboratory at Virginia Tech is one of the world leaders in FPGA research and applications. By replacing the controlling laptop and control circuits with a single FPGA, the total power consumption of the robot can be lowered. Power usage is critical to the performance of any autonomous robot because batteries are a major part of the payload. In a real world application, reducing the battery weight increases the payload and range. Also decreasing the number of components may decrease the cost as well. All these factors are influential in convincing users such as the military to adopt a swarming system over conventional forms of surveillance and information gathering.

References

[1] A. Adamatzky, P. Arena, A. Basile, R. Carmona-Galan, B. Costello, L. Fortuna, M. Frasca, and A Rodriguez-Vazquez, “Reaction-Diffusion Navigation Robot Control: From Chemical to VLSI Analogic Processors,” IEEE Transactions on Circuits and Systems, Vol. 51, No. 5, May, (2004).

[2] R. Alami, S. Fleury, M. Herrb, F. Ingrand, and F. Robert, “Multi-Robot Cooperation in The MARTHA Project,” IEEE Robotics and Automation Magazine, Vol. 5, No. 1, pp. 36-47, (1998).

[3] K. AlbeKord, A. Watkins, G. Wiens, and N. Fitz-Coy, “Multiple-Agent Surveillance Mission with Non-Stationary Obstacles,” Proceedings of 2004 Florida Conference on Recent Advances in Robotics,” pp.1-5, May, (2004).

[4] R.C. Arkin, “Behavior-Based Robotics,” The MIT Press, Cambridge, MA, (1998).

[5] J. Arquilla and D. Ronfeldt, Swarming and the Future of Conflict, RAND, (2000).

[6] G. Beni and J. Wang, “Distributed Robotic Systems and Swarm Intelligence,” Proc. Of IEEE Int. Conf. Rob. Autom., pp. 1914, April, (1991).

[7] K. Boehringer, R. Brown, B. Donald, J. Jennings, D. Rus, “Distributed Robotic Manipulation: Experiments in Minimalism,” Proceedings of the Fourth Int. Symposium on Experimental Robotics, pp. 11-25, (1995).

[8] E. Bonabeau, G. Theraulaz, J. Deneubourg, S. Aron, and S. Camazine, “Self-Organization in Social Insects,” Tree, Vol. 12, No. 5, pp. 188-193, (1997).

[9] E. Bonabeau, M. Dorigo, and G. Theraulaz, Swarm Intelligence: From Natural to Artificial Systems, Oxford University Press, (1999).

[10] D.C. Brogan and J.K. Hodgins, “Group Behaviors for Systems with Significant Dynamics,” Autonomous Robots, Vol. 4, No. 1, pp. 137-153, (1997).

[11] R. Brooks, “A Robust Layered Control System for a Mobile Robot,” IEEE Journal of Robotics and Automation, Vol. RA-2, No. 1, pp. 137-153, (1997).

[12] A. Cai, T. Fukuda, F. Arai, T. Ueyama, and A. Sakai, “Hierarchical Control Architecture for Cellular Robotic System – Simulation and Experiments,” Proceedings of the IEEE Int. Conference on Robotics and Automations, pp. 1191-1196, June, (1995).

[13] Y.U. Cao, A.S. Fukunaga, and A.B. Kahng, “Cooperative Mobile Robotics: Antecedents and Directions”, Autonomous Robots, vol. 4, pp. 1-23, (1997).

[14] K. Chakrabarty, S.S. Iyengar, and H. Qi, “Grid Coverage for Surveillance and Target Location in Distributed Sensor Networks,” IEEE Transactions on Computers, Vol. 51, No. 12, pp. 1448-1453, Dec., (2002).

[15] C. Chang, “Biomimetric Robotics: Application of Biological Learning Theories to Mobile Robot Behaviors,” PhD Thesis, Boston University, MA, (1999).

[16] W.W. Cohen, “Adaptive Mapping and Navigation by Teams of Simple Robots,” Robotics and Autonomous Systems, Vol. 18, pp. 411-434, (1996).

[17] Colorni, M. Dorigo, F. Maffioli, V. Maniezzo, G. Righini, and M. Trubian, “Heuristics from Nature for Hard Combinatorial Problems,” International Transactions in Operational Research

[18] D.E. Culler and J.P. Singh, Parallel Computer Architecture A Hardware/Software Approach, Morgan Kaufmann Publishers, (1999).

[19] S. Das, G. Beni, and S. Hackwood, "An Optimal Sensing Strategy for Mobile Robots," Proceedings of the 1992 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 393-398, July, (1992).

[20] J.P. Desai, V. Kumar, and J.P. Ostrowsky, "Control Changes in Formation for a Team of Mobile Robots," Proceedings of the IEEE Int. Conference on Robotics and Automation, pp. 1556-1561, (1999).

[21] M. Dorigo, "About Ant Colony Optimization," http://iridia.ulb.ac.be/~mdorigo/ACO/about.html, December, (2004).

[22] "Evolution Robotics," http://www.evolution.com, (2003).

[23] J.M. Epstein and R. Axtell, "Growing Artificial Societies: Social Science from the Bottom Up," The MIT Press, Cambridge, MA, 1996.

[24] "Federal Government Spending on Surveillance Technology 2005," http://www.cpsr.org/issues/privacy/surveillance05report, (2005).

[25] L.M. Gambardella, E. Taillard, and M. Dorigo, "Ant Colonies for the Quadratic Assignment Problem," Journal of Operational Research Society, Vol. 50, pp. 167-176, (1999).

[26] V. Genovese, P. Dario, R. Magni, and L. Odetti, "Self Organizing Behavior and Swarm Intelligence in a Pack of Mobile Miniature Robots in Search of Pollutants," Proceedings of the 1992 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1575-1581, July, (1992).

[27] J. Hoffmeyer, "The Swarming Body", Proceedings of the Fifth Congress of the International Association for Semiotic Studies, pp. 937-940, (1997).

[28] O. Holland and C. Melbuish, “An Interactive Method for Controlling Group Size in Multiple Mobile Robot Systems,” International Conference on Advanced Robotics, pp.201-206, (1997).

[29] S. Ichikawa, F. Hara, and H. Hosokai, “Multi-Robot System using Hello-Call Communication,” Proceedings of the 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1149-1156, July, (1993).

[30] S. Ichikawa and F. Hara, “Experimental Characteristics of Multiple-Robots Behaviors in Communication Network Expansion and Object-Fetching,” Proceedings of the Third Int. Symposium on Distributed Autonomous Robotic Systems, pp. 224-234, (1994).

[31] A.J. Ijspeert, A. Martinoli, A. Billard, and L.M. Gambardella, “Cooperation through the Exploration of Local Interactions in Autonomous Collective Robotics: the Stick Pulling Experiment,” Tech. Report LAMI-EPFL, (1999).

[32] K. Inoue, J. Ota, T. Hirano, D. Kurabayashi, and T. Arai, “Iterative Transportation by Cooperative Mobile Robots in Unknown Environment,” Proceedings of the Fourth Int. Symposium on Distributed Autonomous Robotic Systems, pp. 3-12, May, (1998).

[33] S. Johnson, Emergence The Connected Lives of Ants, Brains, Cities, and Software, Simon & Schuster, (2002).

[34] I.D. Kelly and D.A. Keating, “Flocking by the Fusion of Sonar and Active Infrared Sensors on Physical Autonomous Mobile Robots,” Proceedings of the Third Conference on Mechatronics and Machine Vision in Practice, Vol. 1, pp. 1-4, (1996).

[35] D.B. Lange and M. Oshima, “Seven Good Reasons for Mobile Agents”, Communications of the ACM, Vol. 42, No. 3, March (1999).

[36] A. Martinoli. "Swarm Intelligence in Autonomous Collective Robotics: From Tools to the Analysis and Synthesis of Distributed Collective Strategies," Ph.D. Thesis, October, 1999, DI-EPFL, Lausanne, Switzerland.

[37] J.H. Moor, The Turing Test, Kluwer Academic Publishers, (2003).

[38] L.E. Parker, "ALLIANCE: An Architecture for Fault Tolerant Multirobot Cooperation," IEEE Transactions on Robotics and Automation, Vol. 14, No. 2, pp.220-240, (1998).

[39] K. Poulsen, "Linux Powers Airborne Bots", http://www.wired.com/news/linux/0,1411,67695,00.html, June 1, (2005).

[40] C.W. Reynolds, "Flocks, Herds, and Schools: A Distributed Behavioral Model, in Computer Graphics," SIGGRAPH '87 Conference Proceedings, pp. 25-34, (1987).

[41] M. Rude, "Collision Avoidance by Using Space-Time Representations of Motion Processes," Autonomous Robots, Vol. 4, No. 1, pp.101-120, (1997).

[42] E. Sahin, T.H. Labella, V. Trianni, J. Deneubourg, P. Rasse, D. Floreano, L. Gambardella, F. Mondada, S. Nolfi, and M. Dorigo, "Swarm-Bot: Pattern Formation in a Swarm of Self-Assembling Mobile Robots," http://asl.epfl.ch/aslInternalWeb/ASL/publications/uploadedFiles/mondada-ascona02.pdf, (2002).

[43] M. Saptharishi, S. Oliver, C.P. Diehl, K. Bhat, J.M. Dolan, A. Trebi-Ollennu, and P. Khosla, "Distributed Surveillance and Reconnaissance Using Multiple Autonomous ATVs: CyberScout," IEEE Transactions on Robotics and Automation, Vol. 18, No. 5, Oct., (2002).

[44] C. Shen and C. Jaikaeo, “ Ad Hoc Multicast Routing Algorithm with Swarm Intelligence,” Mobile Networks and Applications, Vol. 10, pp 47-59, February, (2005).

[45] L. Spector, J. Klein, C. Perry, and M. Feinstein, “Emergence of Collective Behavior in Evolving Populations of Flying Agents,” Proceedings of the Genetic and Evolutionary Computation Conference, pp. 61-73, (2003).

[46] K. Sugawara and T. Watanbe, “A Study on Foraging Behavior of Simple Multi-robot System,” University of Electro-Communication, (2002).

[47] “StarLogo,” http://education.mit.edu/starlogo/, June, (2004).

[48] D. Talbot, “The Ascent of the Robotic Attack Jet,” http://www.technologyreview.com/articles/05/03/issue/feature_jet.asp?p=0, March, (2005).

[49] V. Varveropoulos, “Robot Localization and Map Construction using Sonar Data,” December 2004, http://rossum.sourceforge.net.

[50] “Webster’s Online Dictionary,” http://www.websters-online-dictionary.org/, July, (2005).

[51] B.B. Werger and M.J. Mataric, “Robotic “Food” Chains: Externalization of State and Program for Minimal-Agent Foraging,” Proceedings of the Fourth Int. Conference on Simulation of Adaptive Behavior: From Animals to Animats, pp. 625-634, September, (1996).

Vita

The author, Michael Brian Marshall, was born in Martinsville, Virginia in 1982. He spent much of his formative years in the hallowed halls of Carlisle School, where he eventually graduated with both high academic and athletic accolades in 2000. The new millennium saw Brian embark upon Computer Engineering studies at Virginia Tech. This new den of inequities, as his grandmother termed any university, presented a wealth of opportunities for the eager, young student. Excelling in his work, he received his B.S. degree summa cum laude in December 2003, but this was not the end of his time at Virginia Tech. Brian remained. He decided to endure the scholarly rigors of graduate school to obtain a M.S. degree and capture a coveted intramural championship, which had eluded him throughout his undergraduate career. Whilst in pursuit of his graduate studies, he was a teaching assistant for Microprocessor Design I and worked part-time at Adaptive Genomics, a small bioinformatics startup.