Swarm Intelligence in Autonomous Heterogeneous Robotic Navigation Over Land and Water Using a Single Algorithm

A thesis
submitted by

Gregory Harry Meyerhoff

In partial fulfillment of the requirements
for the degree of

Master of Science
in
Mechanical Engineering

TUFTS UNIVERSITY

May 2012

Advisor:
Professor Chris Rogers

Abstract

Exploration, Surveillance operations, Search and Rescue, and Security are becoming more important to both civilian and military endeavors every day. However, several applications are dangerous for humans, which points to a critical role for robots. Teamwork is often carried out by someone commanding the operations as a central leader, but if contact with this person is lost, the team often has to quickly regroup and review options under high pressure which can potentially be dangerous and a threat to success. Such situations may be resolved by employing autonomous agents to work together to accomplish a goal in which there is no full-time central leader. Agents employing swarm intelligence offers a significant advantage over a group with an assigned leader because there is no single point of failure. This allows for continuation of the mission, even if agents are lost during it.

For my research, I developed a single algorithm for a scalable heterogeneous swarm of agents to reach a beacon located on land or water. Agents collectively determine which is the closet to it and sends that agent forward. They make decisions on which terrain they can transverse using local surface reflectivity as a cue to check knowledge of their own physical capabilities or message another if it is more fit to traverse. The swarm consists of a water-agent and multiple land-agents that were able to distinguish land from water with 100% accuracy in experiments performed. Results of various tested scenarios demonstrated that while the dynamic ad-hoc mesh network I created allowed the swarm to succeed in broadcast communication with each other 78% of the time, the swarm had an overall success rate of 84% in reaching the beacon in experiments performed. These results demonstrated that the heterogeneous swarm was robust and capable of overcoming several challenges it faced in the scenarios. This research can provide directions for future research in swarm intelligence using heterogeneous agents with numerous real-world applications.

Acknowledgments

I want to express my gratitude to the members of my thesis committee, Professor Ethan Danahy, Professor Jason Rife, and especially to my advisor, Professor Chris Rogers, who has been a constant and supportive influence in ensuring my success through my masters.

Also, I am in continuous thanks to my loving mother and father who have always stood by me, offered advice, supported my efforts, and have helped guide me throughout my life. Words cannot express the depth of my love and appreciation for you both.

Table of Contents

List of Tables

List of Figures

Nomenclature

A_{smax} Theoretical maximum speed of an agent in which it can recognize a change in a terrain and output a command to the motors to brake before it reaches the new terrain

B_{Tbbr} amount of time required to broadcast message to swarm the responding agent has reached the beacon

BT_m amount of time required to send a message, 8 seconds

$data_{es}$ elapsed time between EOPD recognition of a new surface and power output to the motors

d_{wl} horizontal distance between the middle of the front wheel and the light beam aimed incident to a surface

IR_{start} threshold signal strength which an agent can first detect a signal

IR_{stop} threshold signal strength which the agent is programmed to stop at from the beacon

m position order of message received

n number of agents in swarm

S_{fwd} speed of agent

T_{A1R} Time until Agent-1 should reset if it has not received a message from Agent-n that it reached the beacon

T_{A1W} time Agent-1 should wait until all agents should have returned signal strengths

T_{ANR} Time that Agent-n should reset after if it has not reached the beacon yet

T_{ANW} time required by any agent within the swarm to return its information to Agent-1

t_c signal time return (system constant), 25 seconds

t_n searching for signal + holding value, 10 seconds (system constant)

T_{RBN} time agents should wait to hear from Agent-n that signal has been reached before resetting

t_s settling time

t_{ts} transient test speed is the time at which the agent was tested at as it changed between surfaces

Chapter 1: Introduction and Motivations

1.1 An Overview of the Need for Autonomous Robotics

Exploration, Surveillance operations, Search and Rescue, and Security are important to civilian and military endeavors. Each area has logistical, practical, and cost-worthy considerations which are taken into account for mission planning. Many applications are dangerous for humans, and thus point to a critical role for robots. [1, 2] Teamwork is carried out by someone running the operations as a central leader, but if contact with this person is lost, the team often has to quickly regroup and review options under high pressure which can potentially be dangerous and a threat to success. To address many of these issues, the Department of Homeland Security is interested in developing standards for robots to be used in Urban Search and Rescue (USAR) applications, such as mobility, sensing, communications, and terminology. [2] Such applications can show benefits employing swarm intelligence, as it allows for a decentralized operating approach, which replaces the standard model of a central leader. One such advantage is if a robot (agent) becomes lost or inactive during the operation, it will not affect the overall execution of the mission since there is no single point of failure. [24] While the state of technology of autonomous vehicles is advanced enough for mission scenarios, [3] a shift in operating procedures would allow for more efficient planning. This would work by allowing the first agent to locate the signal (beacon) to broadcast this information to the swarm of agents, instead of having to route information through a specific central leader. This would create a dynamic ad-hoc mesh network of autonomous agents, each which are capable of becoming a temporary leader, therefore leaving no single point of failure within the swarm. As technology becomes more sophisticated, and both military and civilian demand for robots increases, swarm intelligent agents make ideal candidates for a variety of endeavors.

In August of 2011, President Barack Obama signed the Budget Control Act reducing the country's deficit by $1 trillion over the next 10 years through discretionary spending caps. [26] Then in February 2012, he unveiled the Budget of the United States Government for Fiscal Year 2013, which was further laced with radical measures aimed to tighten the country's national deficit. Of specific interest to this thesis, military spending is expected to decrease 31% to $88.4 billion, down from $115 billion in 2012, which is a significant amount of spending. This plan shines in the light of Congress's ambitious goal of reducing $487 billion in Pentagon spending by 2021. In addressing this, Secretary of Defense Leon Panetta stated, "…The military will be smaller and leaner, but it will be agile, flexible, rapidly deployable and technologically advanced. It will be a cutting-edge force." [17]

With the decrease in proposed military budget spending over the next 10 years, this amounts to an estimated job loss of -13% in forces from the Army, down from the current personnel enlistment of 560,000 to 490,000. Similarly, the Marines are expected to cut -10% of their forces, from the current 200,000 personnel to 182,000. These cuts amount to a decrease of 84,000 jobs, or -23% over the next 10 years. In other words, about 1 out of 4 military personal jobs in these areas will be eliminated. [17] These budget reductions come as an ease to the current hefty price tag accommodating the deployment of large forces overseas at a rough annual cost of $1 million per solider. [9]

As an upside for the military, the budget cited the success of UAVs in the successful usage in recent wars in Iraq and Afghanistan, and calls for an increase in spending on these technologies. [17] This recognition points to a direction to offset these job losses through direct interest in technological advancement of autonomous robotics through job creation to meet the military's goals and budgeting financial interests through technological advancements. Furthermore, if the technology can fulfill the interest and desire of the military, as the government is requesting, job growth in technology sectors may increase as a result. The movement to create more autonomous robotics for military applications will also be more pleasing as an option to both the proponents and opposition of war in the broad agreement amongst experts that "… [autonomous robotics] cause far fewer unintended deaths and produce far fewer refugees than either ground combat or traditional airstrikes". [9]

Additional motivation for autonomous robotics for military applications can be drawn from the financials of unmanned aerial systems (UAS). After the call for the introduction of them in 2001, [27] they now represent 41% of total aircraft in the military (up from just 5% in 2005), while the remaining 59% comprises manned craft, which accounts for 92% of total operating costs of all aircraft combined. This significant overhead is further alarming considering that UAS now have remote control flying capabilities as safe as a F-16, [28] consuming only 8% of total cost. This demonstrates how cost-effective, reliable, and advanced the field of autonomous robotics is and how fast it is progressing. And with Congress's goal of arming the military with 33% of unmanned craft (which is part of UAS) by 2011, [14] it demonstrates the seriousness the government has at fulfilling its autonomous vehicle commitment.

With budget cuts and concerns by the government and military over war casualties, precision, and goal to maintaining world-class leadership in technological advancements in the spotlight, this sets the right time for advancements of autonomous unmanned ground vehicles (UGVs), which have become popularized through navigational races and challenges, such as the Defense Advanced Research Projects Agency (DARPA) Grand Challenge. Since the competition's initiation in 2004, Congress has authorized the distribution of cash prizes to encourage technological achievements in autonomous robotic vehicles to purge technological discoveries for military interests with a goal of the

Congressional mandate securing one-third of military ground combat vehicles unmanned and autonomous by 2015. [14]

This pivotal move in autonomous military robotics interest and budget allocation for it comes in parallel at a time when military analysts are suffering the burden described as an “information overload” from too much data to analyze, possibly undermining its own efforts to gain more intelligence from what they are tracking. [50] Already, UAVs produce 39 video feeds, 24 hours a day, which may jump to 3,000 feeds as airborne surveillance programs expand in years to come. This 760% increase would be in addition to the manned Air Force aircraft and the Army’s UAV surveillance programs. [51]

Instead of gathering larger volumes of aerial data, which analysts will have to sift through even more looking for items of interest, a different approach to gathering information in inaccessible areas could be to use ground agents to provide surveillance work for the military. This new data is not necessarily intended to add to an already overwhelming stock pile of information to sift through, but rather provide high-resolution imagery from cameras onboard UGVs, as opposed to increasing the supply of typically lower-resolution images of aerial data gathered from UAVs. [52]

Equipping the military with small UGVs capable of multiple terrain traversing would be a significant and advantageous approach to surveillance and data-gathering methods, as it would allow for military intelligence into unknown areas. This approach may one day replace potentially deadly manned ground missions as these are far too dangerous for soldiers because of possible attacks by ground ambush, IEDs, or from weapon-launched artillery firing. This is potentially more hazardous in areas where GPS, communication signals, and aerial surveillance may be weak, unavailable, or doesn't provide useful results, such as in mountainous regions, within caves, or isolated areas. UGVs could provide the visual intelligence to military analysts to detect and track suspected targets of interest without subjecting humans to these missions or environments.

If a target of interest is located, each agent can communicate its current distance away from it, thus enabling the swarm to figure out which agent is closest to the target – in which case that agent would be sent towards it. Using a swarm capable of multi-terrain transversal allows for an autonomous ground or water-capable agent to reach the target, regardless of whether it is located on land or separated by water. This could save valuable time and resources by leveraging the strength of individual agents to reach the target.

1.2 Thesis Statement

In this thesis, I aim to demonstrate swarm intelligence applied to autonomous heterogeneous agents that are capable of reaching a beacon, regardless of the terrain it is located on. Agents are able to make decisions on which terrain they can transverse based on the surface reflectivity of locally encountered terrain as to whether it is fit to transverse towards the beacon upon signal recognition based on knowledge of its own physical capabilities. The swarm leverages agents' strengths by prompting a call to another agent if the responding agent determines that another is more capable to transverse the terrain. Any agent within the swarm has the potential to be able to respond to a beacon located on land, with the main advantage that land-agents are more agile because of their smaller size and can reach it quickly if one is the closest. If the beacon is found to be separated by water from a responding land-agent, then it will message the water-agent to be sent instead. The heterogeneous swarm of agents designed were all programmed with the same algorithm

1.3 Thesis Objectives

The specific goal of this thesis is to demonstrate that a heterogeneous swarm of agents, using a single algorithm, can reach a beacon located on land or water. By using surface reflectivity of locally encountered terrain to cue knowledge of their own physical capabilities, an agent can decide if it able to traverse the terrain towards the beacon upon recognition of it, or if another is more fit to do so. Agents employing swarm intelligence offers a significant advantage over a swarm with a single leader because there is no one point of failure in a decentralized swarm. This is important because it means the mission can continue, even if multiple agents are lost during it. This will be accomplished through the development a single code, which will be loaded onto each agent so that they can act autonomously and search for the signal, and then communicate with each other in the field who should go to the located beacon. Using a single code allows for easy updating of the algorithm which can be deployed on all agents, regardless of the terrain which they are assigned to, instead of having to customize it for individual ones assigned to different terrains. The swarm algorithm, as tested, was set for two land-agents and one water-agent, and is scalable for both types.

Secondary objectives include:

- Developing a scalable surveillance system of n-autonomous agents.
- Development of a scalable collision avoidance system which calls in specific sensors for obstacle avoidance for an agent, depending on which terrain the agent is assigned to transverse over.
- Demonstrate that by placing an agent within a low-cost plastic ball, and changing a simple text file (which is read by the algorithm) from "land" to "water", the agent can now drive across water, where the land-agents cannot.

1.4 Thesis Organization

In Chapter 1 I discussed the needs for autonomous robots in military and civilian applications. The thesis statement and objectives outline the main concepts for my research, pointing to new methods of employing swarm robotics with important designs for further research directions. I begin Chapter 2 with discussion of how human intelligence and swarm intelligence are defined, which I then further dissect by discussing swarm behavior found in nature. Finally, research and applications of swarm intelligence, along with advantages and disadvantages in methods of communications between agents are discussed.

Building on the previous chapter's foundations, I discuss in Chapter 3 how I created the algorithm which is used on the swarm of agents in my thesis and how it works. Chapter 4 provides the overview of the physical robotics used in this thesis. Specifications on the sensors, beacon, and other hardware used are also discussed.

In Chapter 5 I discuss how the agents communicate with each other when deciding who should traverse towards the signal. I discuss how the swarm communicates with each other and how messages are packaged with important information for an agent. Additionally, equations I developed which describe the timing of events and how the agents communicate with each other during missions are explained.

In Chapter 6 I show how I tested important sensors which the swarm uses to identify terrains surfaces and locate the beacon, as well as the results and analysis of these tests. In Chapter 7, I explain how I used the results of the sensor testing in Chapter 6 to make determinations on the scenario testing conditions in which the swarm of agents were tested. Results and analysis of this testing are then discussed in terms of how accurately and robust the swarm and the communication between the agents was.

Chapter 8 details specific reasons how several functions of decision making in the code were designed and the favorable outcomes they serve both in terms of effective autonomous and swarm robotics, as how the intelligence and behaviors of the agents are designed. A scalable sensor design is discussed for how the agents know which sensor to use based on the terrain they are assigned to traverse on. The chapter concludes with the significance of the design of a single code which any agent can use, regardless of which terrain it is on, for as many agents as which are used in the field.

The thesis concludes in Chapter 9, which discusses the thesis as an overview, my results from testing the swarm, and highlighting of the main points of interest in my thesis. Outcomes, limitations, and implications of this research are related to future work with the swarm. Finally, future directions for research and potential usage for real-world applications which can be drawn from my research are discussed.

Chapter 2: Background

In this chapter, I introduce background into the origins of swarm intelligence and how in recent years it has evolved for research and into applications for society. I will introduce several examples of current research, followed by discussion of a sampling as they relate in particular to this thesis. This will be followed by an overview of the advantages and disadvantages of crucial considerations when designing the communication amongst a swarm of agents, and a working model for designing a swarm algorithm appropriate for a scenario based on goals to be accomplished by the swarm.

2.1 Intelligence

One of the longest and most deeply debated topics appearing in classical antiquity continuing through today encompasses the meaning of "intelligence", how to harness it, and how to use it. But it seems that as we gain more of an understanding of how we think and learn, we produce more ideas in a seemingly continuous list of what intelligent humans and animals are capable of while diverging from the arrival of a concise definition of the word itself. Such an inability to define such a common word and its meaning leads to perhaps the best understanding we have of intelligence as quoted in 1923 by Edwin G. Boring: "Intelligence is whatever it is that an intelligence test measures." [4]

In the 20th century, computers were introduced into society, boasting several benefits, such as speed, automation, and information storage and retrieval. As society witnessed the accelerating rate of acceptance of computers into our daily lives, speculation arose that these machines could become more intelligent than people. As it turned out, this is not the case, as modern computers are actually poor at thinking or solving certain types of problems for us, [6] but they are excellent at carrying out instructions we give them to follow. In this regard, it seems fitting that many, if not all discussions of intelligence between humans and computers eventually should give way to the Turning test – if you can trick someone into not knowing the difference between a human response to a question or a that of a computer's, then you have blurred the boundary between them. [5, 11]

In 1956, generic algorithms designed to perform decision-making choices began surfacing when John McCarthy, a computer and cognitive scientist, co-organized the first Dartmouth Summer Research Conference on Artificial Intelligence, which officially launched AI into academic recognition. [6, 7] Even though AI's main goal as a field was to have a machine think like humans do [8] has yet to be accomplished, this interest led to subsets of AI, such as "swarm intelligence", a phrase coined by Beni, Hackwood, and Wang in 1989 as a means to describe the generation of patterns and self-organization in cellular robotic systems. [49]

2.2 Swarm Intelligence

While swarm intelligence may be classified as a subset of AI, one fundamental difference is that while AI is concerned with intelligence in order to duplicate or improve upon some functionality of that of which a human is capable, research into swarm intelligence deals with breaking down the overall demands of a system into simpler processes. Another difference is that AI deals with how one agent would handle all obstacles encountered – and overcome them through adaptation – while the swarm approach uses several agents with lower-level cognitive functioning, but with the ability to share, or distribute information gathered amongst several other agents in a manner in which the sum of the parts may be more efficient at accomplishing a task than attempting to design an all-in-one system to reach a predetermined goal. [10] This is reinforced by the fundamental design of swarm robotics in that unlike classical AI robots, a swarm does not need exact knowledge of an environment and of the individual robot's positioning to accomplish the goal(s), thus making goal-reaching easier to do at a cheaper cost. [10]

However, because much research and applications utilizing swarm intelligence have been used to design a homogeneous swarm of robotics for a task or environment, a heterogeneous swarm offers theoretically no guidelines of design to respond to varying stimuli. [10] Using this idea, I have designed a heterogeneous swarm to use a single algorithm, as presented in this thesis.

2.3 Biological Inspirations

Many models of swarm intelligence can be observed by studying the social behaviors between groups in different species. It should be noted ahead of time that the swarm behavior of one species is not necessarily the same as that of another, but rather illustrates how collective efforts of a species works together to accomplish a common goal.

One influential model of swarm intelligence on how birds flock together was made by Craig Reynolds in 1987. He showed this flocking pattern, termed “boids”, [5] were driven by three rules:

- pull away before they crash into one another
- try to go about the same speed as their neighbors within the flock
- try to move toward the center of the flock as they perceive it

This simulation showed that not only were the birds able to fly together following these rules, but that if they were separated by some object, such as a large rock, they would split into two randomly sized groups, each flying on different sides of the rock and rejoin into a single flock again on the other side. The effect of this simple model of these three rules was so inspiring that this decentralized algorithm has been at the heart behind several cinematic sequences of herds and flocks in today’s animated movies. [5]

One heavily studied area of swarms is the food foraging strategies of ant colonies. [5, 10, 29, 30] Ants do not travel in flocks like boids, but rather follow one another in a trail-following behavior. These trails are formed when an ant deposits a chemical substance called a "pheromone" when transporting food from the source of it to their nest. Other ants smell this trail, and help further the process of bringing food to the nest, in what is called "mass recruitment". [10] As more ants reach the pheromones, they deposit pheromones themselves, and the swarm increases in size, speeding up the rate at which food is carried to the nest through a positive feedback loop. Additionally, while the birds in flight seem to always avoid crashing into each other, ants crawl over each other in the process to carry out these tasks.

One problem with programming a heterogeneous swarm of robots using a swarm algorithm based on a biological species which acts in swarm behavior is that the algorithm is often modeled from species containing a homogeneous population. However, when designing heterogeneous robotics, you can create a significant advantage by designing bodies fit for specific or additional tasks and calling these agents when necessary, while still running the same algorithm on all agents. In this thesis, one of the agents will be placed in a ball, capable of traversing water as well as land. While all agents have the same algorithm and perform the same actions and behaviors, only the agent which is designated for water-capabilities will actually be able to enter water. This assignment of a water-capable agent is handled by an external text file, which the algorithm checks to see if it is fit for land-only or water-capable. The advantage here is that the water-agent can also traverse land as well, at the main trade-off that the ball places it at a larger size, which may occasionally be more challenging to maneuver around objects.

2.4 Applications of Swarm Intelligence

Several companies have applied concepts of swarm intelligence for routing techniques and supply-chain management optimization to make improvements within. Some of these companies include UPS, [31] France Télécom, British Telecom, MCI WorldCom, and Unilever. [12] In 2000, Southwest Airlines was noting several problems with its cargo operations, which was creating bottle necking in routing and the handling system. After applying an ant colony algorithm, the company made several improvements to their routing systems which allowed them to maximize their company fundamentals, estimating an annual increase of $10 million after implementation. [12]

To date, there are few real-world applications utilizing swarm intelligence for robotics. However, there are several laboratories conducting research-based examples, as well as companies which have used applications of swarm intelligence, as shown in table 2.1.

Application	Company / Sector
'Seaswarm' - Autonomous robots, aiding in the clean up of oil spills by floating on water, acting as conveyer belts to absorb oil on the water's surface [18]	MIT
Autonomous Underwater Explorers' - Autonomous robots for a variety of underwater applications, suchas tracking tsunamis and aquatic life [19]	UCSD
Autonomous Guided Parachute Systems' - used for dropping supplies to troops in a coordinated manner by keeping the swarm near each other after being dropped from a plane [20]	Onyx
Search Engine Optimization' - methods used to increase a webpage's natural ranking in search engines [32]	World Wide Web
Digital ants looking for a worm in a computer infection [33]	Pacific Northwest National Laboratory (Department of Energy)
UAV SWARM Health Management Project - surveillance and monitoring of ground based objects or vehicles [34]	MIT
Traffic routing methods for telephone networks [12, 35]	France Télécom, British Telecom, MCI WorldCom
Supply-chain routing [12]	UPS, Unilever, Southwest Airlines

Table 2.1: Research or applications of Swarm Intelligence.

2.5 Similar Work

Previous research in swarm intelligence using NXT robotics from the LEGO Group used robots to forage for simulated food sources. [46] Pásztor et al. designated a full-time master/slave relationship between the robots. While it was discussed how information was distributed between robots, in the case of any damages to them, specific roles could not be transferred to another in the swarm as they were predefined. I argue that this is not swarm intelligence because of the full-time designated control and lack of ability to transfer roles. My thesis is different in that robots do not have predefined roles, and any agent is capable of, and has the same chance as any other agent, to temporary control the other agents, in which it then disconnects from agents to resume a decentralized swarm in which damage or loss of a robot does not affect the roles of the swarm.

Additional research with NXT robots traveling as a flock in a coordinated pattern has been performed. [47] Brigandi et al. also preconfigured the swarm in a master/slave relationship, citing that the maximum size of the swarm is limited to 4 robots, and that this number represents an upper limit because of Bluetooth communication and master/slave abilities. I again argue that this study does not constitute swarm intelligence because of the preconfigured master/slave relationship, for which my thesis is different by the aforementioned capabilities of the swarm I created.

In both cases, the master/slave relationship can be changed by implementation of a dynamic ad-hoc network which I developed, allowing for any agent to connect, control, and disconnect to all other agents within the swarm. Additionally, this allows for scalability by predefining the number of agents within the swarm, which is discussed more in depth in chapter 5.

2.6 Emergence

Because purposeful autonomous actions of an agent are not observed by another (unless the agents are distributing and receiving specific information), a global understanding of the swarm is unknown. However, while the agents may have a shallow understanding of the system beyond their specific task or role(s), the swarm's response to the sum of the agents within it when performing simple tasks creates a more complex behavior of the group. This provides a clearer understanding of the progress the swarm has made in accomplishing their end-goal, which follows local rules, and is oblivious to any higher-level input, organization, or commands, which is a determined macrobehavior. [13] This is why the term "emergence" is used – it reflects the sum of seemingly disconnected units to work together towards a goal with an identifiable pattern as a whole, which is unknown on the individual level, including local specifics or location of an agent ahead of time. [5, 10, 36]

2.7 Advantages and Disadvantages of Types of Control Architecture in Swarms

A case can be made for either type of a control swarm, from both a centralized leader, to a decentralized approach - the latter of which is commonly found in nature in large swarms. As I will discuss more in chapter 5 in relation to the design for a heterogeneous swarm's communication protocol, a hybrid method allows for advantages with each type of control to be maximized, while disadvantageous to be minimized as much as possible.

The number of advantages and disadvantages can be numerous and table 2.2 outlines some of the important ones associated with each type of communication design. [24, 25] As I will show herein, these begin to be explained in the motivation for design, and reasons why certain methods were chosen over less effective ones will become apparent.

APPROACH	CRITERIA	DESCRIPTION
Centralized	Advantages	Optimal plans can be produced. The leader can take into account all the relevant information conveyed by the members of the team and generate an optimal plan for the team.
	Disadvantages	Strongly rely on communication. Thus, when a communication failure takes place, it results in a failure of the entire system.
		A strongly centralized system can fail in accomplish- its task when its leader goes out of order.
		System response to changes in the environment is sluggish since all relevant information must be conveyed to the leader before any action can be taken.
Decentralized	Advantages	Do not have a single point of failure. The loss of a single agent will not cripple the system, as can be the case in single-agent or centrally controlled systems.
		Can achieve complex results with relatively simple system design. The designer need only create simple, low level behaviors, instead of a single, computationally intense control system to govern all possible situations.
		Are inherently parallel, which allows for extremely scalable systems and faster task completion.
	Disadvantages	Often result in highly sub-optimal solutions because all plans are based solely on local information.
		Independent task execution by the system components causes problems in the area of coordination between the system agents.

Table 2.2: Advantages and Disadvantages of Control Approaches. [25]

2.8 Towards a Working Model of Swarm Intelligence

Since nature is always evolving and updating the behaviors of creatures within a species to help it best survive within their habit, a swarm intelligent model based on biology can always be improved upon. The emergent nature of a swarm of agents usually follows a distinctive pattern, but is not programmed to follow a step-by-step set of actions in that the methods by which swarm accomplished a task the first time may not be exactly the same as the next, but the overall pattern remains unchanged. However, if the methods we chose to accomplish a task are successful, and are repeatable to specific specifications which are predetermined, then this process may lead to the Law of Sufficiency: If a solution is good enough, fast enough, and cheap enough - then it is sufficient. [5] This law seems very satisfactory for real-world solutions because it contains the possibility of errors built-in which could make the solution even better than the prior one if we learn from any mistakes or find a more optimal method (e.g. saving time for completion) to improve upon it. All human tasks seem to follow this pattern, right down to our analytical thought process and design, and even our scientific modeling in that the model is good enough if it incorporates all known data and produces a repeatable outcome or is met by the same goal every time it is employed.

In specific relation for swarm intelligence, biologically-inspired algorithms reiterate the concept of the “working model” by illustrating that such algorithms do not have to be designed completely following accurate or true models of these biological species. The main purpose of the algorithm and this research is not exactness, but rather efficiency and robustness, comprising the flexibility to adapt to a new environment the swarm may encounter to meet their end goal. [10]

Chapter 3: Swarm Surveillance System

3.1 Motivation to Design a Universal Algorithm

The reason for designing a single algorithm for swarm intelligence is to apply it to a heterogeneous swarm in which low-cost, robust agents can reach a beacon, while overcoming obstacles and other taxing events which may impede their progress, regardless of the terrain on which the beacon is located. The algorithm should be scalable to the inclusion of more agents and capable of levering agents of specific capabilities when their specialties are needed, and still complete the task if an agent fails during it. Designing such an algorithm which can run on all agents to accomplish a goal may be faster to update and more robust than designing different ones to run on separate agents within a heterogeneous swarm.

While several biological species were researched for their swarm intelligence properties, the behaviors of ant colonies are used as a model for my swarm intelligence algorithm. This is because the tasks involved in scavenging for food by ants is similar to the surveillance duty of agents in that ants use decentralized swarm acting collectively on the local level. In other words, there is neither a full-time "lead" ant nor a "master" agent which controls the orders for the entire swarm. Ants roam in random directions looking for food, synonymous to agents roaming and looking for a beacon (which is infrared light in this thesis). Because a food source may appear in an area that an ant has previously looked in, but discovered none, it makes sense then that the ant does not provide its location back to the other ants telling them that it found no food at this location because a source could appear later. This provides a basis for the surveillance approach at the start of the algorithm in how to search for the IR light as no paths which the agent moves through is communicated back to the swarm.

If food is found during an ant's search, pheromones are deposited by the ant leaving a trail from the food back to the nest as it carries it. As more ants sense the trail, they deposit their own pheromones, creating a gradient of chemical odor for the other ants to follow to the food source. As more ants deposit pheromones, this creates a stronger feedback loop to attract even more ants, and the rate of food carrying from the source to the nest increases as the size of the swarm to accomplish this task does. The concept of the pheromone and the food is synonymous in the robotic swarm where the "pheromone" is replaced by an agent's checking with each other who is the closest to the food source – the beacon. Instead of having all agents go to the beacon, only the closest one does.

This method of following a gradient to a source is an example of the nature of emergence. An individual ant is acting autonomously, and does not know the movements of other ants or even necessarily when other ants find the food, until itself is in a random location which is close enough to the gradient that the pheromone is now detectable. This is different in the case of the swarm of agents where once the beacon is located, the agent explicitly communicates this to the swarm. Nonetheless, the similarities between an ant and an agent is important because neither does the ant know its orientation within the colony or field, nor does an agent in this thesis know its global position in physical space. Furthermore, if an ant goes missing in the search or retrieval of food, this does not stop the swarm from continuing the task of transporting food back to the colony because the pheromone gradient still exists for the ants to sense, and thus the source of it, the food, does as well. Similarly, if an agent goes missing in the field, the mission can still continue because the signal is continually propagated from the beacon which other agents can receive. Because a pheromone trail evaporates over time, when the food source has been extinguished, ants are less likely to return to it, and therefore, less likely to lay more pheromones for other ants to follow. This decreases the number of ants going towards the original location of the food source, and because of this, the trail is eventually gone. Similarly, when an agent reaches the beacon, it messages all other agents that the beacon has been reached (keeping in mind again that this was the goal) and for the other agents to end their programs.

While the ant foraging strategy is continually evolving to be the most fit for the species in their current ecosystem at the present time, it is limited by its homogeneous method of movement in that ants are confined to land to scavenge for food. This means they cannot enter areas which contain food that are separated by water since pheromones would dissolve in it and an ant swarm would never be reached.

This problem can be solved by the introduction of heterogeneous robotics to overcome the separation of the beacon by water from the agents. While all agents are identical, one of them is designated for water and placed within a plastic ball, thus providing the capability to traverse land or water. This gives an advantage to the swarm in that the beacon can now be reached on any terrain (as will be shown in the physical design and results section in Chapter 7). Therefore, if an agent responding to the beacon determines that it would have to enter water to reach it, and it is not able to, it would use information from the message it received from the agent which originally located the source to then send a water-capable agent to it. This is advantageous from a design point of view because the agents all have the same cost, with the only additional cost of the ball itself - not an engineering redesign to the agent. This allows the swarm to traverse multiple terrains simply by placing one of the agents in an inexpensive plastic ball and downloading a file on the agent declaring it a water-agent.

3.2 Design Criteria

This section details the global overview of my thesis and will explain how swarm intelligence will be applied to heterogeneous agents navigating in an unknown area and across multiple terrains. The goal is to leverage each other's terrain-specific capabilities in any location and in closest proximity to a beacon to reach it, regardless of the terrain on which the beacon resides. This work is significant because most research, models, algorithms, and applications of swarm intelligence usually use or assume swarms are homogeneous. [37, 38, 39, 40] As discussed in Chapter 2, the potential for an agent to reach a beacon located in any terrain gives rise to a theoretically unbounded swarm in response to stimuli, regardless of location or terrain it is in.

To reach this end goal, the algorithm employed on each agent meets the following rigorous testing guidelines:

- At the start of all scenario tests, all agents within the swarm know their name and the number of agents there are in the swarm. Additionally, each agent has knowledge of whether they are assigned as a land or water-agent (through a pre-loaded text file declaring it as one or the other), but does not know another agent's terrain-capabilities.

- When any agent locates the beacon, it will assume the title of "Agent-1" and broadcast a message to the swarm instructing it to look for it to determine which agent is the closest.

- All agents then send their signal strengths, with larger values indicating closeness to the beacon, back to the Agent-1. Agent-1 then sorts the returned values, adding its own signal strength into the mix. The closest agent (which will be referred to as "Agent-n", since all agents have the same chance of being the closest), which has the largest signal, will be dispatched to the beacon. (This means that Agent-1 can also be Agent-n if it has the largest signal in the swarm.)

- If Agent-n discovers that it is separated from the beacon by a body of water, and it is not capable of traversing water because it is a land-agent, it will message the water-agent, to the beacon to head towards. The water-agent will drive from land into water, traversing toward the source, halting when it reaches a predetermined threshold distance to it. (The identity of the water-agent was made known to Agent-1 as packaged with its signal strength when it was sent back. Agent-1 then sent this information packaged to Agent-n in the same message instructing it to go to the beacon.)

- When the beacon has been reached, the responding agent will broadcast a message to the swarm notifying all agents of this. However, if the message is not received in an expected amount of time, the swarm will assume that either the agent has been lost, or the beacon has moved, and therefore reset and begin looking for the signal again.

- If Agent-n is removed from the field as it approaches the beacon, the swarm will reset after a clock within the algorithm expires. The value of the clock is based upon the number of agents in the swarm at the beginning and the speed of the agents, which anticipates how long it should take for Agent-n to reach the beacon and broadcast the message of its arrival to the swarm.

- However, if Agent-n is the water-agent that is removed, then the swarm will still reset, but when a land-agent reaches the body of water, it will broadcast to the swarm to shut down as the beacon cannot be reached.

3.3 Final System Design

The criteria outlined in the previous section have enabled the development of an algorithm to allow for the creation of a heterogeneous swarm capable of reaching a beacon on various terrains. It should be reinstated that if the robot does not reach the beacon in a time proportional to the speed of an agent and number of agents within the swarm (which will be shown by equations 5.3 and 5.4), then the swarm considers this agent lost, and the agents will continue in its absence by individually resetting their algorithm after the expiration of this time. It is the messaging by the agent which reaches the beacon to the other agents that informs the swarm that it has been reached. This message instructs the agents to end their programs.

Figure 3.1 shows the flowchart of how the entire system I have designed works. It describes how it works in every step, incorporating the criteria from section 3.2, from start to a finish.

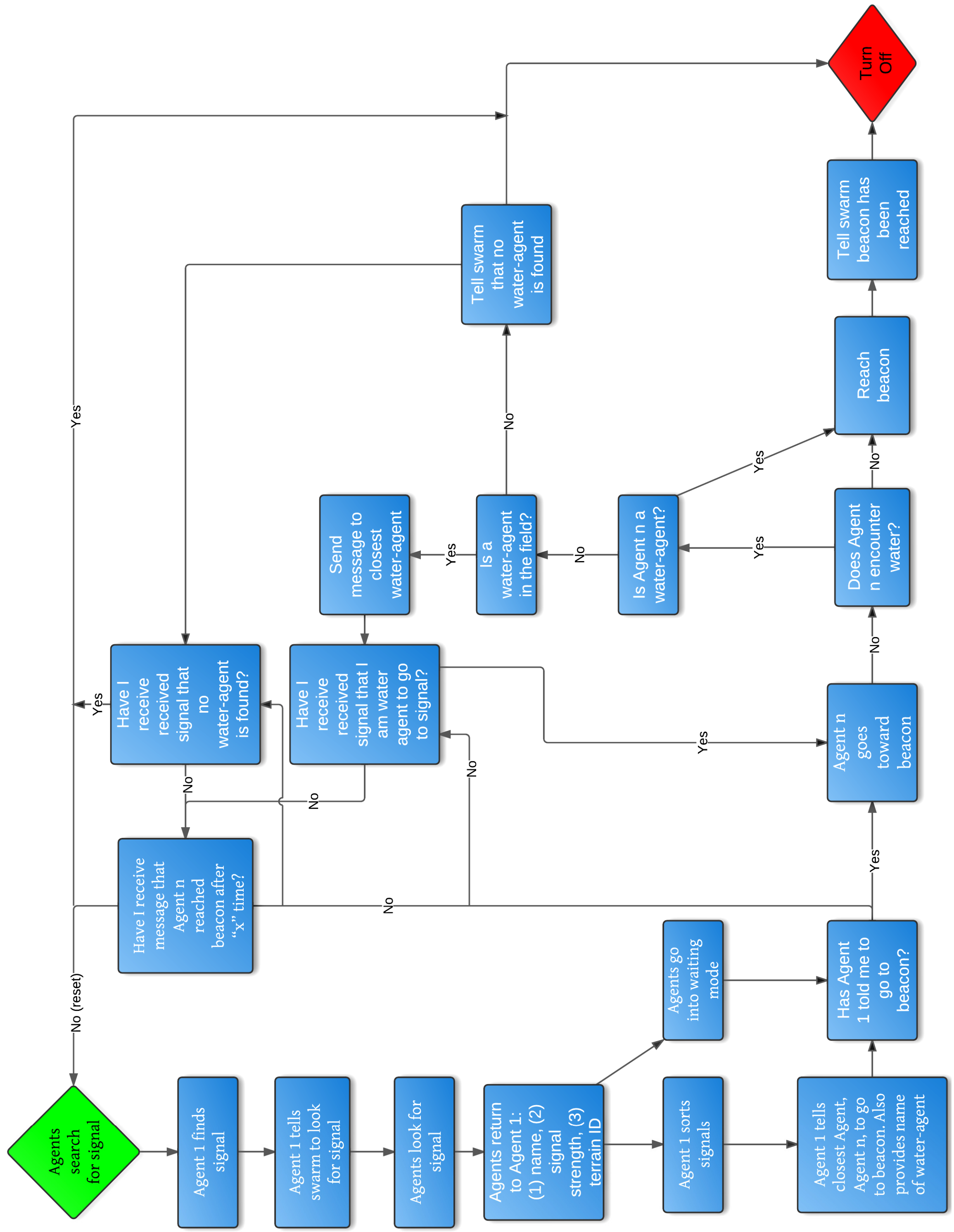


Figure 3.1: Flowchart for the heterogeneous swarm algorithm with goal of reaching the beacon.

3.4 Swarm Algorithm Review

This algorithm demonstrates the possible scenarios the swarm of agents may encounter to reach the beacon. It outlines the paths which it can take to reach the beacon and how to message the swarm when this occurs. If time-expires when waiting for a response from Agent-n that it reached the beacon, the algorithm will reset on each agent. It also shows how the only route to failure is by a land-agent encountering water as it heads toward the beacon, and determining there is no water-agent in the field, for which it instructs the swarm, and itself, to end its program. When an Agent must message the entire swarm, it does so by broadcasting it to them. While figure 3.1 displays when these actions occurs, the broadcast algorithm itself will be shown in depth in Chapter 5.

Chapter 4: Physical Robotics and Sensors

This section outlines the physical components of the system and how they were programmed to work together. Pictures of the agent are shown, with sensors labeled for reference. The plastic ball used for the water-agent is shown with the agent inside.

4.1 NXT Robotics

The robot hardware is the NXT brick, from the LEGO Group, as are the pieces which its physical body is built from, as shown in Figure 4.1. The sonar [53] and light sensors [54] are also from LEGO, while the EOPD [44] and Infrared Seeker [41] sensors made by HiTechnic. I programmed the algorithm using LabVIEW by National Instruments. The ball consists of two 40 gauge plastic domes, made by Barnard Ltd. [42] Specific product specifications are outlined in the respective sections, as are details of construction and implementation of system and rationale for design where appropriate. The infrared ball is made by HiTechnic. A LEGO light sensor was used for obstacle detection for the water-agent, while sonar was used for the land-agents. Two Servo Motors, also made by LEGO were used for the motor attaching the wheels. Programs developed for the NXT are downloaded to the brick and are run remotely (in Remote Mode). There is an option to control the NXT from the computer if desired, in what's called "tethered' mode. (For complete technical specifications for components used in the swarm, see Appendix A.)

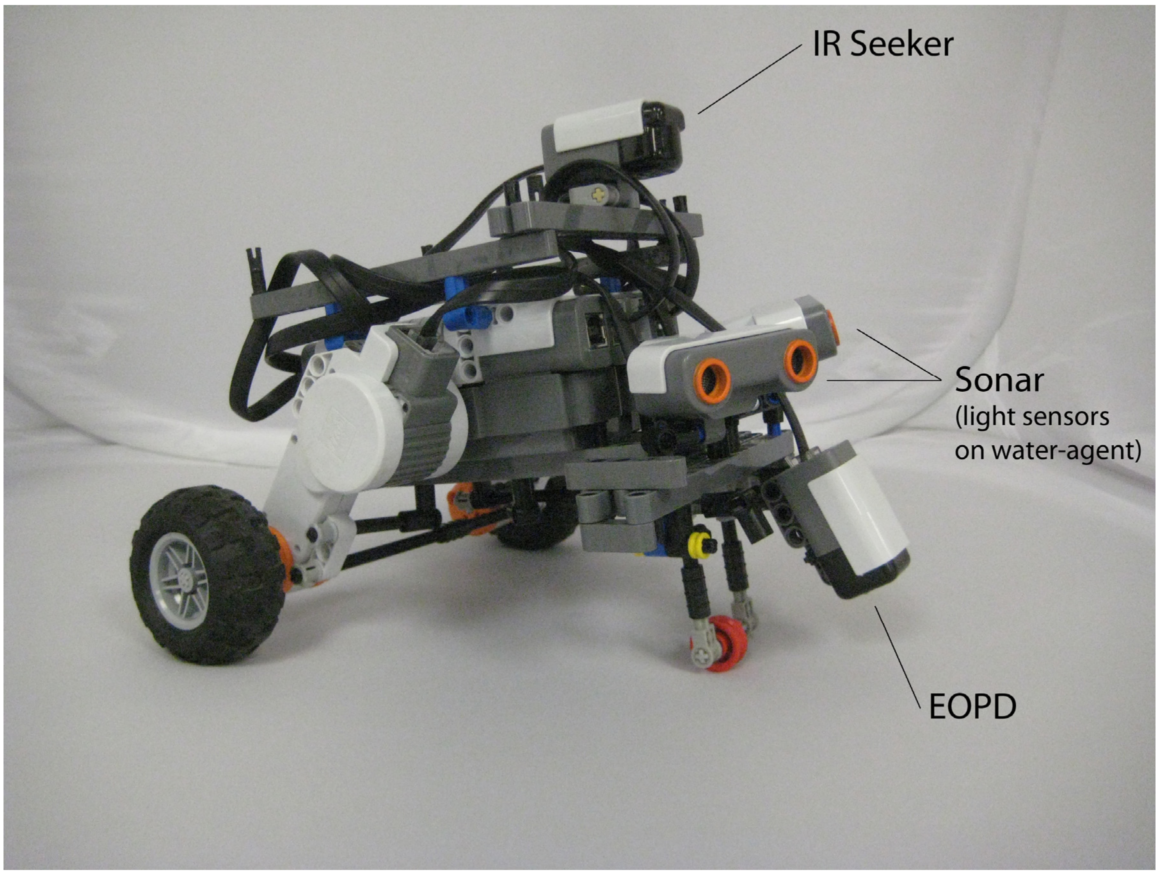


Figure 4.1: Land-agent and associated sensors. The land-agent uses two sonar sensors for obstacle detection, while the water-agent uses light sensors.

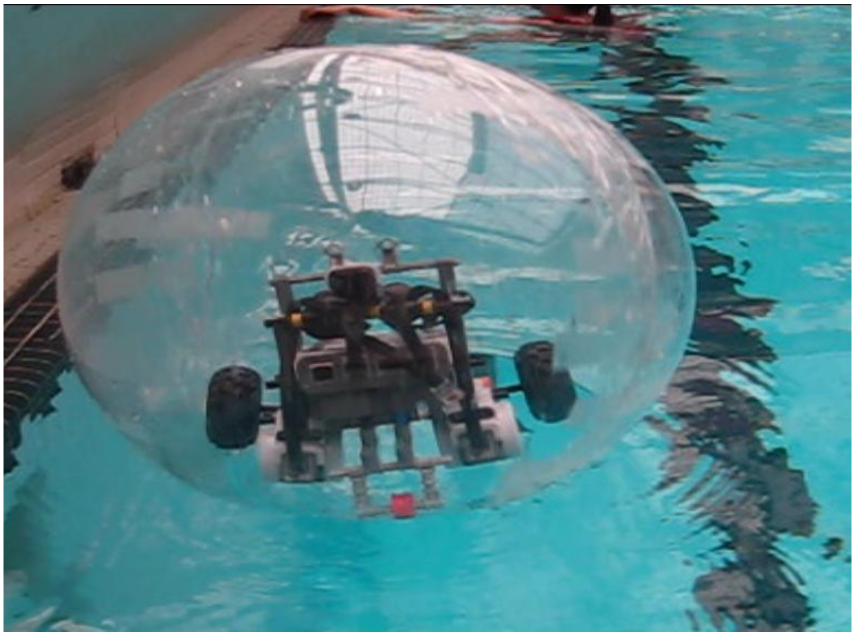


Figure 4.2: Water-agent inside the plastic ball on water.

4.2 Infrared Sensor

The infrared sensor, called the IR Seeker V2, was set to receive frequencies tuned to 1200 Hz square wavelengths modulated at this frequency from the IR Ball. It uses advanced DSP to filter other IR signals, thereby decreasing the risk of interference from artificial light or sunshine. The sensor comes equipped with a 270^{o} viewing angle, which is split into 9 directions, allowing for detection of the direction a signal is received from, as shown in figure 4.3. (A value of 0 is returned if no signal is received.) The IR sensor was set to detect AC signals.

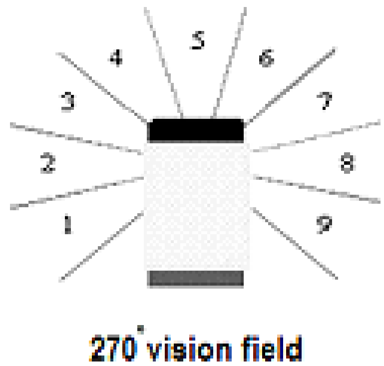


Figure 4.3: Top view of the IR Seeker. The sensor contains 5 receivers internally, and between them, they are divided into 9 channels, spanning 270^o for the receiving range. [41]

A direction of a “1” means a signal is to the left of the sensor, while a “9” means it is to the right, with 5 being the center channel. If a signal is detected in the vision field when an agent is in pursuit of the beacon - the IR ball - the agent’s behavior is set to orient itself to the center channel so that the signal is straight ahead. (A rigorous analysis of the IR Seeker and its capabilities and limitations to receive infrared light as a function of distance is presented in Chapter 6.)

4.3 Ultrasonic Sensor (Sonar)

The sonar sensor, shown in figure 4.4, estimates the distance between the senor and the closest object from it and communicates with the NXT over i2C communication protocol to allow for digital communication from the physical environment. It has a range of 0 to 100 inches (~255 cm), +/- 7.6 in (3 cm) precession.

Figure 4.4: Ultrasonic Sensor. [53]

Two sonar sensors were placed 80^{o} from each other on the front of the agent to act as an obstacle detector. Change in a value below a predetermined threshold in either sensor would signal the agent to change its behaviors to avoid the collision with whatever object lay ahead in the agent's path. The results of indoor testing, as shown in figure 4.5, shows a linear relationship between the distance an object is from the sensor and that which actually was recorded by it. Because the actual size of the indoor testing area was 9ft × 14ft, the agents were programmed to only avoid each other, other obstacles, or a wall if they encounter at approximately 8 inches from it.

Testing showed that at 0 inches from an object, the sensor registered a value of 5 cm, or just under 2 inches. However, this result does not affect the linear relationship of the sensor versus distance an object is from it, as shown in figure 4.5, but may indicate noise in the sensor when it is very close to another object. However, this finding is of no consequence since the agent will avoid an object if the sonar's value falls under the threshold of 8 inches.

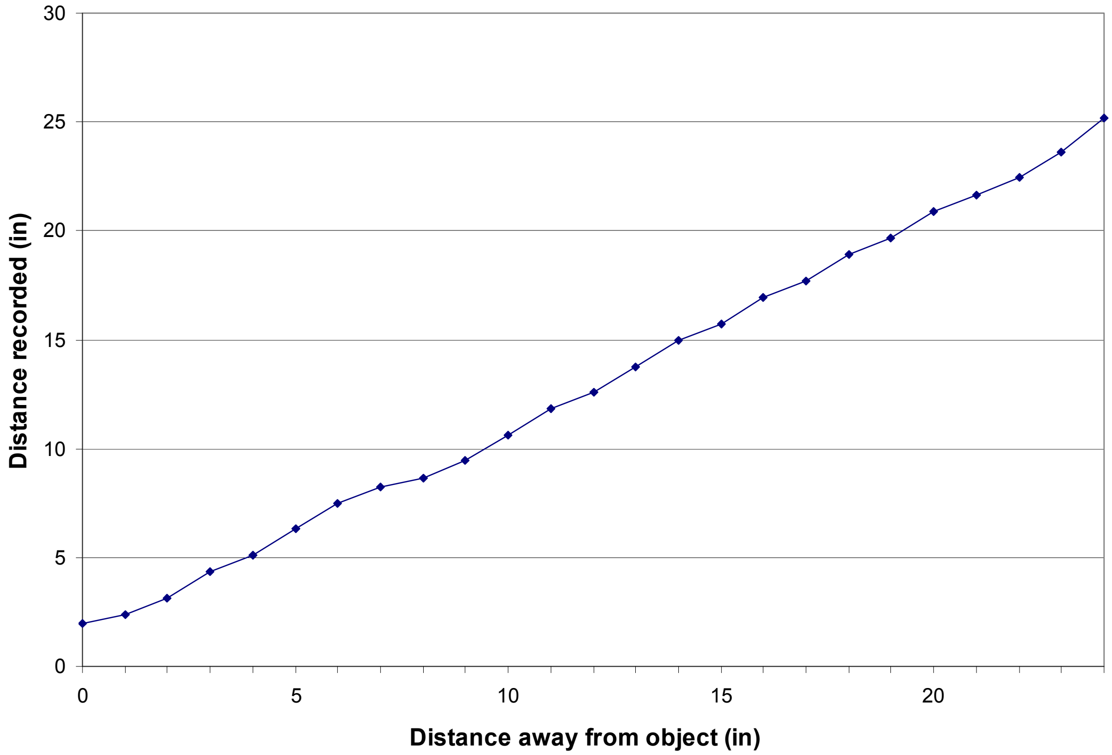


Figure 4.5: Actual distance an object was from the sonar, as tracked by tape measure vs. distance which the sensor recorded it was.

4.4 Light Sensor

The light sensor, shown in figure 4.6, is used as the collision detector for the water-agent within the plastic ball. Because the ball is transparent, light propagates through the plastic ball just as ambient light does.

Figure 4.6: Light Sensor. [54]

The light sensor is an analog input, has a sampling rate of 3ms, and is coupled with a receiver on the top. It was set to read the reflection of red light which it emits from its internal LED and tested in a room containing ambient light.

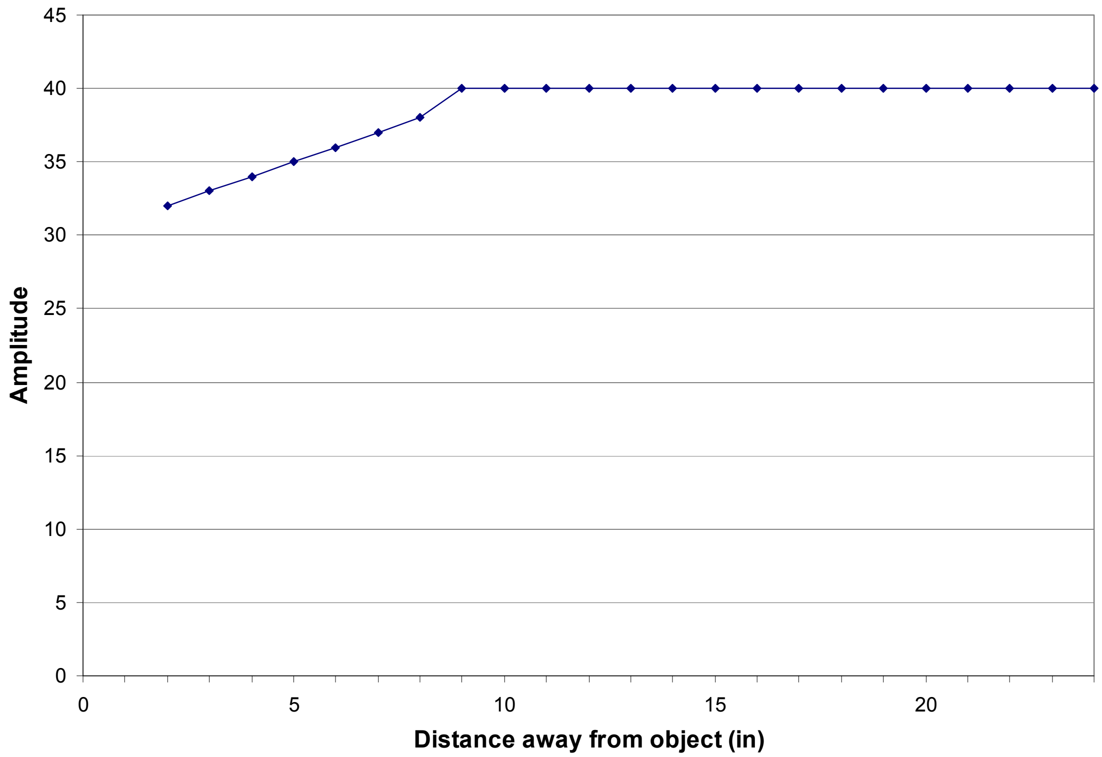


Figure 4.7: Distance light sensor is from an object, as measured from within the plastic ball of water-agent.

Because the EOPD sensor is located 2 inches in front of the light sensor, the closet the light sensor can get to the edge of the ball is this distance. This is represented in figure 4.7 where data begins to be recorded at the 2 inch mark.

4.5 EOPD Sensor

The Electro Optical Proximity Detector (EOPD) sensor emits pulsed red light with the purpose of eliminating interference of ambient and/or background light from the sensor's receiver through onboard signal processing to filter non-pulsed light which is returned to the internal receiver. The sensor works by reading the light measurement at the detector prior to pulse emission minus the light measurement during it. This difference is a direct measurement of the reflected light energy after removing any dependence on ambient light. The EOPD performs data sampling at 300 samples per second. Internally, the EOPD sensor actually samples between 350 to 400 data per second to ensure there is always a new reading for the NXT each time a value is called. (A rigorous analysis of the EOPD sensor used to distinguish between different terrains based on reflectivity of light directed incident upon them, as well as transient responses to these terrains are presented in Chapter 6.)

Figure 4.8: EOPD Sensor. [44]

4.6 Servo Motors

Two Servo Motors were used to drive the wheels on the NXT. These motors are capable of 170 rmp under no load, and have an accuracy of $^{+}/_{-}$ 1^{o} of rotation. [45]

4.7 IR Electronic Ball

The IR ball, show in figure 4.9, is used as the beacon in this thesis. The ball is set to pulse at 1200 Hz to match that of the IR Seeker which receives at the same rate. There are 20 infrared spectrum LEDs symmetrically positioned on the ball.

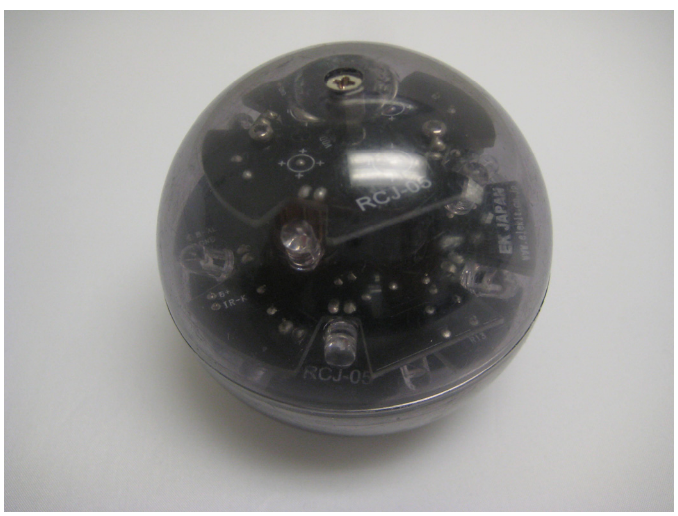


Figure 4.9: Infrared ball.

4.8 Plastic Ball

This plastic ball consisted of two domes, each made of 40 gauge plastic. An agent was placed between the domes, and then the edges were taped together to form a ball, as shown in figure 4.2 displaying the water-agent. The plastic is elastic and deformable in that if something bounces against it, it can deform and bounce back to retain original form under moderate impulses.

All agents tested used a forward linear speed of 0.28ft/s. While the water-agent was used on the floor in the tested scenarios which will be presented in chapter 7, it had demonstrated ability to rotate in water with an angular speed at 50deg/s, while the land-agent rotated at 71deg/s when tested on the floor. These rotation tests were performed by aiming the agent's IR Seeker's direction 1 at the IR ball, and then allowing it to rotate 270° until the direction 9 received signals from the IR ball.

The water-agent operates by using momentum inside the ball as well as friction on the exterior surface of it. It showed that it can traverse both loose gravel and across water, as well as on grass and asphalt. The water-agent is able to move on low-friction or unsteady surfaces since the front-to-back length of the robot's wheels is less than that of the ball's diameter. This design feature allows for the use of momentum to get the ball moving by taking advantage of the irregularities of the ball and the increased surface area of it, which would allow for better traction on these surfaces than that provided by the two rear tires of the land-agent. Additionally, the ball is also capable of deformation and elastic recovery, and thus, it can take small impacts and still retain its form.

4.9 System Overview

Each agent is built using the same components and sensor, with the exception of the water-agent which uses the light sensor for collision-avoidance. The accuracy of the light sensor to detect an obstacle will vary widely due to the imperfections in the shape of the plastic ball with the agent rolling within. The two domes must be taped together, which is difficult to do because the dome is not perfectly symmetrical with each other, in that one dome was measured to be approximately one-half inch smaller than the other. This means there are gaps when the lips of the domes are placed atop each other, and occasionally creates difficulty when the water-agent rolls within as the shape of the ball is not a perfect sphere. At times, the agent rolls up the side of a sphere when it is moving, but will drop quickly drop due to the weight of it as it pushes the ball forward. This prevents the collision detector to remain horizontal, and thus, it can miss objects with such occurrences.

This upward swing of the agent in the ball also affects the IR sensor and the EOPD, too. The IR sensor may miss or loose the IR beacon if this occurs. Furthermore, if the pitch of the agent is large, this would affect the EOPD readings to incorrectly identify water due to low light reflectivity as the pitch angle increases with respect to the surface. The only possibility of an issue here would be if the water-agent is heading towards the beacon on land, and because of this, it incorrectly reads “water” when it is still on land. When the agent perceives water, it outputs the drive to full speed. This could cause the water-agent to roll over if it swings upward on full power, resulting in a failure in the field.

This upward swing is small when the water-agent is rolling. This creates a small “rocking motion”, and acts as momentum when it rolls across surfaces. This can help both in traversing through water, as well as turning in it. While it was not tested on very low friction surfaces, such as ice or oil, the rocking motion could help to propel it forward by using the generated momentum.

Chapter 5: Communications

In this section, I describe how communication between agents within the swarm occurs. In order for inter-swarm communication to succeed, it's important that agents use a dialogue which is robust, efficient, scalable, and practical. While I have discussed several requirements to building the algorithm in relation between a swarm of ants and the swarm of agents, there exists a few distinct differences between them. While the algorithm is similar to ant colony behavior, the criteria for implementation is different in that using explicit messages employing a dynamic ad-hoc mesh protocol where any agent has the ability to inform other agents that it has found the beacon, or if a land Agent needs assistance crossing water, requiring the closest water-agent to be called. This broadcast messing protocol allows for freely-associated connectivity between any agents, allowing for a decentralized swarm, where an agent can temporarily become the message broadcaster. Because there is no designated operator, the loss of one agent does not impact the ability of the swarm to communicate and operate to complete the goal of reaching the beacon.

5.1. Leveraging Ant to Artificial Swarm Communications

Many research studies have analyzed the effectiveness of ants to act within swarm behavior and examined how they communicate. [5, 10, 29, 30]

Ants use an implicit form of communication, or *stigmergy*, which is an indirect method of communication by a change in the environment by one agent in which others can detect. With ants, this is carried out through the deposit of pheromones to signal other ants about presence of a food source across a gradient to it. While implicit communication may work well for ant swarms, the environment which the swarm operates in should be the deterministic factor of which form of communication to use, rather than declaring the mode of communication as the best option for any swarm just because it works well in a specific environment or has similar behavior, such as in an ant colony. The main reason for this is that the ant colony consists of many ants living and working in very close proximity to each other, which allows implicit communication to work well because it can reach a large number of ants quickly. In practical applications, such as in a military scenario, explicit forms of communication would be the preference since adversaries could intercept signals through stigmergy, and thus, you only want to communicate directly with specified, indented receivers of the information. While it can be variable based on surroundings, the NXT uses Bluetooth, which has a limited range of approximately 10 meters. [43] However, because agents can theoretically be at any location, explicit forms of communication are indeed practical for swarm robotics, as well as exchanging specific information about the agent and its environment to others.

5.2 Dynamic Ad-Hoc Mesh Network

Because autonomous swarm intelligence consists of a decentralized control hierarchy, an ad-hoc network is an appropriate choice for this type communication. This type of wireless network allows any agent to make a connection with any other agent within the swarm, without a predefined access point (such as a central leader, or *master* agent).

This is effective because each agent's Bluetooth address acts as a single node which is not connected in a path to any other nodes, nor does it act as a router, unless the agent is the one who discovers the beacon. The agent who makes the discovery, called "Agent-1", broadcasts information about its signal strength to all other agents to make comparisons using their own observed values. In order to scale the size of the network, The Agent-1 creates a link with the second agent, Agent-2, in a numerically-assigned predetermined list of n-agents in the swarm (where Agent-2 is one of n-1 agents within the swarm). In order to effectively broadcast the message, both agents break the link on their own ends; Agent-1 breaks it after the message is sent, and Agent-2 breaks it after it receives the message. After Agent-1 and Agent-2 have broken the link, it then connects to the next agent in queue, which is Agent-3, and so forth, until a total of n-1 messages have been delivered in the swarm.

To get the dynamic ad-hoc mesh network working, it was determined through experiments using the Bluetooth on the NXT hardware for timing that the time within a link will be 8 seconds, requiring 6 seconds to wait after initiating a connection to an agent, and 2 seconds to wait after initiating to disconnect.

When Agent-1 makes the connection to another Agent, it essentially "drops off" the message while it moves onto to the next agent in queue. While Agent-1 is now messaging Agent-3, Agent-2 is looking for the signal. This significantly saves time because Agent-1 does not have to pause and wait for Agent-n to look for the message and return it before moving on to the next agent. Agent-1 drops off all the messages, and then the agents return their messages to Agent-1 in the order in which they were received.

Figures 5.1 shows the dynamic ad-hoc mesh network in detail. As can be seen in figure 5.1(A), once Agent-1 has located the beacon, and has made a connection to Agent-2 to send this information to it, the bidirectional path from Agent-2 to the swarm has been replaced by a unidirectional path from the swarm to Agent-2. This occurs because when Agent-1 "drops off" the message to Agent-2, instructing it to spin and look for the signal, this also ends Agent-2's surveillance search mode (as shown in the first row in figure 5.1) so that it is now only spinning and looking for the signal. This was performed so that it would not be possible for Agent-2 to locate the signal, and try to connect to the other agents, as Agent-1 did. After Agent-1 disconnects from Agent-2 and repeats this step with Agent-3, it waits for the agents to then return their signals, finally ending in figure 5.1(E). After Agent-1 has received all of the swarms' signals, Agent-1 will then sort the signals, add its own into the mix, and then determine which agent is the closest to the beacon, messaging this agent to go forward.

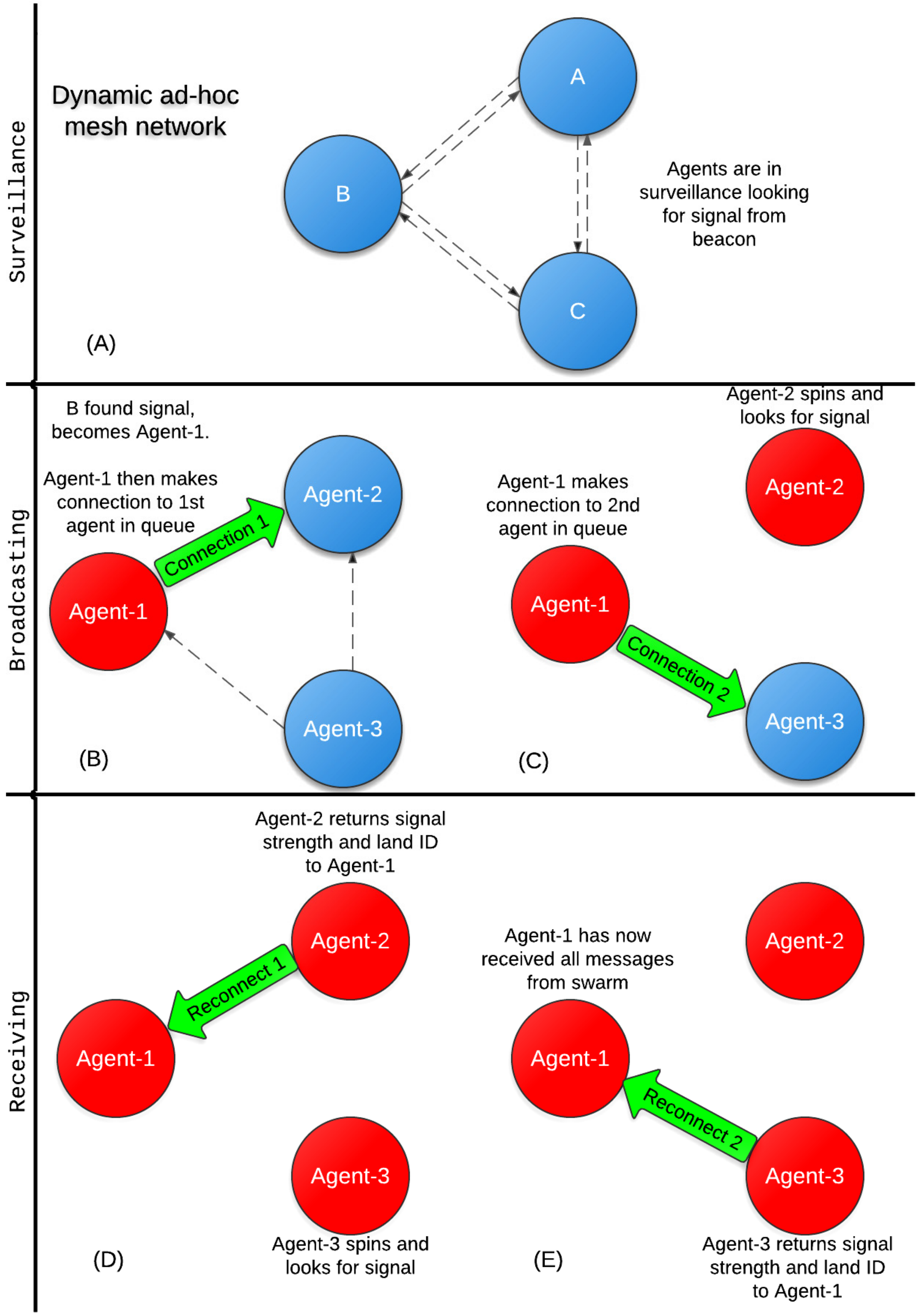

Surveillance
Dynamic ad-hoc mesh network
A
B
C
Agents are in surveillance looking for signal from beacon
(A)
Broadcasting
B found signal, becomes Agent-1.
Agent-1 then makes connection to 1st agent in queue
Agent-1
Connection 1
Agent-2
Agent-3
(B)
Agent-2 spins and looks for signal
Agent-1 makes connection to 2nd agent in queue
Agent-2
Agent-1
Connection 2
Agent-3
(C)
Receiving
Agent-2 returns signal strength and land ID to Agent-1
Agent-2
Reconnect 1
Agent-1
Agent-3
Agent-3 spins and looks for signal
(D)
Agent-1 has now received all messages from swarm
Agent-2
Agent-1
Reconnect 2
Agent-3
(E)
Agent-3 returns signal strength and land ID to Agent-1

Figure 5.1: Dynamic Ad-hoc mesh network for broadcasting to agents when the beacon is located. (A): Agents are roaming in surveillance mode, with no connections between them, and are arbitrarily named 'A', 'B', and 'C'. (B): Agent-B has spotted the beacon, and is now referred to as 'Agent-1'. Agent-1 is now sending information to Agent-2, the 1st agent in queue to receive the message, to look for beacon. (C): Agents-1 and 2 have broken their link; Agent-1 is now sending to Agent-3 the command to look for beacon. (D): Agent-1 and Agent-3 have broken their link; Agent-2 is returning its signal strength it found to Agent-1. (E): Agent-1 and 2 have broken their link; Agent-3 is returning its signal strength it found to Agent-1.

This method works because all agents have equal capabilities to be either a temporary master or slave, without designating one as a full-time central router. However, because any agent has an equal chance of finding the signal, any link between two agents is broken following transmission and reception of data, thus creating a decentralized network of agents.

5. 3 Mathematical Modeling of Communications

Each part of the communication between agents is followed by a "waiting" time until another event occurs which are described by equations 5.1-4. For example, equation 5.2 describes how after Agent-1 broadcasts to the swarm the command to look for the beacon, it waits a time proportional to the number of agents within the swarm before it sorts the returned messages to determine which agent is the closest to the beacon before sending this one forward.

5.3.1 Agent-n Signal Return Time to Agent-1

This is the time required by any agent within the swarm to return its information to Agent-1 after it receives the message to look for the signal. This information is comprised of the following:

- Agent-name
- signal strength of beacon
- terrain ID (land or water)

The equation which describes this time is the following:

$$T_{ANW} = t_n \times (n - 2) + (n - m - 1)\ ,\ n > 1\ \ ,$$

(5.1)

where T_{ANW} represents the time required by any agent within the swarm to return its information to Agent-1. When the last agent in the swarm, m, has received the broadcast, given by m = n – 1, the first to receive it returns its information to Agent-1, as shown in figure 5.4.

Note that the timing is important because the Bluetooth does not have a busy signal function built in, so that if an agent attempted to contact another while it was in a link with a different agent, the message would be not received.

5.3.2 Agent-1 Wait Time for Returned Signals from Swarm

Agent-1 uses equation 5.2 in determining how much time it should wait in order to receive all of the messages in return from the swarm following its broadcast to it to announce that the signal has been found, as shown by the following:

$$T_{A1W} = t_c \times (n - 1) \text{ , } n > 1 \text{ ,} \qquad (5.2)$$

where T_{A1W} represents the time Agent-1 should wait until all agents should have returned signal strengths. Note that t_c was determined experimentally so that all agents were able to return their signals to Agent-1 consistently.

While the exact time to return a message can slightly vary due to uncertainty in the time it takes to execute commands on the agents, equation 5.2 has a safety time factor of approximately 1 second for n = 2, and approximately 6 seconds for n = 3 (which was the maximum number of agents tested).

It takes Agent-1 9 seconds to send the message to Agent-2 that it found the beacon for n = 2, and 18 seconds when n = 3, which was expected.

5.3.3 Agent Wait Time for Reached-Beacon Notification from Agent-n

Equation 5.3 describes how long an agent should wait to receive a message from Agent-n that the beacon has been reached. Note that equation 5.3 takes into account the possibility that a land-agent was initially responding to the beacon, but encountered water in its path, and thus needed to make a call to a water-agent to traverse toward the beacon instead, as shown by the following:

$$T_{RBN} = t_n \times (n - m) + (n + m - 1) + \frac{(IR_{stop} - IR_{start})}{S_{fwd}} + BT_m + BT_{bbr}, \; n > 1, \tag{5.3}$$

where T_{RBN} represents the time agents (excluding Agent-1) should wait to hear from Agent-n that signal has been reached before resetting. If the responding agent to the beacon does not reach it in this amount of time clocked by Agent-n, then Agent-n will reset its program to look for the beacon again.

5.3.4 Agent-1 and Agent-n Reset Time Waiting for Agent-n

Agent-1 will reset if it has not received a signal from the responding agent to the beacon in a time described by equation 5.4. It also represents the time to reset for

Agent-n, which would occur if the beacon has moved further away during its traversal towards it. These events are shown by the following:

$$T_{A1R} = T_{ANR} = T_{RBN} + BT_m , \tag{5.4}$$

where T_{A1R} represents the time until Agent-1 should reset if it has not received a message from Agent-n that it reached the beacon. Equally, T_{ANR} represents the time that Agent-n should reset after if it has not reached the beacon yet. If a signal is not located, or is has moved, agents must know when to begin looking for the signal again. In order to do this, all agents would have to restart their programs.

5.4 Communication Weakness

The dynamic ad-hoc mesh communications I have shown is robust using the NXT robotics for swarm intelligence allowing for a 78% success rate, as will be further discussed in chapter 7. The global option would be good because if one agent located the signal, it should tell the others to stop looking for it as quickly as possible. Because it takes 9 seconds to full connect and disconnect from an agent after Agent-1 begins the broadcast that the signal has been located to the other agents (1 second wait before each send to an agent in the programming loop + 8 seconds to connect-disconnect), there is a chance that if another agent is also close to the signal, then it could also become Agent-1 and thus the swarm would not operate correctly. This problem has been minimized by using a threshold value from the beacon, thus reducing the chance that all agents in the field would locate beacon.

This works well because it will run, even if no agents respond. For these reasons, these models are appropriate for estimating the response time of the swarm and accurate enough because they account for the worst-case scenario where if all agents but one were missing, the remaining agent could still accomplish the task. The one caveat would have to assume that the remaining agent was a water-agent if there was water separating the agent from the beacon.

Instances of the swarm not communicating are likely due to a failure of link communication between two agents when one is transmitting and while the other has encountered a system timeout and therefore is busy and did not receive the message. This is because the NXT is prone to dropped messages because the 'send' ('NXTCommBTWrite') and 'read' ('NXTMessageRead') commands share the same radio. Because the default timeout for 'read' is set at 200ms, you need to increase the 'send' timeout from the default of 50ms to a number greater than 200ms in order to outlast the 'read' which is occupying the radio. Using 210ms for this new value resulted in a 78% connection rate using the dynamic ad-hoc mesh protocol. Because an agent will be checking for a message which is sent to it more often than it will be sending one, it is more logical to lengthen the send timeout rather than shortening the read.

The other fallacy which may occur as a part of the 22% failure rate of connection is when an agent has tried to disconnect from the link with another agent, but it was unsuccessful. These issues may point to problems related with the firmware of the NXT, rather than the methods which were designed to accomplish link communication between agents.

5.5 Swarm Communication Review

The equations I presented have allowed the swarm to operate as outlined in the flowchart shown in figure 3.1. The ad-hoc broadcast protocol I developed allows the system to use an n-dimensional swarm, with the only theoretically limitation of it based on the range of Bluetooth radio communication, which reflects a limitation on the hardware, not in the design of the swarm algorithm itself. However, equations 5.3 and 5.4 have a few uncertainties which are not taken into account:

1. If a land-agent responds to the beacon, and encounters water, the water-agent will take over, and head towards it. However, because the threshold is the minimum any agent must be from the beacon to initially receive it, a water-agent could be anywhere in the field recording a signal strength value less than the threshold. In other words, the water-agent can actually be a further distance away from the beacon than the threshold is set at, but

the time it would take to drive to the beacon, in this case, would be larger than the time component in equation 5.3,

$$\frac{(IR_{stop} - IR_{start})}{S_{fwd}}.$$

This only denotes the estimated time the responding agent located at the minimum threshold would need to reach the beacon.

2. Because the component in equation 5.3,

$$\frac{(IR_{stop} - IR_{start})}{S_{fwd}} + BT_m,$$

describes the time needed for an agent to traverse to the beacon from the minimum threshold, plus the time required to message a water-agent if a land-agent encounters water, it does not say how much time would be spent on land versus how much time would be spent traveling in water during the overall trip. While the speed of the water-agent inside of the ball is faster than that of the speed of a land-agent, net drag on the ball from the water is not accounted for; friction on the ground and slipping are not either.

3. This formula does not take into account the possibility of agents entering into a blockage state, where they are having difficulty traveling around each other. Because non-responding agents to the beacon remain

stationary, if an agent becomes trapped near a board which contains the field, it could have difficulty navigating around it to clear itself.

These uncertainties of time are due to the fact that an agent does not update the other agents where it is in the field, or what terrain it is currently on. This is because NXT Bluetooth communication connectivity would act too slowly if an agent were to send broadcast updates to other agents in the field requesting more time. However, the swarm, which was tested with 3 NXTs at a fixed maximum distance from the beacon (which is described in detail in Chapter 7) was able to have an agent reach it in 42 out of 50 trials, for an 84% success rate. This shows that the aforementioned factors are limiting, and would only represent a small number of cases where the equation would have an expired time for restarting. Because of this, the system would still actually reach the beacon, since the restart allows for such possibilities, and therefore, this demonstrates that the applied equations enable a robust swarm.

Another uncertainty within the equations is due to the algorithms varying in time to complete processes. Because of this, experimental time adjustments need to be made at specific parts of the algorithm, specifically when the agents are clocked to reset. This is important, because you do not want an agent to reset too far ahead or behind the other agents. This is because if an agent hasn't reset by a certain time, and another agent who has reset finds the beacon and broadcasts this to the swarm, the agents who haven't reset won't receive the message. This problem with resetting could be fixed easily on another platform which allowed for spread-spectrum broadcasts of a reset command and accurate software timing capabilities and faster connectivity between agents or with multiple access channels.

Chapter 6: Sensors and Data Analysis

In this chapter, I explain how I tested the EOPD and IR sensors on each agent. The EOPD sensor was tested and analyzed to measure data accuracy and reliability of surface recognition. The results of these tests are important for the agent so it can ultimately differentiate between land and water surfaces. The IR Seeker was tested to determine accuracy of the distance an agent is from the beacon upon infrared signals generated from the beacon. Data was gathered by each sensor and then the results were analyzed to determine how each agent should respond to specific threshold values set by the sensors, and what the behaviors of the agent should be at this value. For example, land-agents cannot enter a terrain if the reflectivity is below the threshold, which would indicate it is water, and hence a hazard to the land-agents. Both the EOPD and IR Seeker values were experimentally determined and had to be calibrated before trials. Uncertainties with each sensor are discussed following the analysis.

6.1 EOPD

The agents used the EOPD sensor to differentiate between land and water. Identification of water enables land-agents to avoid entering a body of water (which would destroy them), but allows water agents to enter it. Also, as will be discussed in Chapter 8, identification of water by a water-agent traversing towards the beacon allows it to enter the water and signals it to increase its speed to traverse it, greater than that which it travels on land to help with agent mobility within the ball to overcome drag forces.

As shown figure 4.1, the EOPD sensor was mounted in the front of the agent, at approximately 50^{o} incident to ground. This angle was experimentally determined to allow for an agent to view terrain ahead of where it is traveling. Larger angles would not allow for proper measurements as higher angles reduced the reflected signal strength, making it difficult to differentiate between asphalt and water (especially when rolling in the ball), and smaller ones did not allow enough time to signal the robot to stop if it reached at water's threshold value, as the EOPD sensor was reading measurements too close to the front wheel.

In order to have the agents differentiate between terrains, I set agents stationary on them and recorded the amount of light reflected at different times of the day: midday, evening, and at night. The times of day for sampling data were chosen specifically for the infrared testing, but were also performed at these times with the EOPD for consistency. The reasons for such decisions will be explained in the next section.

Transient response tests were also performed to see how fast an agent could recognize a new terrain when it was suddenly introduced to it. This was conducted by holding agents 5 feet above the terrain, and then lowering them onto it in 1 second. Because it is vital for a land-agent to be able to recognize and distinguish between land and water, the settling time which it takes to recognize water is the most important find of these tests. This time can then be used to find the theoretical maximum speed an agent can travel.

6.1.1 EOPD Terrain Testing Results:

All data were sample at 3ms per sample over 15 seconds and was taken across all surfaces.

Stationary Surface Tests

As shown in figure 6.1(A, B, C), there is a noticeable pattern of the mean reflectivity amongst the surfaces when the agent is tested in a stationary position. The range of each surface on all three charts shows a small deviation from the mean, indicating that the mean of the terrain is representative of a typical value measured from it for when an agent is not moving, nor is the environment (at least noticeably enough to alter the reflectivity pattern, such as could occur with wind gusts). Grass provided the largest change in amplitude of all surfaces, both between time of day and between agents. This is partially due to the sensitive EOPD sensor gathering different reflectivity values at small changes in the pitch of the agent on the grass, which is a function in the irregularities of soil height below the grass blades. Additionally, the grass tested was on winter grass, and thus the grass blades were not all the same color, as some were green, while others were a light, tan color. Because the EOPD is very sensitive to height changes, such as small differences in sampling heights of above terrains, and since random colored grass patches occurred during sampling, any differences in reflectivity is an expected data result.

The main consistency is shown in the amplitude of water, with values much lower than all other surfaces. This is because while the other surfaces only reflect and absorb light, water is the only medium which light is also refracted when the light is incident upon it. Therefore, less light will be reflected back to the EOPD than with the other surfaces sampled. Exceptions to water refraction would be in the case if a shallow body of water is measured, such as that of the gradual incline when entering into a lake or ocean. In swallow water, refraction would be small, and thus much of the light would be reflected, therefore reducing the ability to identify water.

One added variation of the grass data resulted from the emitted light penetrating between the grass blades. While part of the light beam will be absorbed by the dark soil, some of the reflected light gets trapped on the underside of the grass blades by the angle the EOPD is at, as well as by the soil which the light reflects off. Agent-1's measurements during the midday and night samplings recorded amplitudes more than half of the evening sample. Because height of EOPD on grass patches, pitch angle, and grass-blade density cannot be controlled for, these findings are considered normal variables in the sampling. While gravel itself is rough and randomly shaped, the average scattering of gravel could be estimated to return a similar amount of light due to packing of the pebbles. Exceptions to this can be seen in the night sampling in figure 6.1(B), and the evening sampling in figure 6.1(C), in which a loose scattering of the pebbles probably

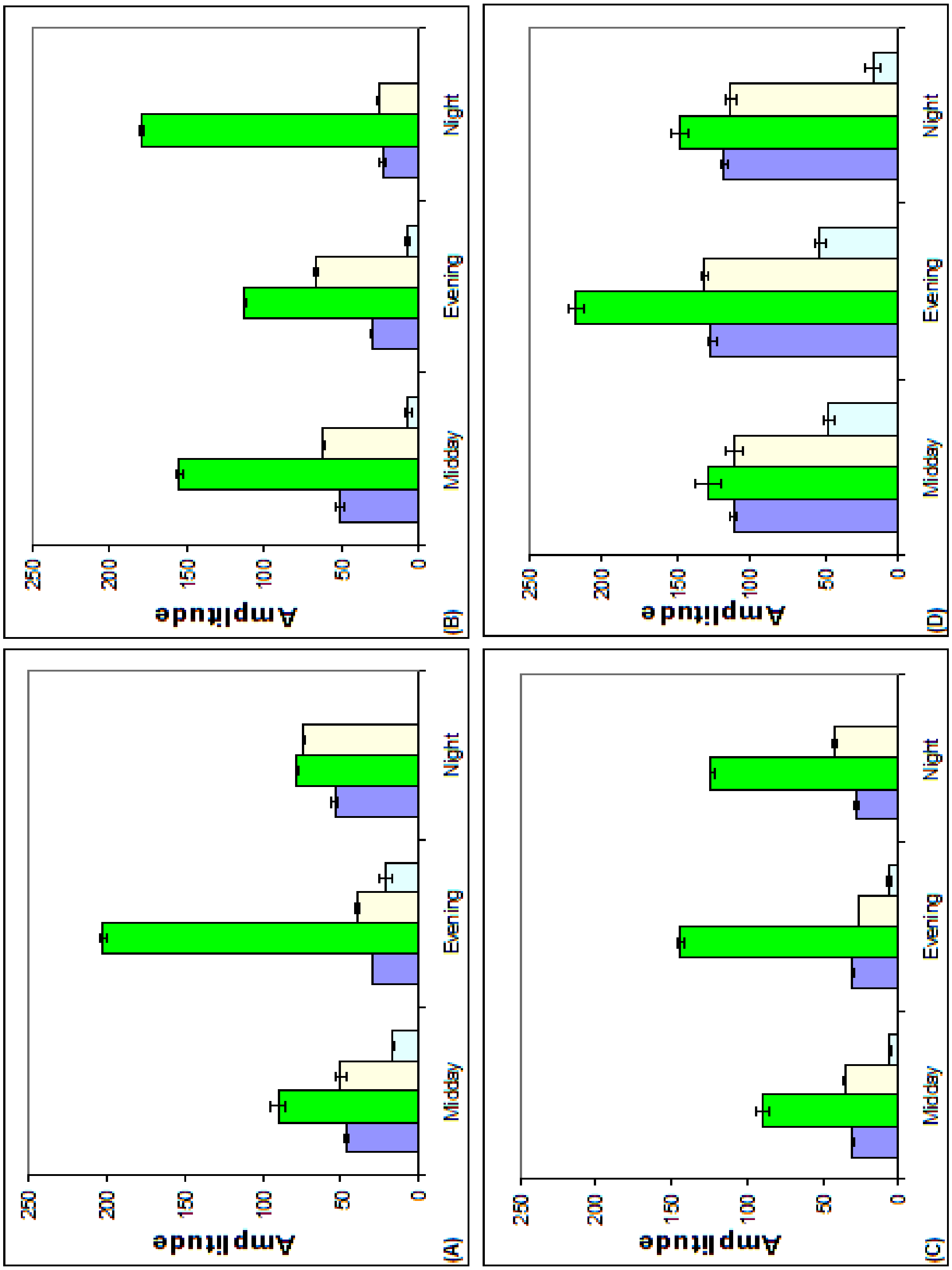


Figure 6.1: Reflectivity of agents when resting stationary on asphalt, grass, gravel, and water. (A): Average Surface Reflectivity vs. Time of Day, NXT 1: Out of Ball (B): Average Surface Reflectivity vs. Time of Day, NXT 2: Out of Ball (C): Average Surface Reflectivity vs. Time of Day, NXT 3: Out of Ball (D): Average Surface Reflectivity vs. Time of Day, NXT 1: In Ball.

accounted for the two lowest sampled means of this surface. The surface reflectivity of water at night is 0 for all NXTs, except for the when NXT-1 was placed in the ball. This is because there is some reflection on the inside of the dome from the EOPD light, which may be partially attributable to scratches on in from increasing wear.

Transient Surface Tests

I performed these tests to demonstrate that agents could recognize a new surface they were suddenly introduced to when initially on another. I held the agent approximately 5 feet above the ground at the beginning of each surface test and began recording data. At this height, the EOPD is unable to read any surface below it (but it still receives a very small amount of light), so the amplitude is small. After 7 seconds had elapsed, I lowered the agent onto the surface in 1 second, and let it record data for 7 further seconds, allowing the program to run for a total of 15 seconds. Using Agent-3 as an example, figure 6.2 shows midday data for the sampling of the surfaces.

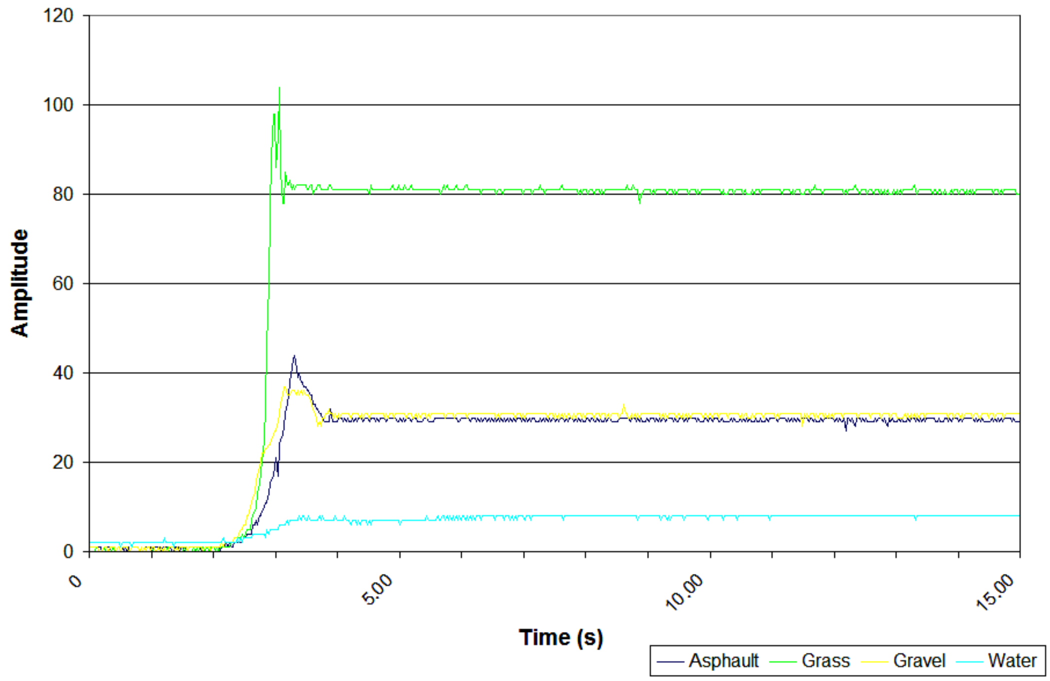


Figure 6.2: Transient Surface Reflectivity, Agent-3: Out of Ball – Midday.

All surfaces displayed characteristic step responses and overshoots (except the for the water test) when the agent was quickly lowered from the air to the surface tested. This was followed by steady-state responses of the transient system once the agent came to rest on the surface and continued to record data. Ideally, the averaged data points in the steady-state of a surface should be approximately equal to its mean value in the stationary surface tests.

However, what is of main concern is what the transient response tells us. Since a land-agent cannot enter water, the time which it takes a land-agent to recognize water, and then output a command to brake is the most important result of the transient tests. Using NXT-3 again as an example, figure 6.2 shows that the transient response for water was approximately 2 seconds. In comparison to the evening test, as shown in figure 6.3, Agent-3 had a transient response of approximately of 2 seconds. However, since at least 1 second of this time is due to the time it took to lower the agent onto the new terrain, the maximum settling time is therefore 1 second.

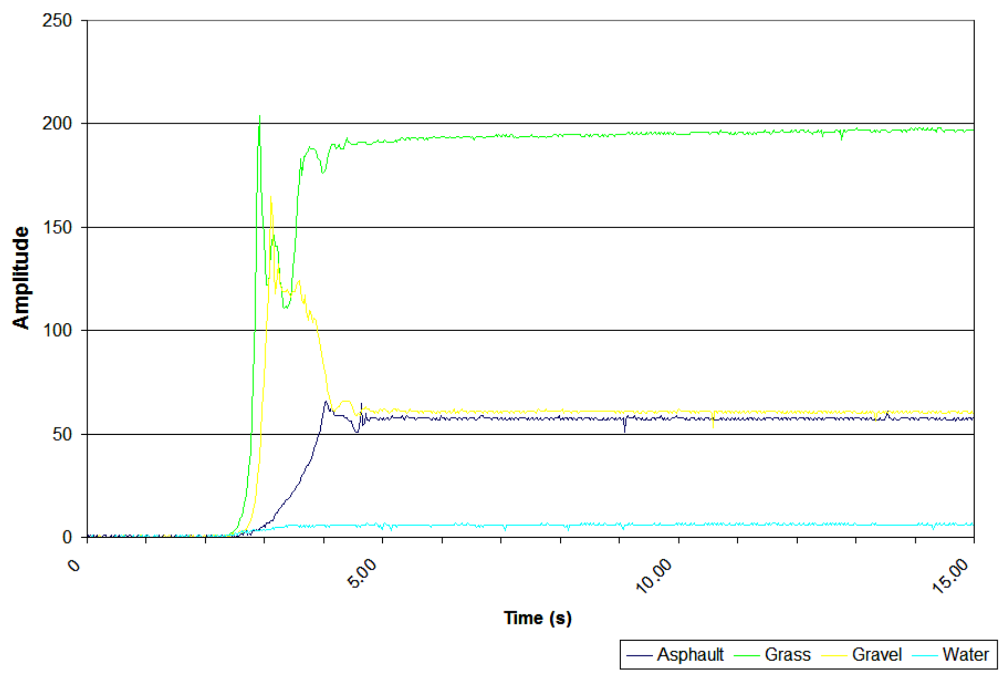
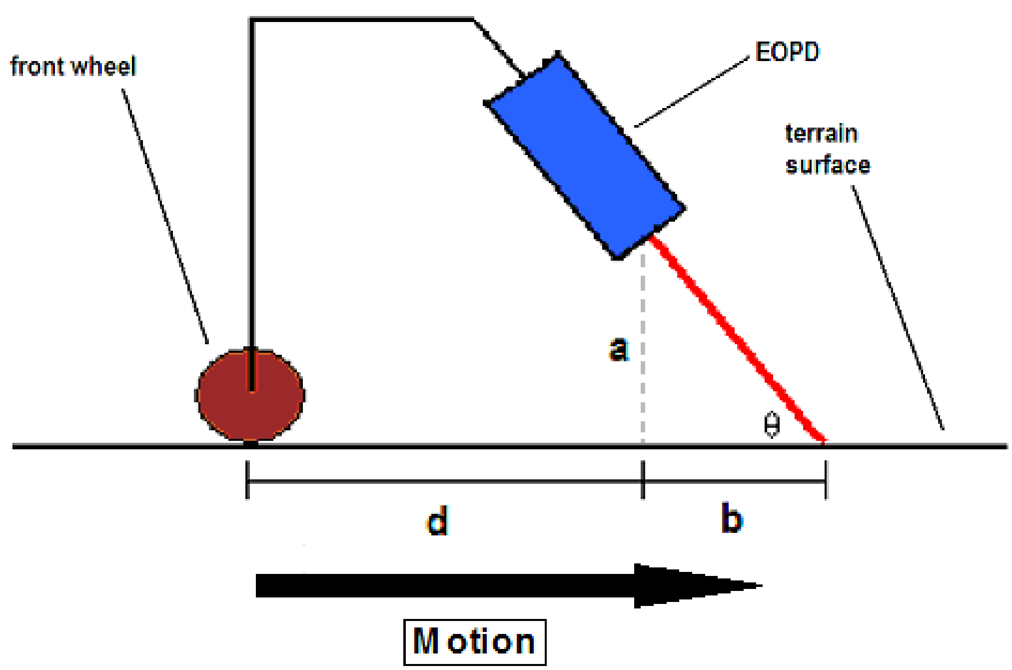


Figure 6.3: Transient Surface Reflectivity, Agent-3: Out of Ball - Evening.

From the transient responses, the settling time can be found. This is important in order to determine the theoretical maximum speed at which an agent can travel while being able to recognize the change between terrains, but most importantly for the land-agents, from land to water, *before* the agent actually enters water. This speed, which is determine by equation 6.1, and is used in the model of the system shown in figure 6.4, is:

$$A_{smax} = \frac{d_{wl}}{(t_{ts} + data_{es}) < t_s} \quad , \tag{6.1}$$

where d_{wl} represents the horizontal distance between the middle of the front wheel and the light beam, and A_{smax} represents the theoretical maximum speed of an agent in which it can recognize a change in a terrain and output a command to the motors to brake before it reaches the new terrain.



$\theta = 50^{\circ}$
$a = 1.5$ in
$d = 2.0$ in
$d_{wl} = d + b = d + \frac{a}{tan\theta}$

Figure 6.4: Geometric approximation of the EOPD sensor incident to a terrain surface.

Overall test results of the EOPD sensor show that under ideal conditions, such as flat terrain, consistent hue of a sampled surface, and pitch angle of the robot during testing can result in correct terrain recognition. However, several variables, as previously mentioned, can throw off the readings between measurements. For example, any differences between graphs in figure 6.1 are due to the fact that the color of the asphalt and the gravel sampled were sometimes similar. This points to limitations of using the EOPD sensor outside of a controlled environment.

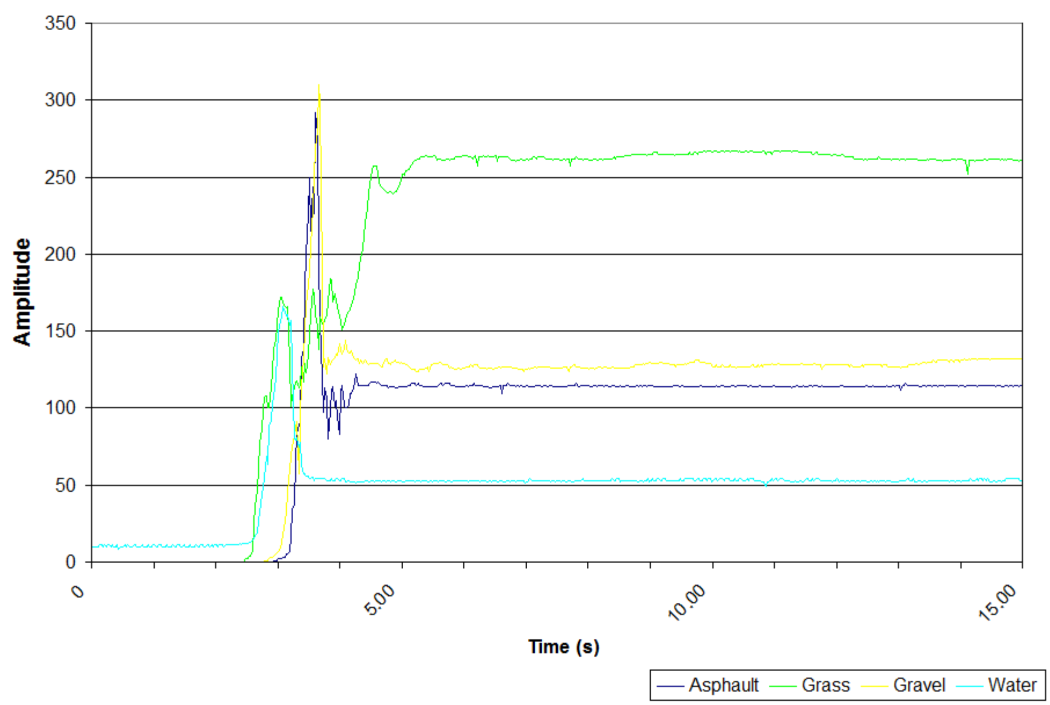


Figure 6.5: Transient Surface Reflectivity, Agent-1: In the Ball - Evening.

In figure 6.5, the water-agent's EOPD was sampled across the four terrains. Results from these tests show that the reflectivity off water, measured by the water-agent at sunset, was significantly greater than that measured by the land-agent in figure 6.3. However, this is of no consequence since the water-agent can survive in water, but nonetheless shows that land and water are also distinguishable within the ball. The larger reflectivity of water is probably attributable to reflection of light from the EOPD on the inside of the ball.

In order to determine if power output changed the results of reflectivity or settling time, NXT-2's battery was drained to one-half. The robot was then subjected to a midday stationary test on asphalt, in which the results were compared to that found from performing the same test at full power, which is shown in figure 6.6. The results of this test showed that when NXT-2 was at full battery, it had a 20% greater steady-state average than that of the half-battery power measurement. However, the maximum settling time was approximately 1 second, which was the same amount of time estimated for any other test.

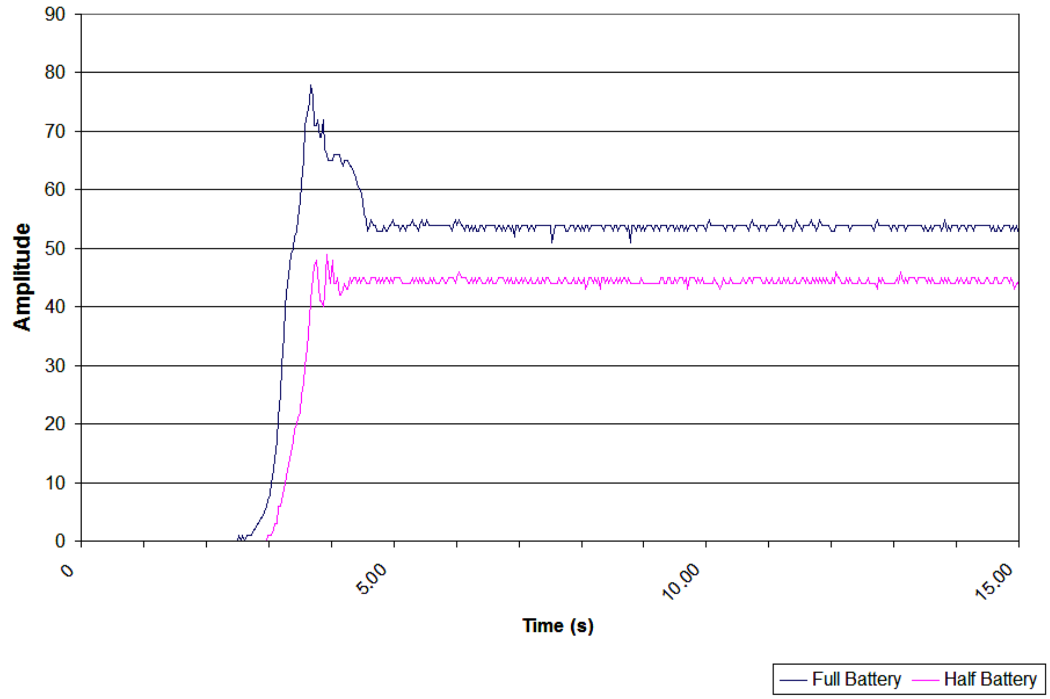


Figure 6.6: Transient Surface Reflectivity NXT-2: Out of Ball – Midday: Full vs. Half-power of the EOPD sensor on Asphalt.

The results of the reflectivity between the full and half battery power differed from expected ones as the EOPD sensor records analog readings, and returns a value over the voltage range of 0-3.3V (max) through a voltage regulator. Because the NXT's rechargeable battery's minimum full charge is 9V, its half power is still greater than the maximum voltage output of 3.3V by the EOPD sensor. Therefore, it would seem unlikely that sensor value readings or sampling rate would be directly affected by the difference between full and half power of the battery. These differences in reflectivity could be explained by the possibility of slightly darker hues of the asphalt which the light was reflected off of.

However, it is important to remember that while surface reflectivity may differ upon repeated samplings, amongst different NXTs, as well as at testing site locations, the most important requirement of the EOPD is to allow agents to distinguish between land and water. The results of all tests show that this was 100% doable, which therefore enables land-agents to recognize and avoid water, and water-agents to enter it when necessary.

6.2 IR Seeker

The IR Seeker was used to measure the signal strength by sensing IR light emitted from the beacon – the IR ball - which was pulsed at 1200 Hz. Larger readings received by the IR Seeker indicated that the agent was closer the beacon. In this section, I discuss how I calibrated the sensor, which turned out to be non-linear.

Data were recorded the same three times of day as that of the EOPD sensor: midday, evening, and at night. The reason I chosen these times of day was because at midday, when the sun is directly overhead, the earth is exposed to all wavelengths of light. At night, there is no sunlight. But at sunset (and sunrise), the sun's rays travel a larger distance through the atmosphere to reach the earth, and the shorter wavelengths dissipate before reaching the earth's surface through Rayleigh scattering as the shorter wavelengths, such as violets, blues, and greens are reflected off of small particles in the air. This allows longer wavelengths to reach the earth, scattering the blue wavelengths more than 9 times that off the reds. [21]. To check for interferences from other IR sources at the testing site, a control test of the IR Seeker with no IR ball present generated a zero signal strength reading at all times of day for all NXTs. This indicated that the IR Seeker effectively was able to filter out light which was not pulsed at 1200 Hz.

In order to determine the size of the testing environment the swarm can be in (in addition to limitations imposed by the range of wireless Bluetooth connection), I measured each agent's raw IR signal strength value as a function of distance from the beacon. This was done outdoors as shown in figure 6.7, at the specified times of day by placing the IR ball 10.5" above ground (shown on the left), and then extending a tape measure out from the base at the IR ball holder, moving the NXTs at 1 foot intervals along it up to the ball.

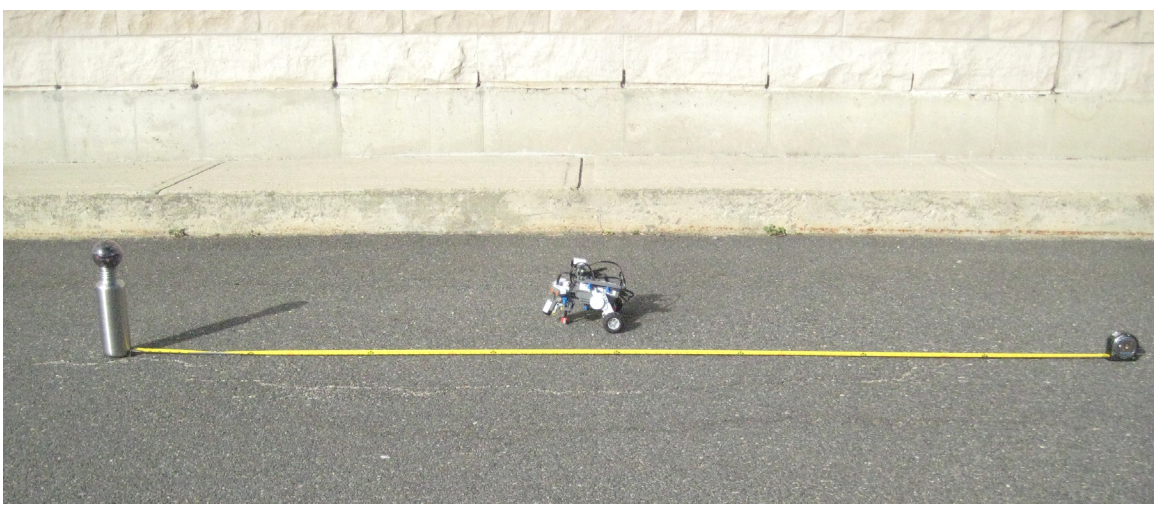


Figure 6.7: Testing of the IR Seeker at midday. An agent's IR signal strength was measured by placing the agent off-center from the IR ball, as seen in the picture.

Because IR can only be measured if a direct line of sight is visible between the beacon and an agent's IR Seeker, the IR ball was placed 3 inches higher relative to the IR Seeker's height of 7.5" so that it could not be blocked by another agent. Additionally, it is unlikely that the IR Seeker will be aimed directly at the IR ball during field testing, but rather it will receive IR light at an angle, between two or more of the IR channels. Since the IR ball emits light using LEDs in the shape of a light cone, and if this cone is weak, it may hit between the IR Seeker's channels, thus not engaging the receivers accurately. To account for this possibility, I measured the signal strength by placing agents off-center from the beacon for all tests, as shown in figure 6.7.

6.2.1 IR Seeker Testing Results:

The results of the IR testing in both figures 6.8 and 6.9 show a nonlinear relationship between sensor value and distance, which was expected as the light propagates outward as a cone. The results of these tests show that the IR Seeker is similar in distance measurements between all NXTs, in the evening test, as well as the night test.

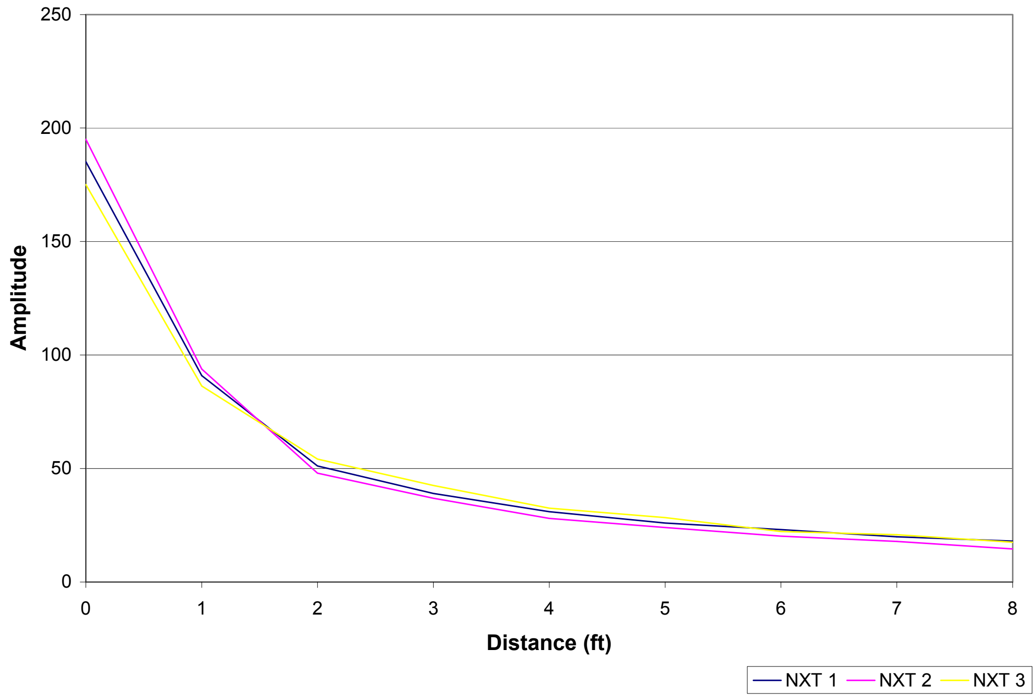


Figure 6.8: Infrared Light from IR Ball vs. Distance, Out of Ball – Evening.

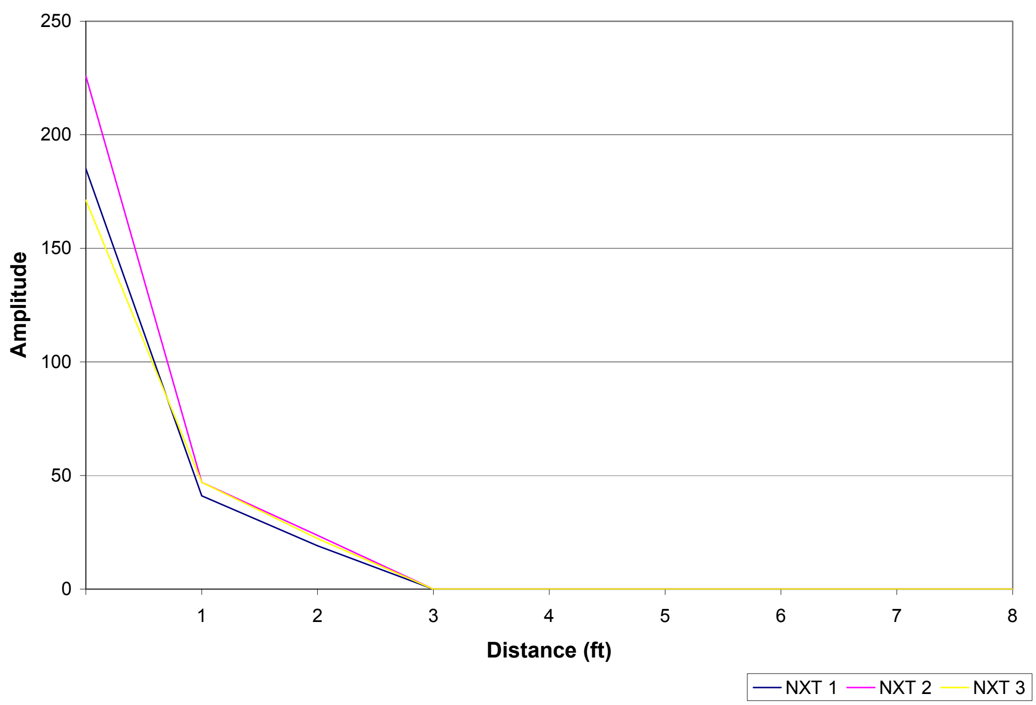


Figure 6.9: Infrared Light from IR Ball vs. Distance, Out of Ball – Night.

Infrared testing performed midday resulted in very little IR reception in the IR Seeker, as shown in figure 6.10. While the reason is unknown, I believe this occurred because of oversaturation of light entering the IR Seeker from the sun during the day creating very high noise. This phenomenon is reduced during the evening and is absent at night, which I believe explains why longer infrared ranges were detected by the IR Seeker, as shown in figures 6.8 and 6.9, respectively.

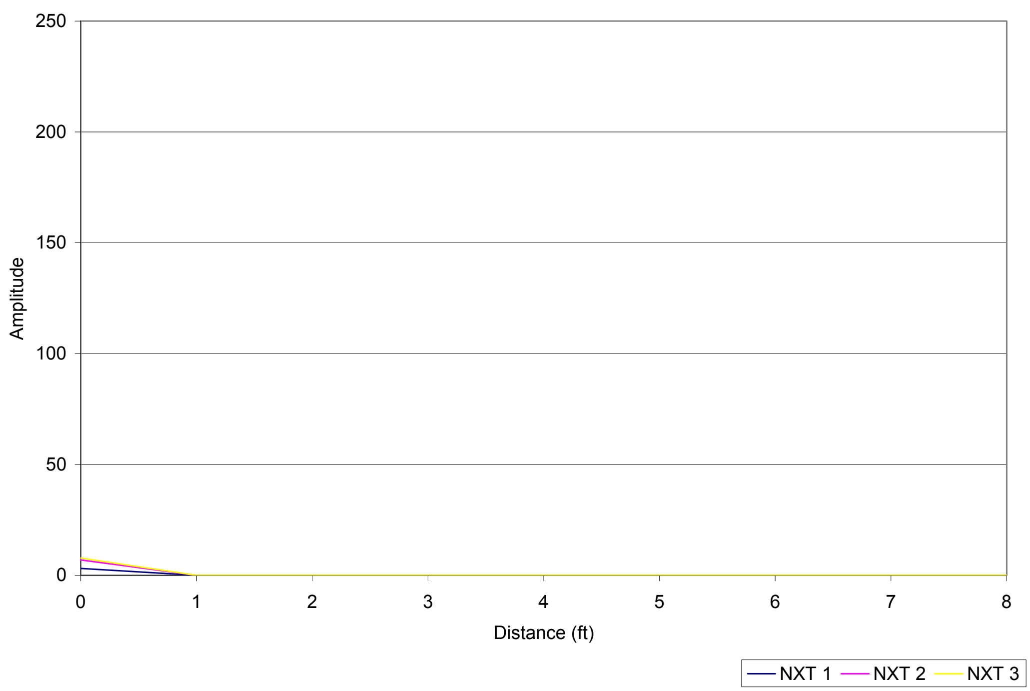


Figure 6.10: Infrared Light from IR Ball vs. Distance, Out of Ball – Midday.

To test this theory, I reran the midday test up to 1 foot away from the beacon. This resulted again with the IR Seeker receiving no signal strength. However, I ran a second trial afterwards and placed a black, rectangular box over the system. This resulted in the IR Seeker receiving signal strength at 0 and 1ft, as shown in figure 6.11, which was not previously seen in figure 6.10. These results provide strong evidence that oversaturation of the IR Seeker was occurring.

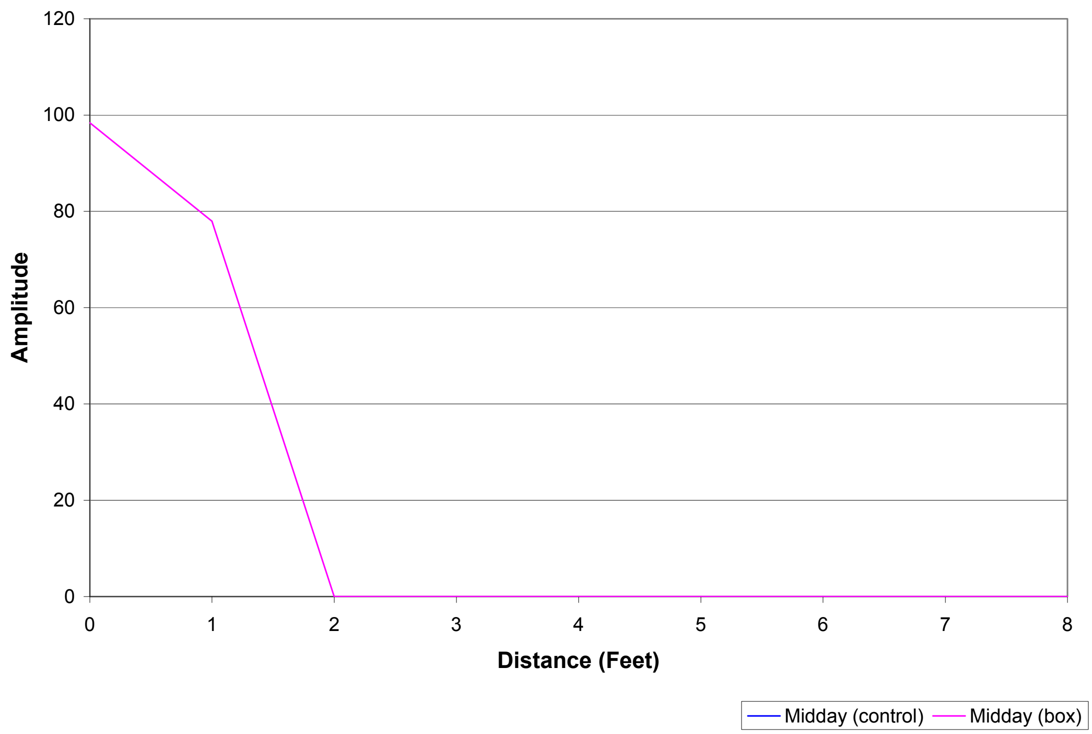


Figure 6.11: Infrared Light from IR Ball at 0 and 1ft, Out of Ball - Agent-2, testing system under box to block sunlight.

When the infrared values received by the IR Seeker from the land-agents are averaged per time of day, as shown in figure 6.12, are compared with the infrared light received by the water-agent in the ball, as shown in figure 6.13, we can see that the trend between the charts is similar.

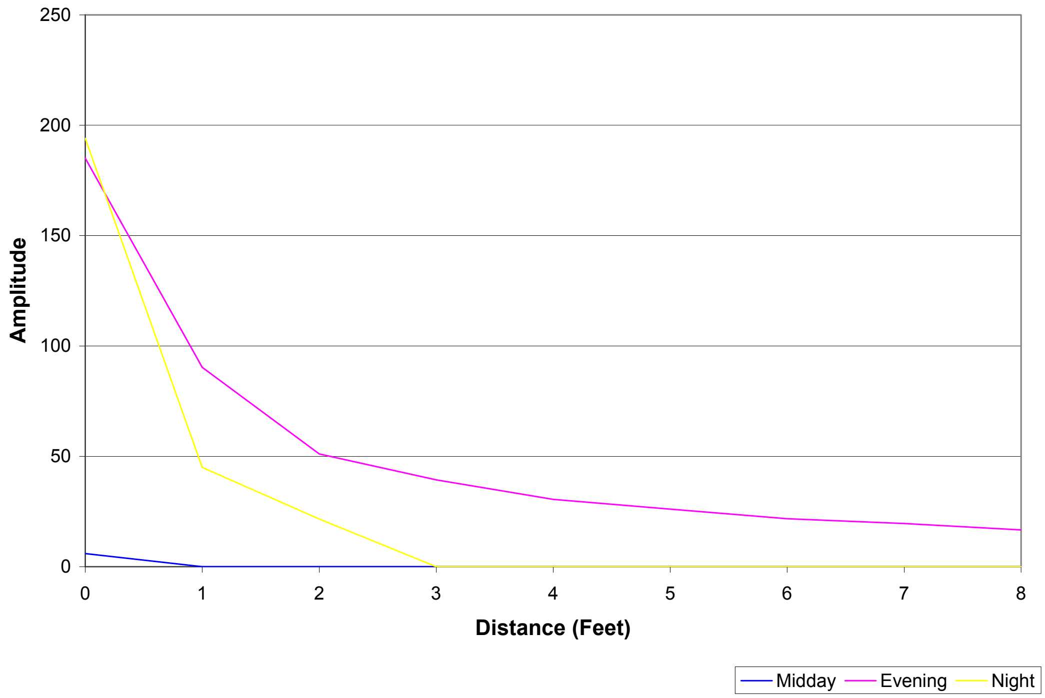


Figure 6.12: Infrared Light from IR Ball vs. Distance, Out of Ball - Averaged Values for all land-agents per Time of Day.

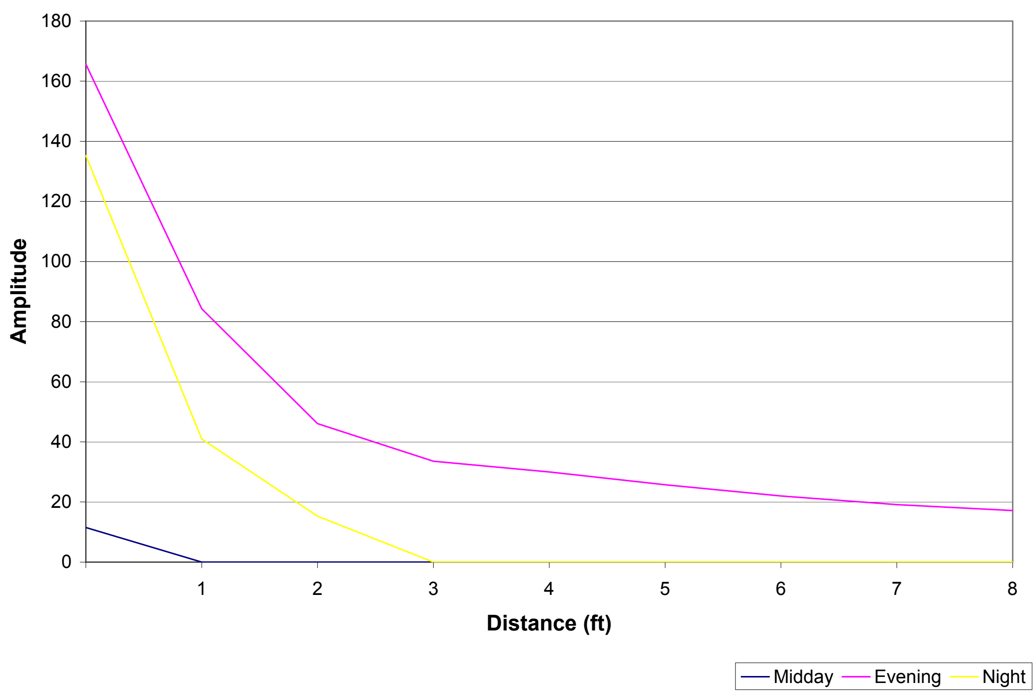


Figure 6.13: Infrared Light from IR Ball vs. Distance, Water-Agent, In the ball: All times of day.

The results figures 6.12 and 6.13 clearly show that the evening creates the optimal time for the largest infrared ranges by the IR Seeker on all agents. Since the general values of the IR signal strength versus distance for all NXTs display a similar trend over all tested times, it seems reasonable to conclude that there are no appreciable differences between the IR Seekers.

There are several variables which can affect measurements of the infrared light by the IR Seeker. For example, infrared light can be reflected off various surfaces, as well as absorbed by them. In order to test for any differences in the amount of infrared light received by the IR Seeker, a series of control tests were performed indoors in a windowless room to the outside, as shown in figure 6.14.

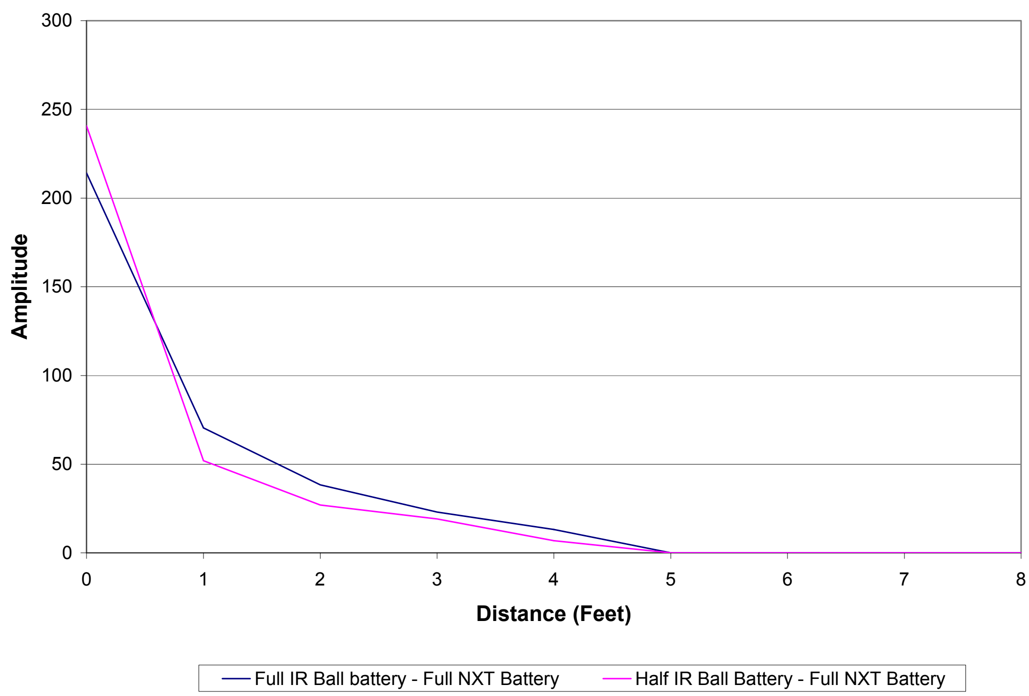


Figure 6.14: Infrared Light from IR Ball vs. Distance, Control Testing: Out of Ball, using NXT-2, Indoors, Full vs. Half IR Battery Power.

The results of figure 6.14 show that no observable change in the trend of the infrared measurements was recorded during the various control tests. These results are consistent with the manufacturer's owner's manual where it states that the IR ball is designed to maintain infrared light intensity even on low batteries at or below 10-20% of full battery life, which would make any decrease in signal strength by the IR Seeker unlikely. Furthermore, the IR Seeker is itself is unlikely to be a source of error at half versus full-battery power. Because it is a digital sensor, signal analysis is performed within the onboard sensor, and then values of signal strength are sent digitally to the NXT. Therefore, the use of a digital sensor makes it highly unlikely signal values will differ between full and half power reads from the NXT battery, which is consistent with the observed results of such control tests in figure 6.14 designed to check for this.

However, while the NXT battery is unlikely to cause a reading error in the IR Seeker directly, a low battery can affect power output to the wheels. This means that even though the sensors are expected to work at low battery power, the wheels will turn slower. This may result in an agent turning to the IR ball when requested to do so with difficulty, especially if the surface has more friction, such as on asphalt. However, because the IR Seeker has a 270^{o} range to receive infrared values, the agent would only need to rotate at most 90^{o} in order to receive a signal, as long as it is within range.

6.3 Concluding Remarks on Sensory Data

Recognition and distinction between land and water not only allows agents to prevent casualties in the scenario, but ultimately affects the swarm as it indicates a call to action by another agent if needed when traversing towards the beacon. A threshold was created for the IR Seeker in proximity to the beacon to prevent all agents to attempt to be Agent-1. This was necessary to do so since the agents would not be able to connect with each other if every agent in the swarm was trying to make a connection with another. All tests conducted with the land-agents where the goal was to distinguish between land and water demonstrated this with a 100% success rate. This allows us to conclude that the EOPD sensor is very accurate for this purpose on the agents.

The results of the IR tests throughout the day, as well as indoors, demonstrated different ranges which the IR light from the IR ball could be detected by the IR Seeker. However, because the trend in the range of all charts is the same for the agents, we can conclude that the IR signal strength measured on the IR Seekers is not significantly different amongst them.

Chapter 7: Experimental Testing

In this chapter I explain how I tested the swarm in two important scenarios – the Water Hazard and Lost Agent scenarios – each where I demonstrate the robustness of the swarm to complete the goal of reaching the beacon and overcome challenges the agents are subjected to during the tests. These tests were important because they placed the swarm in situations which required an emergent solution to be developed in order to reach the beacon – one in which may not be the same each time. Results of the test scenarios are discussed following each one, along with uncertainties and limitations of them, and are summarized in table 7.2.

7.1 Experimental Purpose

In this section I explain how the agents communicate amongst each other, decide which agent should head toward the beacon when located, and how to behave if a responding land-agent is unable to reach it due to a water hazard blocking its path towards it.

Table 7.1 shows the information which was available to the entire swarm - "Swarm Knowledge", versus that which was only known to individual agents – "Agent Knowledge". When the beacon is located by the first agent to see it, Agent-1, it broadcasts a message to all other agents in the swarm, one after another, to spin and look for the beacon. Each agent then responds to Agent-1 with a message containing all information listed under "Agent Knowledge". This allows for Agent-1 to sort returned information from each agent by signal value of all agents from closest to furthest from the beacon, as well as by signal value for water-only agents. (While there was only one water-agent used in this swarm, this latter sorting demonstrates how the system is scalable to the inclusion of both more total agents, as well as more water-agents.). The larger the signal value returned from an agent, the closer it is to the beacon.

Swarm Knowledge *(Known to every agent within swarm)*	Agent Knowledge *(Known only to individual agents)*
number of agents within swarm	name ID
	locally measured sensor values (EOPD, IR)
	terrain ID (land or water)

Table 7.1: Information known before program to either swarm or individual agents.

After 50 seconds has expired in equation 5.2 (for n = 3), which is the expected time for all agents to have responded back to Agent-1, Agent-1 will sort these signals, and send a message only to the agent who responded with the largest signal, Agent-n, containing the command to go to the beacon, as well as the name of the water-agent who is the closest to it as well.

If this responding agent encounters water separating itself from the beacon, it checks its own terrain ID to see if it is a water-agent. If it is, then it will enter the water, and continue unaffected towards the beacon. However, if it is a land-agent, it will send a command to the water-agent instructing it to go forward. Non-responding agents wait a time described by equation 5.3, which was 50 seconds for the tested scenarios, until they reset and look for the beacon again if they haven't heard from Agent-n that the beacon was reached. Similarly, Agent-1 waits a time described by equation 5.4 to reset, amounting to 54 seconds. If Agent-n does not reach the beacon in a time also described by equation 5.4, then it will assume the beacon has been moved further away from its initial location and will begin looking for it again. While the time to reset is highly dependent on input variables and location of the agents in the field, the differences in reset times of 4 seconds noted between equations 5.3 and 5.4 point to an uncertainty in the exactness of the equations. However, given that it takes 6 seconds for an agent to connect to another before sending a message, the variables used for these equations in these scenarios would not produce an issue because the messaging part of the algorithm is independent of the actions and behaviors, and therefore, the delayed agent would be able to receive a message before it reset and thus is without any consequences.

7.2 Test Scenarios

The purpose of this section is to outline the Water Hazard and Lost Agent scenarios in which I've tested the swarm. These scenarios were chosen because each demonstrates how the swarm responds to challenges it encounters, which were previously unknown and unexpected at the beginning of either test. Testing was originally planned outside at sunset to provide the longest range of infrared light from the IR ball to be used, but this short time of day prevented a consistent IR range as the evening advanced to night. Additionally, securing a location for reproducing tests could not be guaranteed, and therefore, indoor testing was performed in the laboratory to circumvent these issues.

Located inside a windowless room, the testing area measured 9ft × 14ft, with a 3.5ft × 3.3ft simulated water-trap (black paper), as shown in figure 7.1.

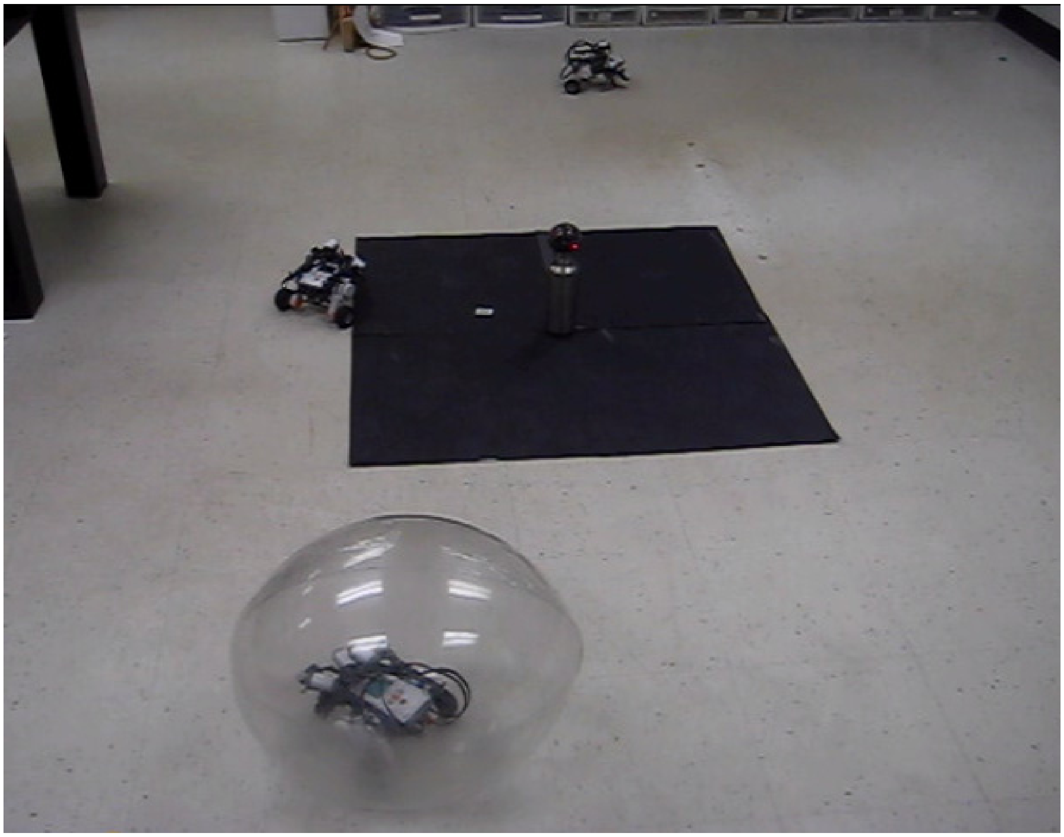


Figure 7.1: Scenario testing indoor. The white floor is simulated land, while the black paper is simulated water.

As was noted in chapter 6.1.1, the distinction between land and water is the most important test because it separates the terrains which a land agent can or cannot enter, respectively. To create this simulated terrain, I placed black paper, which represented water, atop of a white, marble floor, which represented land. Transient tests conducted by the land-agent, NXT-2, for both the floor and black paper was plotted in figure 7.2. The results show that these surfaces are distinguishable for a land-agent as a significantly larger reflectivity occurred off the floor than off the black paper.

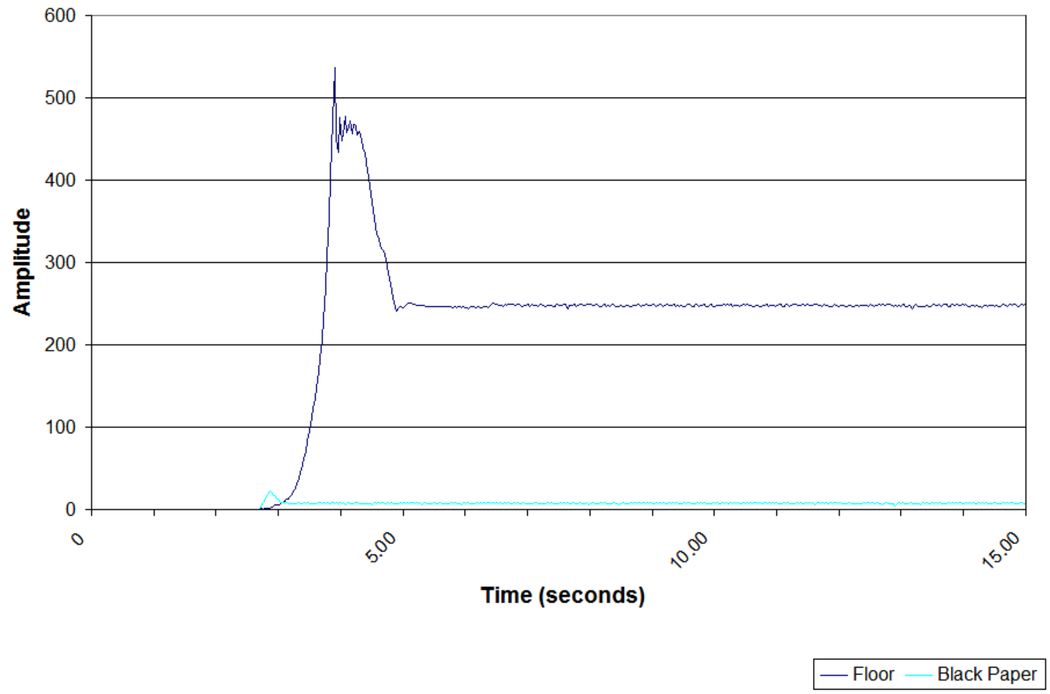


Figure 7.2: Simulated Terrains - Floor represents land and black paper represents water. Agent-2: Out of Ball.

A similar test was run for the water-agent, as shown in figure 7.3.

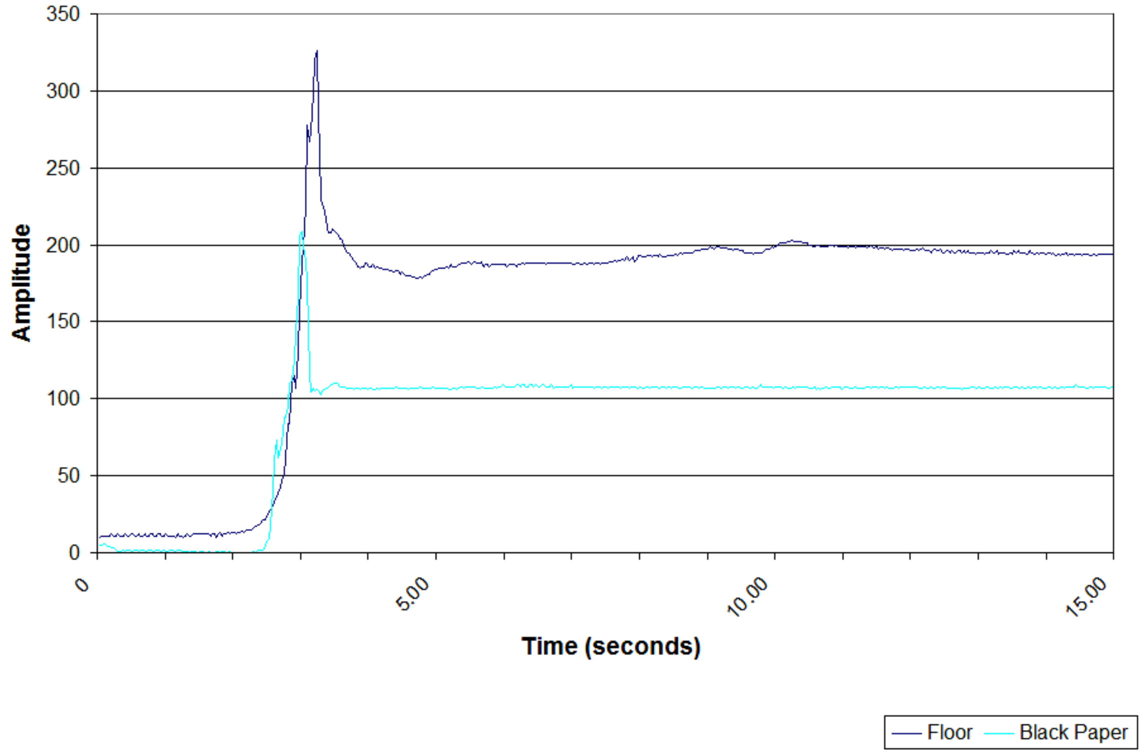


Figure 7.3: Simulated Terrains - Floor represents land and black paper represents water. Water-agent, in ball.

While the steady-state in figure 7.3 of the water-agent on detection of the black paper is above that of the land-agent in figure 7.2, it clearly is still approximately 82% below that of land. Since it is only necessary to distinguish the land and water surfaces from each other for the land-agent, this result is non-consequential.

In order to find the theoretical maximum speed a land-agent can travel, as determined by equation 6.1, the time needed to reach steady-state on the black paper surface shown in figure 7.2 must be found. Difficulties in determining a true settling time from this arose because it is unknown exactly at what time the agent was placed on the ground, at what speed this consistently occurred at, and the exact rate at which the agent was lowered at. Therefore, the maximum settling time is estimated at approximately 1 second.

However, due to the aforementioned difficulties in determining a maximum speed, linear speed testing had to be performed to determine the speed at which a land-agent would be able to repeatedly recognize water. This experimental speed was found to be 0.28ft/s, which is good enough because it allowed for 100% recognition of the black paper during the tested scenarios on approach, and never crossed onto it in 50 out of 50 trials. While faster speeds may have been possible, they would offer no advantage since equations 5.3 and 5.4 incorporate the linear speed as a variable in determining timing for the system.

As shown in figure 7.1, the beacon was placed on the black paper. In order to test the robustness of the swarm, three characteristics were measured in each test:

1. Did all agents within the swarm repeatedly connect?
2. Was the swarm able to reach the beacon?
3. What was the completion time?

The test completion time was measured starting when Agent-1 found the signal, and ended when Agent-n successfully reached the beacon, as measured by an external clock.

The connection of the swarm was measured by Agent-1 successfully receiving all each agent's signal strengths after the broadcast to look for the beacon, and the agent closest to the beacon successfully traversing towards it. This was done to test the integrity of the dynamic ad-hoc mesh messaging protocol I created. It should be noted that the ad-hoc network is dependent on if the agents are within range of each other to allow for connection to one another to be made.

7.2.1 Water Hazard Scenario

The swarm was tested in a surveillance operation approach where all three agents were driving autonomously on land looking for the presence of the beacon. During the test, the nearest agent to the beacon was a land-agent, which became Agent-1 upon signal recognition. However, upon approach to the beacon, it encountered water in its path, as shown in figure 7.1, and thus needed to message the water-agent to respond to the beacon instead.

The time it took for each test to be completed was then placed in one of the time interval slots in the histogram of figure 7.4 in which it fell into. For example, figure 7.4 shows that there were two tests which each took between 1:01 – 1:10 minutes to complete, while there was only one test which took 3:01 – 3:10 minutes.

The results of this scenario testing found that the swarm was able to connect to each other with a 72% success rate, but was able to reach the beacon 88% of times overall in the water-hazard.

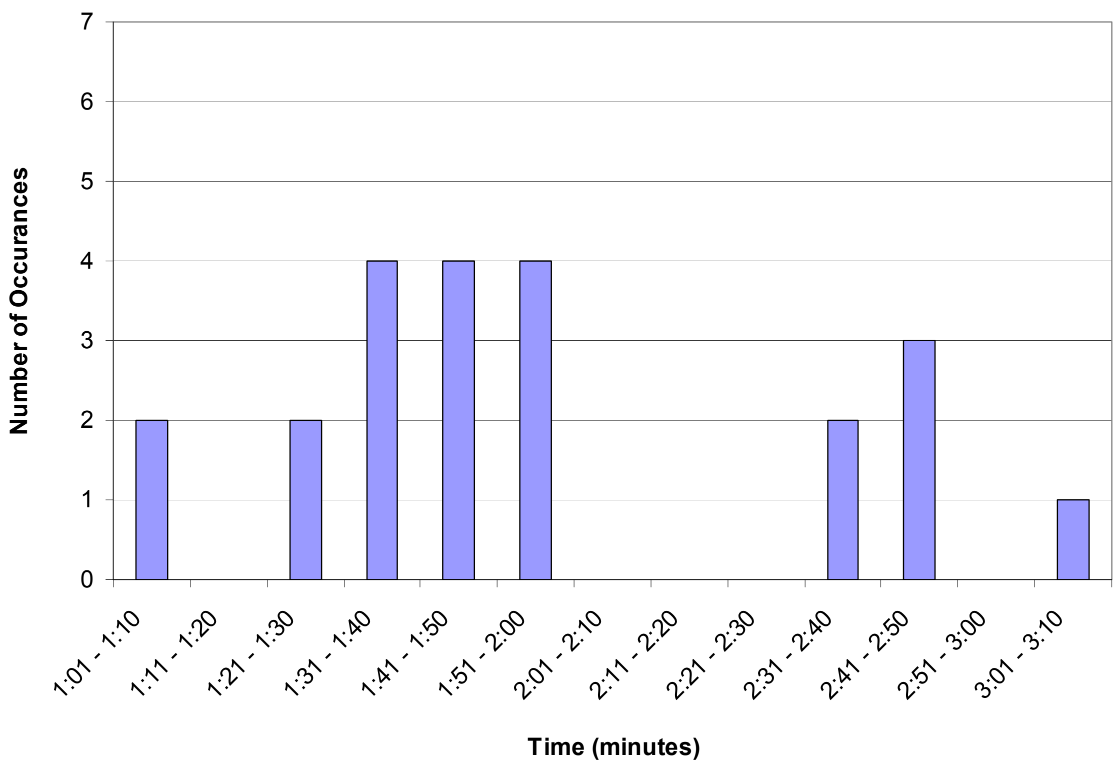


Figure 7.4: Time intervals for which the swarm was able to reach the beacon in the water-hazard scenario. The swarm reached the beacon in 22 of 25 tests (88%). The dynamic ad-hoc mesh network success rate of connection amongst the swarm was in 18 of 25 tests (72%).

A 30 second gap can be seen between 2 minutes and 2:30. This represents the time in which the agent's algorithms had reset, and begun looking for the signal again. In 3 out of the 6 trials which resulted in a reset, 50% of the time the swarm was able to reach the beacon. In the three tests which the swarm did not reach the beacon, Agent-n, the water-agent, was unable to locate the beacon after too much time had expired during each search, with a 5 minute cut-off for attempts to reach per trial. These failures occurred during tests when the ad-hoc broadcast from Agent-1 was successful.

7.2.2 Lost Agent Scenario

The goal of this scenario is to examine how the swarm completes the task of reaching the beacon if the responding agent, Agent-n, is physically removed from the swarm before it reaches it. The other agents wait for a broadcasted message from the responding agent that the signal has been reached. Their waiting time is determined by equation 5.3 which instructs the agents to reset their algorithm to look for the beacon again if the responding agent does not broadcast a message that the beacon has been reached. A reset is performed individually by each agent, if the time waiting for the responding Agent n's notification that the beacon was reached exceeds the time allotted by the reset equation; Agent-n will also reset if it does not reach the beacon in the expected time, as given by equation 5.4. Note that it was not possible to have Agent-n "check in" with the other agents while traversing towards the beacon due to memory size limitations on the NXT to program it to, as it was at capacity.

The results of this test, displayed in figure 7.5, showed that the swarm was able to connect to each other with an 84% success rate, and was able to reach the beacon 80% of times overall in the lost-agent scenario.

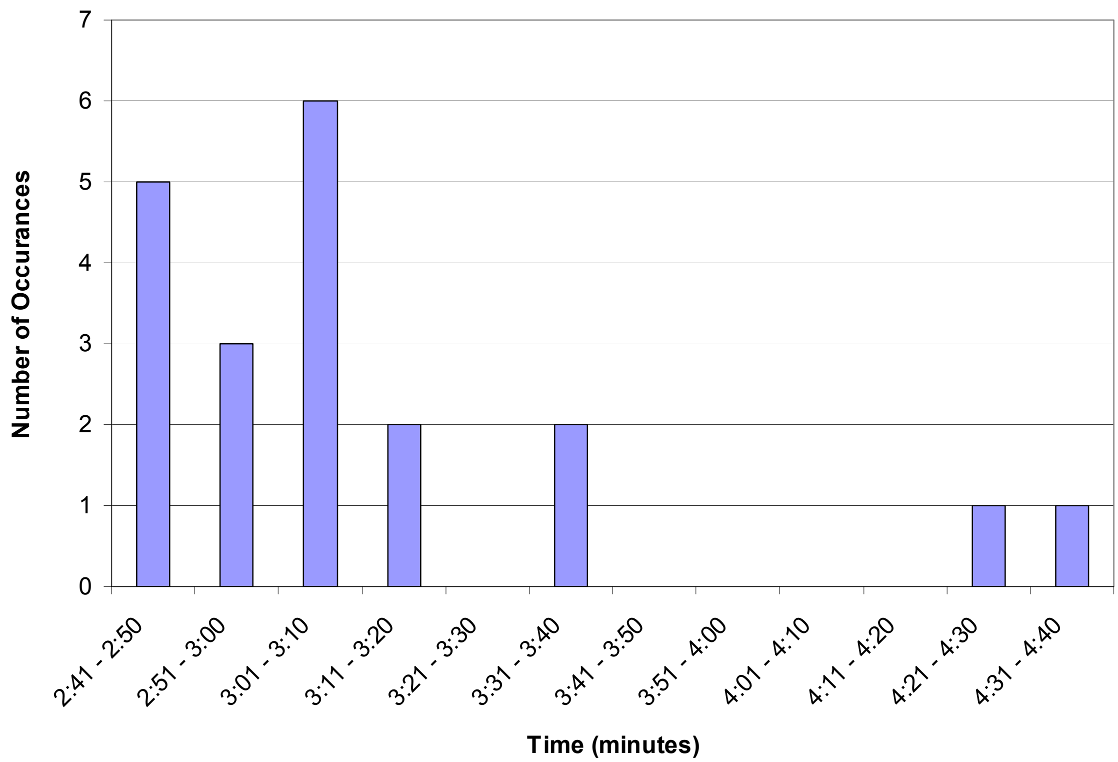


Figure 7.5: Time intervals for which the swarm was able to reach the beacon in the lost-agent scenario. The swarm reached the beacon in 20 of 25 tests (80%). The dynamic ad-hoc mesh network success rate of connection amongst the swarm was 21 of 25 tests (84%).

A 50 second gap can be seen between 3:40 and 4:20 minutes. This represents the time in which the two remaining agent's algorithms had reset, and begun looking for the signal again since the new Agent-n (after the original one was removed from the system) did not reach the beacon in time allotted by equation 5.4. In 2 out of the 5 trials which resulted in a reset, the swarm had a success rate of 40% in reaching the beacon.

In the 5 tests which the swarm did not reach the beacon, Agent-n, the water-agent, was unable to locate the beacon upon multiple resets of its algorithm after too much time had expired per reset, with a 5 minute cut-off of total time allotted for attempts to reach the beacon per trial. The test failures occurred during tests when the broadcast from Agent-1 was successful in 3 out of 5 of the trials. The 5 minute cut-off is independent of the number of resets which occurs, as it is measured by an external clock to the system. Tests showed that if the swarm hadn't reached the beacon after 5 minutes, it was unlikely that it was going to, due to some failure or blockage of the water-agent. Additionally, battery life would have been drained as testing progressed, and therefore, test cutoff after 5 minutes seemed justified because of these reasons.

The failures in this scenario occurred because only Agent-1, a land-agent, had initially discovered the beacon as it was the only agent which registered the signal strength larger than the threshold. When Agent-1 was removed from the swarm after broadcasting, the other agents never crossed the threshold during the test, and therefore, the swarm was not able to complete it. Because the range of the IR ball indoors measured only 4ft, the agent chosen to be Agent-1 was required at the start of the test to be very close to the signal to guarantee crossing the threshold. The threshold ensured that only the closest agent would receive the signal, which was necessary so that multiple agents did not receive it and try to connect with each other – which would have defeated the purpose of the scenario for testing the algorithm. This meant that when the water-agent reset its algorithm, even though it was receiving signal strength, it was less than the threshold, and therefore, it was unable to traverse towards the beacon within the maximum trial time allotted. The reason the water-agent in the water-hazard scenario was able to reach the beacon when it had to reset is because the threshold only applied to Agent-1 locating the beacon. When Agent-1 sent a message to the water-agent to head towards the beacon, it was able to do so because the agent only required the presence of signal strength to move forward.

	Beacon Reach (success rate)	**Ad-hoc mesh (success rate)**	**Average time to complete (including resets)**
Water-scenario	88%	72%	1:56
Lost-agent scenario	80%	84%	3:11
Overall	84%	78%	N/A

Table 7.2: Summarized results from the Water and Lost-Agent Scenario tests.

7.2.3 Lost Water-Agent

A special case in the lost agent scenario occurs when the water-agent is removed from the swarm. As would be the case when either Agent-1 or Agent-n is removed from the swarm, this would result in a time expiration of equation 5.3, indicating one of two events has occurred:

Possibility 1: The responding agent failed.
Possibility 2: The beacon has moved further away from the responding agent than equation 5.3 anticipates the time required would take to reach it, and by the system restarting, it will determine if this agent is still the closest to the beacon, or if another agent is.

Because this thesis is on a swarm of agents capable navigating over multiple terrains, which includes water, the swarm would not be able reach a beacon separated by water from the field a water-agent failed. Therefore, the system is programmed to shut down if a water-agent is not included in the swarm if a land-agent encounters water on approach to the beacon. This was designed because it would be useless and a waste of resources (such time and energy) for a swarm to continue to operate in a field if it encounters water with no water-agents available to be called to cross it. This is an advantage of using heterogeneous robotics for swarm intelligence, where such terrains would allow for any ground mission, while the lack of a terrain-specific agent in a swarm can provide feedback to the swarm about a lack of ability to continue on with the mission, thus saving resources.

Note that the time for reset occurs following an agent's returned signal to Agent-1, and therefore varies by the position in which the signal was received so that the agents will reset ideally at the same time. If this does not occur, there are a couple of issues which may occur.

If an agent resets and locates the beacon before the other agent does and is unsuccessful in an attempt to connect because the other agent has erroneously not broken its connection from a previous link, it would not be able to tell this agent to look for the signal. The scenario may unfold in that both agents wind up heading for the beacon separately. This presents a scenario where agents independently found the beacon (which turns off their ability to receive a broadcast from an agent who also, by chance, found the beacon), and determined that no other agents were available after it tried broadcasting to the swarm and got no responses, and thus heads to the signal.

The outcome of Possibility 2 leads to three scenario outcomes:

1. Multiple agents are traversing towards the beacon. The agents could result in a bottle neck as they approach the beacon, unless one of the agents

blocks the line of sight from the others of the IR light as it is very close, and eventually reaches the beacon itself.

2. Multiple agents reach the signal at various times. If an agent does reach it, it will send a broadcast message out to the swarm that the beacon has been reached, which instructs the swarm to turn off. However, this only works if the swarm is in the waiting mode "listening" for response that the responding agent has reached it. If other agents are instead heading to the beacon themselves, then when these agents receive the signal, it will turn off their ability to receive messages anymore, and they will keep trying to reach the signal. This has to do with the structure of the algorithm, where messages to be received are in a parallel process to the sensors and behaviors part of the code.

3. A variation of #2 becomes true where multiple agents are traversing towards the signal, they will eventually reach the water hazard. If they are not water-capable, they need to call a water-capable agent to transverse it. However, if this agent contains no information of a water-capable agent, then they will broadcast to the swarm to turn off, and then shut themselves off as well.

The result here is that all agents have had their ability to read messages turned off, and are acting as if they are the solo agent left in the swarm. However, once the land-only agents reach water, there will eventually come a time where they will turn off because they cannot reach the water agent and cannot traverse the water, as described in #3. Therefore, the water agent theoretically will still move in the scenario, and reach the beacon. The only part where it would fail is if the agents are blocking the water-agent from entering the water, and thus, the system would eventually fail, as the water agent's battery drains.

Note that this communication pitfall occurs because the agents expect a certain amount of time from the responding agent to declare that it reached the beacon. This amount of time is variable, as outlined in the equations in chapter 5. However, these issues could be overcome if the agents were able to exchange information quicker and if the NXT had more memory. This would allow the agents to check in on the responding agent by pinging it, and the responding agent could update the others on its position at all times. Additionally, if the agents didn't hear from the responding agent, indicating that it was lost, they would broadcast over a spread spectrum quickly. The issue of the ad-hoc network where it is only able to communicate with one agent at a time could be fixed by using a flooding algorithm, where other agents act as transmitters to specific agents in the swarm (but this would only be useful if $n > 3$), as well as equipping agents with a call-waiting radio option, or multiple receiving channels. However, due to the aforementioned issues with limited memory, as well as hardware capabilities of the NXT, a different platform would have to be used in order to implement these solutions.

7.3 Data Logging

A data logger integrated into the algorithm as a means for agent communication and events to be recorded. Every time a message was sent or received by an agent, this message was saved to the file with a time stamp when the event took place.

Having a data logger in a swarm is advantageous for several reasons. On the programming side, it allows for debugging by seeing what messages were sent and received by the agents. Because messages are bundled with several pieces of information in a packet to each other, reaching the data log allows the transcript of an agent, and the swarm to be analyzed to recount what occurred in the mission. In the test scenarios, the logger was used to retrieve information about where in the field agents were when Agent-1 located the signal. This helps construct a picture afterwards of how far agents were away from the signal when they received the command from Agent-1 to look for it by comparing their signal strengths with table 7.3, and at what time an event occurred, relative to their individual internal clock on start-up.

Table 7.3 shows an example of the benefit of using a data logger, which was obtained from NXT-2, one of the land-agents in the water scenario testing after a trial.

Line	Messages (encoded)	Messages (decoded)
1	046-2$1-IR/n9.06	NXT-2 located the IR beacon with signal strength value of 46 (NXT 2 is now Agent-1)
		NXT-2 sent NXT 1 signal strength value of 46 at 9s
2	046-2$2-IR/n19.1	NXT-2 sent NXT 3 signal strength value of 46 after 19s
3	042-1%water-mySignal/n42.6	NXT-1 returned signal strength value of 42, and declared it was a water-agent at 43s
4	026-3%land-mySignal/n51.6	NXT-3 returned signal strength value of 26, and declared it was a land-agent at 52s
		NXT-2 had the largest signal strength value in the swarm, and thus was closest so it sent itself towards the beacon
5	1-NXT-2%water/n65.5	NXT-2 encountered water, read it's terrain file that it was a land-agent, and stopped
		NXT-2 messaged the water-agent, NXT-1, commanding it to go to beacon at 66s
6	NXT-1-signalReached/n110.8	NXT-1 broadcasted the message to the swarm that it had reached the beacon at 111s
7	/n110.8	NXT-2 turned off registering that the mission was complete at 111s

Table 7.3: An example of a data log, from Agent-1 during a water-scenario trial.

Having this data log installed in the algorithm allows for a human-operator to contact the agent during the mission (as long as it is not in connection with another agent), and access its file. This will allow the operator to view what communicative events the agent has made to others and vice versa. This can also be accessed after a mission is complete to get a historical recall of the mission, as shown in the “Messages (decoded)” column.

7.4 Concluding Remarks on Swarm Testing

Results of the combined 50 trials from the scenarios demonstrated that the dynamic ad-hoc mesh worked in 39 of 50 tests for a 78% success rate. Individual tests results in figure 7.4 produced an average time of 1:38 minutes to reach the beach on the first attempt. However, the physical testing presented problems as testing progressed. Initially, the water-agent was able to drive in the ball without problems. Because the body of the agent had independent rear axles, this allowed the water-agent to drive on curved space without incident. However, securing the domes on top of each other was cumbersome. When the water-agent was placed in the lower dome, it stretched it approximately 0.5 inches larger than the top dome making it difficult to tape them together, as shown in figure 7.6, and required two people to do so in order to get a sufficient fit to allow the agent to drive straight. However, upon successive testing, wear made the connection more difficult, inhibiting the water-agent to occasionally roll straight as the front wheel becoming slightly stuck in the gap where the difference in the diameters the domes was increasingly evident.

Figure 7.6: Area where rim of top and bottom hemispheres are misaligned, creating a lip in the seal between them.

When this occurred, it was necessary to lightly knock the wheel out of the groove so that it could roll straight again. This was important to do because this physical problem with the domes would have masked and impaired the results of testing the algorithm of the swarm to reach the beacon - the goal of this thesis. When this problem was corrected, the swarm succeeded in 42 of 50 attempts to reach the beacon for an overall 84% success rate between the two scenarios. Additionally, the swarm was able to reach the beacon on the first attempt upon beacon recognition in 34 out of the 42 successful trials, for an overall 81% success rate without the algorithm resetting.

The timing between the reset of the non-responding agents and that of Agent-1 was 4 seconds in the tests. This points to an uncertainty in exact timing between equations 5.3 and 5.4. However, because the main point is to reach the beacon, even if the agents do not all reset at the same time, the water-agent should still be capable of reaching the beacon alone, unless it experiences difficulties of avoiding obstacles with the light sensor, as was previously mentioned was occasionally an issue.

A solution to fix the issues with agents not being able to receive a message to turn off if they are in a different part of their program could not be implemented because memory on the NXT was at capacity. However, because the goal of the swarm is to have Agent-n reach the beacon, this issue is then secondary to the main objective. The cause of failure, as previous mentioned, was that the water-agent was not able to reach the beacon in under the 5 minute cut-off time. One reason was because it got stuck behind Agent-1, which was stationed at the water-trap and could not navigate around it. The other reason was that the water-agent collided with a wall, and the light sensor did not instruct the agent to avoid it. I believe this occurred because as testing progressed, several scratches developed on the ball due to wear, which turned many parts of the originally tested transparent surface of the ball to an opaque one. This created a reflection inside the ball, which made the water-agent receive higher values due to this reflection.

An additional issue with the light sensor occurred as the land-agent was positioned at the edge of the black sheet, and needed to make a call to the water-agent instead to continue to continue the mission, the water-agent often encountered a blockage of the land-agent separating it from the beacon. This resulted in a collision of the water-agent and the now stationary land-agent. However, a positive result emerged from this in that the ball was able to absorb part of the impact from the collision by folding inward as the water-agent continued driving forward, as shown in figure 7.7.

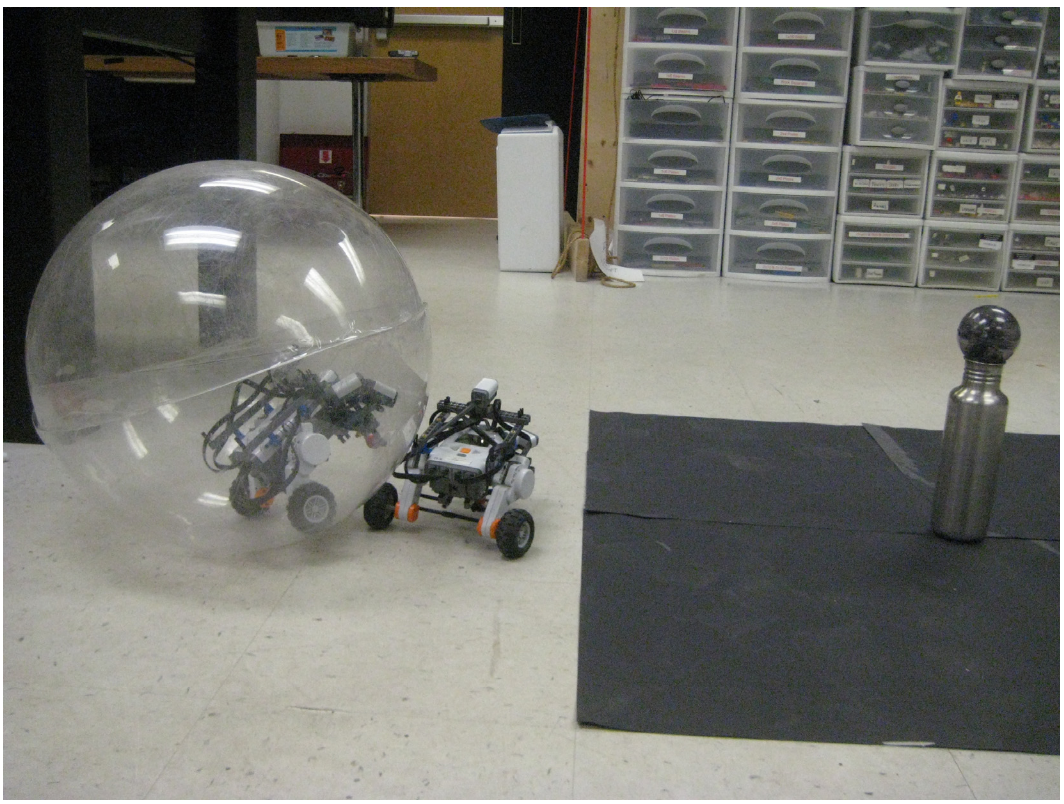


Figure 7.7: Water-agent driving toward beacon, colliding with a land-agent in its path.

As the water-agent drove forward, further absorbing impact, it began to clear the land-agent, and part of the energy from the collision was elastically returned to the ball. Furthermore, both agents were undamaged during these collisions. Since the water-agent's wheels are raised on the curvature of the ball (as it drives on curved space), this allows the plastic beneath the agent's undercarriage to contact the ground and elastically deform when it is placed on or rolls over an elevated surface or object. Elasticity of the plastic allows the ball to absorb a small impact, while keeping the agent at a nearly unchanged height relative to the base of the elevated surface and allowing for elastic recovery.

These results have demonstrated that both the dynamic ad-hoc mesh and the swarm algorithm are robust to allow the swarm to reach the beacon. While the misalignment of the domes prevented the water-agent to always execute desirable driving control, this did not prevent demonstration of the swarm's robustness, as well as the elasticity of the ball as it was used to the advantage of the swarm to help reach the beacon in lieu of underperforming light sensors for collision detection.

Chapter 8: Sensors, Controls, and Behaviors-Based Actions

8.1 Scalable Sensors

In this section I introduce a scalable approach for implementing an intelligent sensor-control design for any agent traversing any terrain that I designed for the swarm's algorithm. An agent's behaviors are controlled as function of data received by a sensor. The scalable sensor function has the ability to decide which sensors to use depending on the terrain which the agent is designated to traverse on, which is read from an external text file on the agent. If the file reads "land", the agent uses sonar. If the file reads "water", then the agent uses a light sensor. This allows data to be collected differently based on user-defined input, without having to change the body of an agent. In this regard, the only changes which are required is the file instructing which sensor the agent should read, as well as the physical sensor itself.

Additional collision-detection sensors to the ones used could be added, as well as their code to read them in the scalable sensor block to provide more information about the surrounding environment, which can supply an agent more data to process and make decisions from. However, as system resources become scarce, this can decrease performance of an agent, and ultimately, the swarm as a whole. Figure 8.4 discusses such practical limitations, and offers a realistic solution to solve these issues to increase efficiency while decreasing overall processing time of a robot, that ultimately leads to a more robust swarm.

When an agent is navigating on land, its drive system is controlling the power output to its wheels (which also controls steering and braking by changing the power allotted to each wheel), which are in direct contact to the ground. It uses sonar to detect an echo off of an obstruction (which includes another agent) in the way of its path, where the agent will alter its behavior in order to avoid the collision by simply backing up and making a slight turn to the left (which was arbitrarily chosen) before driving forward again. However, when the same agent is within a ball, the drive system remains unchanged. This is because the agent drives on curved-space, due to the ability of the axels to bend, which is also good for shock absorption. However, this would present problems if it tried to also use sonar for obstacle avoidance because the sonar will echo off the interior of the ball and not penetrate the plastic. Since the ball is made of clear plastic, generate light works well to penetrate this medium. Therefore, a beam of light was chosen for this mode of transportation, as discussed in Chapter 4. Because each agent in the swarm uses the same algorithm, there needs to be a way for each agent to know whether it is meant for ground-use only, or if it is meant to drive within the ball (which makes it a water-capable agent as well as ground), so that the agent would know to use light or sonar for obstacle detection and avoidance.

To make this customization, a text file is loaded onto the agent prior to deployment in the swarm. The file contains the string of either “land” or “water”, which is the terrain input.

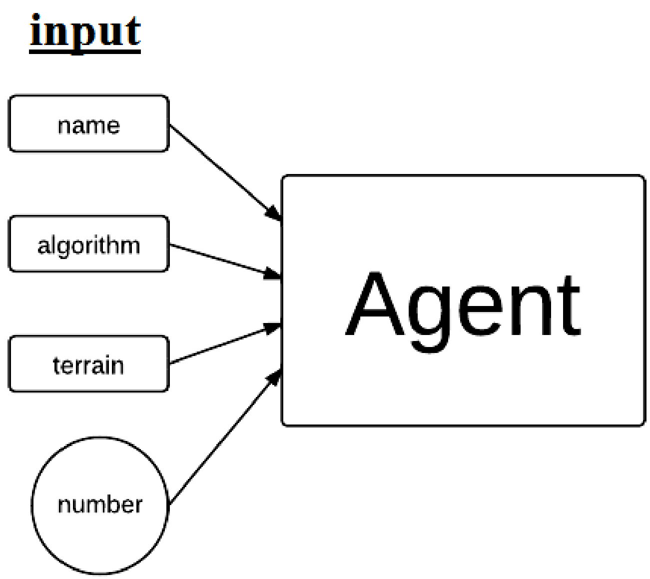


Figure 8.1: Prior to the start of a program, 3 files and a number are downloaded onto an agent’s memory.

In figure 8.1, we see that there are three files and a number downloaded to the agent before it is launched with the swarm:

- *name* – this is the name of the agent to which agents can connect to, e.g. NXT-3, identifying them as a member of the swarm

- *algorithm* – this is the identical code which each member of the swarm runs

- *terrain* – this is a text file which contains the name of the terrain the agent will be set for, e.g. “land” or “water”

- *number* – this is the number of agents in the swarm

The algorithm is designed such that it knows which sensor (which is manually attached to the agent) to use because there is a function in the algorithm which calls the terrain file into the function to determine which terrain, e.g. land or water, it is supposed to operate in. The program begins by reading the file and uses one or the other sensor, as shown in figure 8.2.

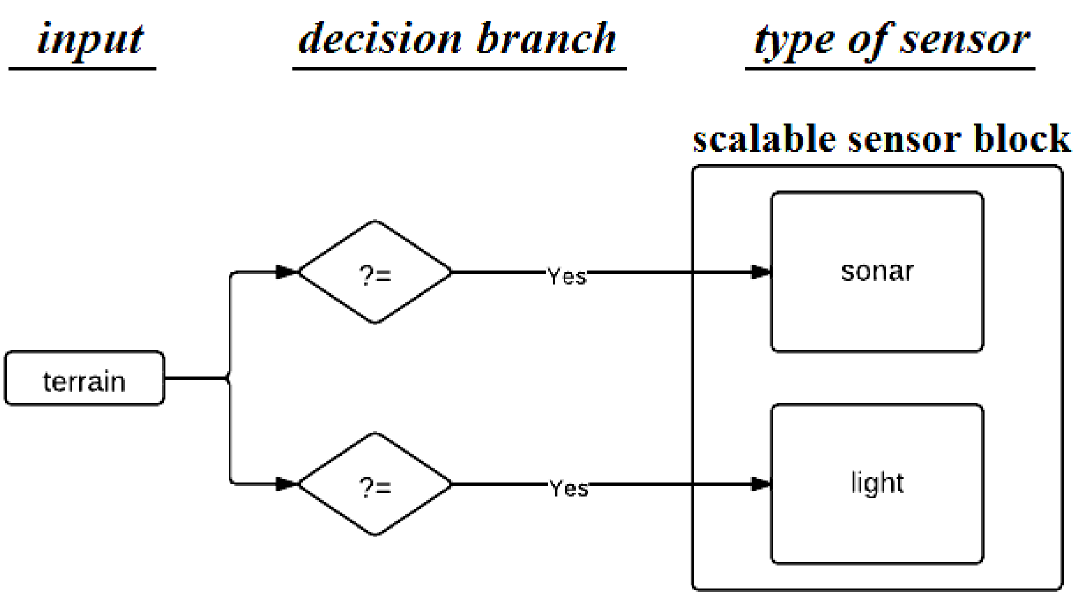


Figure 8.2: The terrain file is fed into a decision block, which reads the file’s text. If it contains “land”, then the type of sensor to use for collision avoidance is the sonar. If the file says “water’, then the agent will use light.

It is worth stating that different electronic sensors may use a different scale which determines the range of values used from the minimum to the maximum. For example, the sonar uses a range of 0-255, which the light uses 0-100. In order to get these sensor readings on the same scale so that they are on the same range of 0-100 of minimum-to-maximum readings they could be linearly scaled.

However, this would require the ideal scenario where all sensors are able to filter noise effectively. Even if this is not the case, the scalable sensor approach allows for individual settings of maximum ranges for each type of sensor by default. Threshold values are applied when that particular sensor is called, and figure 8.3 displays the front panel used to choose thresholds (as well as all other inputs related to the algorithm) using the LabVIEW environment.

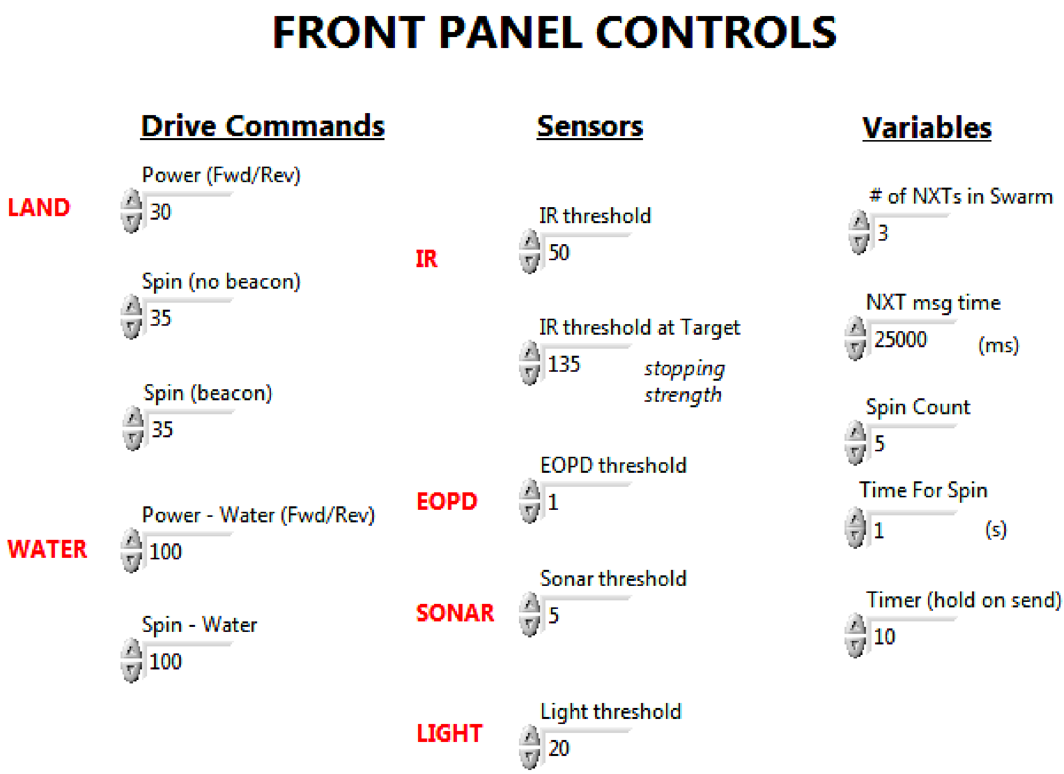


Figure 8.3: This shows user-defined values for variables related to the algorithm in LabVIEW.

This design is scalable in that if a user wanted to add a different type of physical sensor, then the terrain file would simply say this new name. Within the scalable sensor function, you would just add another sensor's code block into it, and the code would be receptive to the terrain file's text, reading it into the scalable sensor block, and calling the appropriate sensor based on what the text file said, without having to change any other code. Because of this, it can be shown that this type of programming allows for sensory input of any type, which would allow the swarm to detect any type of stimuli as long as the appropriate sensors were attached to the agent, demonstrating how the scalable sensor function can be applied to any algorithm which requires a collision detector. The only addition would be choosing the proper physical sensor for the agent and adding in the sensor's code to the scalable sensor block.

As I discussed in Chapter 5, communication within swarms is usually implicit or explicit. The swarm which I built uses an explicit, direct connection in a link between agents. However, one needs not to limit a robot to only one of these forms of communication. Building on the previous example of how you could create a program for any terrain via scalability, you could not only scale the type of sensor to be used, but also the number of sensors to be used, with no limit placed on the form of communication – only the physical attachment of the sensor and the addition of the device's code into the scalable sensor block would be all that is required for use after adding the chosen type of sensor to the terrain text file.

Up until now, I have discussed the notion that any number of sensor detections will result in a choice of behavioral response to some predetermined threshold, which then affects the behavior of the agent. For example, if the light sensor's value drops below a certain threshold, which would indicate that an obstacle is near, the agent would want to alter its path to avoid collision. However, if multiple sensors are employed, then how does the robot know which sensor is the one to activate the behavioral change of the agent? The answer I discuss in the next section.

8.2 System Intelligence and Behavior

In order to control a robot's behaviors while simultaneously making decisions based on data obtained from sensory input, one must understand the physical limitations of the platform being designed for ahead of time and implications of the choices for the robot's behaviors as an outcome on account these choices. The main physical attributes of the system which are important to keep in mind is the processing speed, memory size limitations, and advantages and disadvantages of parallel processes. Regardless of which platform a program is design to run on, there are optimal ways of processing information and using sensory input which ultimately affects the actions of not only one autonomous agent, but the collective and emergence capabilities of the swarm as a whole.

It is therefore necessary that the most important sensory data received by the agent is made to have the highest priority for making decisions. While the global goal of the swarm is for an agent to arrive at the beacon, this is not the most important to the agent's local behaviors. Because not all agents are capable of water entry, a method must be determined in how to decide what is water and what is land. If the terrain can be identified as either land or water, then an agent will check its terrain capabilities and make a decision of what it can do based on the surface reflectivity of a terrain.

The rationale here is that if an agent is meant for land-only traversing, and it is nearing the edge of water, then the most important thing is to alter its behaviors to avoid the water so that it does not get electrically destroyed by sinking. Even if a collision did occur, it may not necessarily destroy it (e.g. depending on the speed of the agent(s) and objects involved). Therefore, the EOPD sensor is the top-priority affecting an agent's behavior, since the values it senses directly affects the decision about the agent which could result in life or death of it. Because all three sensory inputs - the IR Seeker, collision sensors, and EOPD compete for the same scarce resource – the behavior of the agent's motors - the EOPD sensor is the highest-priority of the arbiter. [15]

This makes sense in that while the goal of the scenario of the agent is to get to the beacon, the beacon cannot be reached if the agent is not able to move because it did not yield to a hazard and thus is now rendered inoperable or has failed. Figure 8.4 shows how these sensory inputs are arranged, with arbiter values of 1, 2, and 3 being assigned fixed-priority levels to the EOPD, Collision Sensors, and IR Seeker, respectfully.

Each one of these sensors has independent instrument readings, along with associated threshold values which are set to trigger different actions and behaviors for each sensor if local sensor readings reach the respective thresholds.

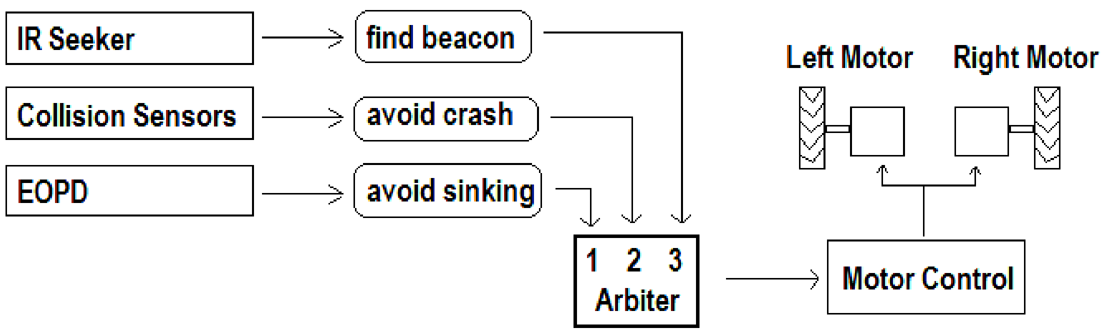


Figure 8.4: The three types of sensors are input into the arbiter, which reads the fixed priority of each one of the sensors. Whichever one has the highest number, where 1 is the highest, takes priority in controlling the motors of the agent, since this is scarce resource. [15]

However, the EOPD has an additional role which it plays after the signal has been found, and when an agent is headed towards it. If the EOPD sensor's value drops below the set threshold, indicating it is encountering water, it will check its terrain file. If the terrain file says "water", then it can enter water and will roll into it. What happens here is that because the agent knows it is in water, it will increase the motor's speed to help drive the ball through the water faster, since the wheels generate the spinning motion of the ball which the agent is in. (Friction on the ball and momentum of the agent inside help propel it in the direction the wheels spin in). If the agent then exits the water (such as would be the scenario if the agent had been separated from the beacon by a body of water), then the EOPD sensor's value would rise above the threshold, and the speed of the motor's wheels would drop back to the default, indicating it is on land again, as shown in figure 8.5,

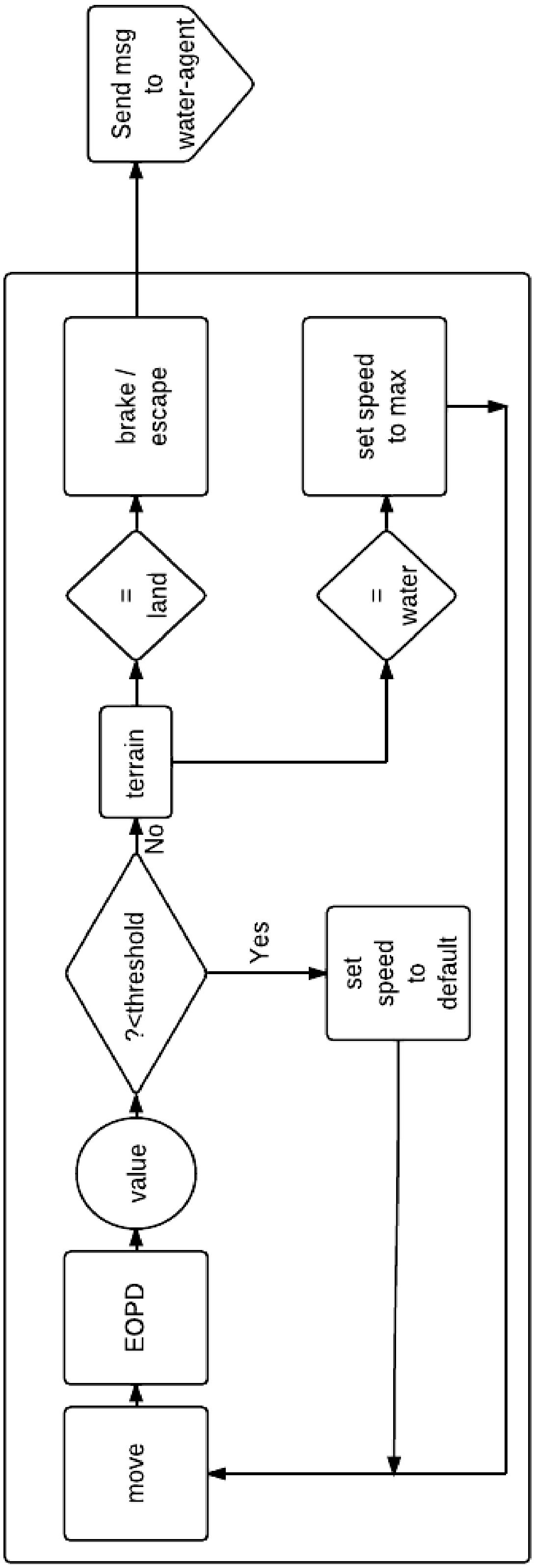


Figure 8.5: This loop shows the part of the algorithm which determines the driving actions of an agent heading towards the beacon, regardless of whether the beacon is located on land or water.

If the terrain file says "land", then the agent will stop and send a message to the nearest water agent instructing it to head towards it.

In general, the methods of development employed within the algorithm offer excellent control and scalability because of the advantage offered by parallel processes. The behavior-based architecture is structured such that the sensory readings of the sensors determine the behavior of the motors. Since the lower level, or *primitive* behavior, which consists of simple commands, is to always engage in motor actions (drive forward, reverse, brake), then the higher level decisions of these behaviors regulates control over them in design known as *subsumption architecture*. [15,16] This is best represented through the general parallel processes run in the algorithm, as shown in figure 8.6, which demonstrates the general idea behind the behavior-based intelligence of an agent in response to a sensor's threshold:

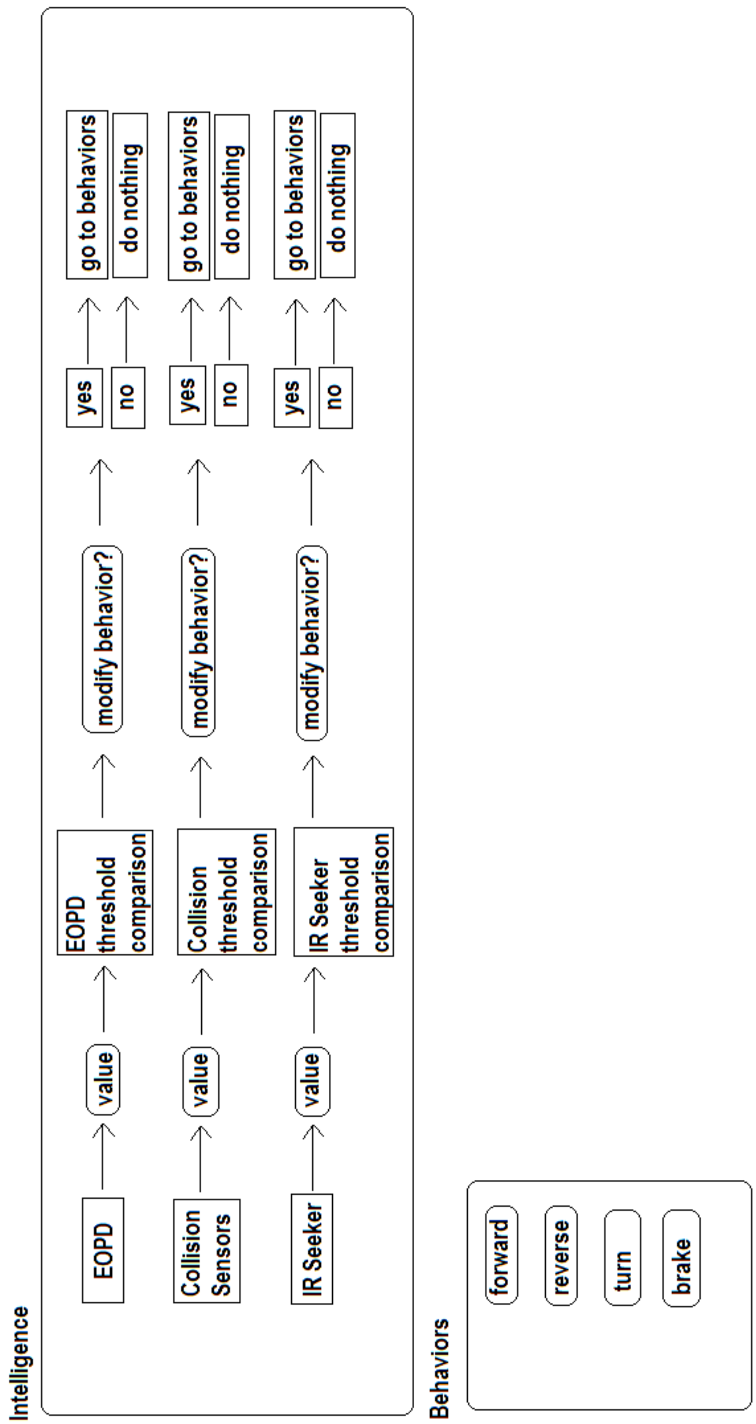


Figure 8.6: This figure illustrates parallel processes of subsumption architecture for behavioral-based controls of an agent. The intelligence loop atop reads sensor values, and alters simple, primitive behaviors of the motors functions in the below loop in response to them.

8.3 Final Remarks

In this section, I showed how a single algorithm can be used on all agents. By adding the terrain file declaring an agent for either “land” or “water”, and placing the water-agent in the plastic ball, this has created a heterogeneous swarm. This adds flexibility to the swarm by allowing it to also be homogeneous by only using land-agents, or only consisting of water-agents.

I also showed that while the overall goal of the swarm is to reach the beacon, the top-priority of an agent is to avoid water if it is a land-agent. This shows that even though the swarm’s goal is reached upon conclusion of reaching the beacon, it is the emergent behavior of the agents which achieves this goal, regardless of terrain the beacon is on. Additionally, the scalable sensor block uses the terrain text file to determine which sensor is to be used in order to prevent agents from collisions. Since the EOPD and the sensors work within the arbiter to fulfill the goal of the agents and the swarm to reach the beacon, it is the order of the arbiter which determines the most important aspects which the swarm needs for survival.

Chapter 9: Conclusions

The swarm of the collective intelligences and emergent of the swarm demonstrated the robustness and practicality of the algorithm I developed to reach the beacon by overcoming challenges faced during the water-hazard and lost-agent scenarios. The dynamic ad-hoc mesh I created successfully allowed the swarm to connect during broadcasting with a 78% overall success rate. The swarm was successfully able to reach the beacon 88% of the time in the water-hazard scenario, and 80% of the time in the lost-agent scenario, for an overall success rate of 84%. This demonstrates that in cases where connection fails, and agents assume lost connection with the swarm, the swarm is still able to show excellent ability to reach the beacon despite any problems in distribution of communication between them. These scenarios were conducted indoors because of issues with IR reception between the IR beacon and the IR Seeker on the agents, which restricted testing to a small amount of time outdoors in the evening, thus making repeatability impracticable.

Failure of tests to not reach the beacon were likely due to problems of the water-agent becoming stuck at a wall or obstacle on account of significant scratches on the ball which made the light sensor reflect in the ball, instead of penetrating it and sensing the obstacle, which would have instructed the agent to avoid it. Another possibility may have been due to the position of the agent itself located further back in the ball while driving, in which an obstacle was out of range for detection by the light sensor. Issues of connectivity most likely accounting for the 22% failure rate when the dynamic ad-hoc mesh did not communicate was due to either a busy signal on the agent which was intended to receive a message and had currently timed-out, or because the firmware did not allow the agents to disconnect completely from each other, even though they were programmed to.

The land-agent was 100% successful in differentiating land from water, as well as simulated land from the simulated water trap, which demonstrated the EOPD sensor was excellent as used within these tests. The IR Seeker, however, presented problems with practicality of outdoor testing, as it was limited to only short periods it could be used in the evening, but this was resolvable by testing indoors, which allowed for repeatable testing.

The driving capabilities of the water-agent could be extended to use on low-friction surfaces where the land-agent would find difficulty in traversal or mobility, such as on ice, slippery liquids, or chemicals, such as oil or grease. These features add to the qualities and capabilities of the water-agent as an integral agent of the heterogeneous swarm. I find it compelling that by the addition of an inexpensive, plastic ball, the average land-agent is able to gain abilities over the other agent's by giving it capabilities to traverse water and have low-impact protection.

It should be noted that the water-agent does have a few disadvantageous to the land-agents. For example, due to its larger size, it would not be able to fit in smaller places that land-agents can. Additionally, it is harder to control, takes more power to operate, and is slower in response than the land-agent. These issues were exacerbated in the current study because the of 0.5 inch difference in the diameter of the hemispheres of the ball, as well as difficultly in creating a good fit in taping them together to form a symmetrical sphere, which would have allowed for more accurate driving.

However, the water-agent's weaknesses are overshadowed by the strength of the addition of it within the swarm. These unique abilities of the swarm have the advantage that agents can be identical and run the same algorithm, and its inclusion in the swarm provides greater possibilities in terrain traversal which could not be achieved with a homogeneous swarm.

9.1 Future Work

The data logger used as part of the algorithm on the agents allows for a remote computer to target individual ones and request their onboard file at any time, and thus will be able to see what the sensors are seeing, where the agent has been, and what messages were sent and received. This could compliment an addition of a video camera onboard an agent, which could give remote operators visual feedback of the field.

Ideas of testing the system amongst several autonomous modes of travel could be incorporated with a heterogeneous swarm of, for example, unmanned ground vehicles, water-agents, and quadcopters. The recognition of a beacon now would not only be taken into account how far the agent is from it, but also how fast it is estimated the agent could arrive at the beacon. This would depend on several factors, such as terrain, obstacles, fuel levels (e.g. battery or solar), etc. Stronger communication than the limited range provided by Bluetooth, such as Wi-Fi, would allow for several more agents to be added into the heterogeneous swarm across a larger area, with constant connections between agents through multiple access channels. Because the outline for a universal code can be used on any platform employing the basic abilities to sense the environment and make behavior or communication decisions between agents based upon sensory data, this allows for the code to be designed robustly, in that sophistication can be increased for each block for different agents. This could again reduce development and debugging time, and may potentially allow for faster system upgrades than if the algorithms were separately designed for different agents. The terrain file ID would allow for the specific calls to the algorithm which would be relevant to the agent's terrain of operation.

If GPS capabilities were added to the NXT, agents which encountered hazards or obstacles could relay the locations of them to other agents and tell them to avoid them. Because GPS has poor reception indoors, this would have to be moved to outdoor testing areas. However, it would probably require a different type of beacon and sensor than the IR ball and IR Seeker due to the encountered difficulties and repeatability problems of the IR range and reception outdoors.

One idea for increasing the intelligence of an agent (and hence the swarm) would be by using a dedicated platform for sensors and data processing, such as a Superpro by HiTechnic. [48] This would allow for data to be gathered, in order to give the agent higher cognitive abilities, and statistical measurements could be applied to process sensory data. The result of the statistical processing, a single number, or even a Boolean data type could then be sent to the NXT to execute a lower primitive behavioral motor command. Therefore, sophistication of the agent's cognitive decision making skills can be improved upon through means of more efficient data processing. The more sophisticated the sensory data processing becomes, the more an autonomous agent approaches using artificial intelligence to make decisions, much like we make our best guess based on the sensory information we received from the environment. Therefore, higher level sophistication is possible by accessing an externally connected platform which could still take advantage of the lower primitive behaviors, yet with more rationale to control them. This would also solve the issue with the limited memory of the NXT with the addition of the external board.

Another idea is that while I used an external file to dictate the terrain (and hence which sensor an agent should use) based on the string contained in it, this file could be eliminated for simple terrain identification by the introduction of a code which would determine which terrain it is meant for by recognizing the sensor which is plugged into it. For example, because the sonar uses i2C to communicate with the NXT, code could be written to look for data exchanged over i2C. If the NXT reads this exchange, then the agent would know to use sonar. Because the light sensor is analog, and hence does not use iC2, a lack of this data exchange would then determine the NXT to be a land-agent instead. (However, the incorporation of the text file does open the door to add more information to it, which could be exchanged when requested between agents and the swarm to provide useful information for various goals.)

9.2 Future Research Directions with Potential Applications

9.2.1 Sensor Scalability

The idea of a system which can be scalable in both swarm size, as well as in heterogeneous nature makes it an advantageous way to incorporate more modes of terrain travel by which agents within the swarm are capable of. Since intelligence and behaviors are designed through subsumption architecture and arbiter design, as discussed in chapter 8, this would allow an agent's intelligence to be scaled to detect more in the environment – everything from obstacles, to chemicals in the air or water, etc. Additionally, how to process information can become more sophisticated, such as through statistical tests and machine learning, through updating of a database of past events (through expandable memory and/or cloud computing as a separate platform to handle the intelligence processing - if a network is available), and then reacts to the commands however the intelligence produces a final decision based upon the sensory data interpretation.

Because all agents are running the same code, any upgrades to the swarm can be broadcasted, instead of updating one agent at a time. This is advantageous for using the same code in heterogeneous (as well as homogeneous) robotics. This allows for the possibility of a reduction in operating costs and time spent on design by not having to custom-fit an algorithm to individual agents.

9.2.2 Military

While the swarm of agents I designed for this thesis can survive when other agents within the swarm are missing, there could also be a way to add additional agents into the swarm. Using the idea presented in the scalability section, you could grow the swarm by having others request inclusion into it, which would require a new agent to exchange some form of a password with an existing swarm agent. Not only would this allow a swarm to grow in size, for example, if new agents were to be deployed into an area of operation, but this also could extend the previous idea into friend-or-foe autonomous determination, by having agents request knowledge from others for identification. If this robot was determined to be a foe, the agent could alert other agents in the field about the detection of the foe. In the event that if one agent got accepted into the swarm, you could have a “code of the day” broadcasted to agents from a control operator elsewhere that has a list of the agents, and thus, upgrades would only target the list, as well as for codes. This would render this “enemy robot” an impostor when asked for ID and password, in case somehow adversaries got a hold of an agent and figured out the password.

9.2.3 Emergencies and Disaster Response

As I discussed in the introduction, when an emergency or disaster occurs, officials often need to regroup quickly and restrategize under stressful conditions of how to handle the unexpected situation. Domestically, many cities across America are lacking in emergency response times and are either masking them, refuse to provide them, or use inefficient or incomplete means of measuring them. [22, 23] While computer-aided dispatching can save time in emergency scenarios, it still is subject to fallacy to commuter traffic delaying response to the scene. If a person is in distress or sees an emergency, the call center could be replaced in the future by usage of a cloud computing system which could not only find the nearest emergency responder in terms of distance, but also one in which would be the fastest to respond through traffic analysis of current roadway conditions. The closest to respond to the distress call is analogues to the usage of the agents within my thesis.

Appendix A: Technical Specifications of Hardware Components

Appendix A contains the technical specifications of the hardware components used in the design of the land and water-agents, as discussed in Chapter 4 and shown in figure 4.1.

Component	**Description**	**Manufacturer**
4.1 NXT (brick)		LEGO
Main processor	Atmel® 32-bit ARM® processor, AT91SAM7S256 256 KB FLASH 64 KB RAM 48 MHz	
Co-processor	Atmel® 8-bit AVR processor, ATmega48 4 KB FLASH 512 Byte RAM 8 MHz	
4 input ports	6-wire interface supporting both digital and analog interface 1 high speed port, IEC 61158 Type 4/EN 50170 compliant	
3 output ports	6-wire interface supporting input from encoders	
Bluetooth	CSR BlueCoreTM 4 v2.0 +EDR System Supporting the Serial Port Profile (SPP) Internal 47 KB RAM External 8 MB FLASH 26 MHz	
4.2 Infrared Seeker V2	Set to receive signals at 1200 Hz	HiTechnic
4.3 Ultrasonic Sensor	Collision Detector range: 0-255 cm, precision: +/- 3 cm	LEGO
4.4 Light Sensor	Collision detector, sampling rate: 3 ms	LEGO
4.5 EOPD Sensor	Terrain detector sampling rate (used): 300 ms	HiTechnic
4.6 Servo Motors	170 rpm (no load)	LEGO
4.7 IR Electronic Ball	Set to modulated 1200 Hz (AC)	EK Japan Co., Ltd
4.8 Plastic Ball	Vacuum Formed Sphere Transparent 2 Piece 40 Gauge, 18	Barnard Ltd.

Appendix A: Hardware specifications.

Bibliography

[1] Blitch, J. (1996). Artificial Intelligence Technologies for Robot Assisted Urban Search and Rescue, *Expert Systems with Applications, 11(2)*, 109-124.

[2] Jacoff, A.S. and Messina, E. R. (2007). Measuring the Performance of Urban Search and Rescue Robots, *Proc. Technologies for Homeland Security, IEEE,* 28.

[3] Durrant-Whyte, H. (2001). A Critical Review of the State-of-the-Art in Autonomous Land Vehicle Systems and Technology, *SANDIA REPORT, SAND2001-3685.*

[4] E. G. Boring. (1923). Intelligence as the tests test it. *New Republic, (35),* 35-37.

[5] Eberhart, Russell C. and Kennedy, James. (2001). *Swarm Intelligence*. San Francisco, CA: Morgan Kaufmann.

[6] McCorduck, Pamela. (2004). *Machines Who Think*, 2nd ed. Natick, MA: A. K. Peters, Ltd.

[7] Crevier, Daniel. (1993). *AI: The Tumultuous Search for Artificial Intelligence*. New York, NY: Basic Books.

[8] Rodney, B. (1991). Intelligence Without Representation, *Artificial Intelligence, 47(1-3),* 139-159.

[9] Scott, Shane and Shanker, Thom. (2011, October 1). Strike Reflects U.S. Shift to Drones in Terror Fight. *The New York Times.* Retrieved February 14, 2012, from http://www.nytimes.com/2011/10/02/world/awlaki-strike-shows-us-shift-to-drones-in-terror-fight.html

[10] Bonabeau, Eric, Marco Dorigo, and Guy Theraulaz. (1999). *Swarm Intelligence: From Natural to Artificial Systems*. New York, NY: Oxford University Press.

[11] Reynolds, C.W. (1987). Flocks, Herds, and Schools: A Distributed Behavioral Model, *SIGGRAPH '87 Conference Proceedings, 21(4)*, 25-34.

[12] Bonabeau, Eric and Meyer, Christopher. (2001, May). Swarm Intelligence: A Whole New Way to Think About Business. *Harvard Business Review*, 104-114.

[13] Johnson, Steven. (2001). *Emergence - The connected lives of ants, brains, cities, and software*. New York, NY: Simon & Schuster.

[14] (2000, October 30). National Defense Authorization, Fiscal Year 2001, *Public Law 106-396, 106th Congress*, 114 STAT. 1654A1-512.
[15] Jones, J. (2004) *Robot Programming: A Practical Guide to Behavior-Based Robotics*. New York, NY: McGraw-Hill Companies, Inc.
Note: Adopted figure is from same source, p. 76.

[16] Brooks, R.A. (1986, April). A Robust Layered Control System for a Mobile Robot, *IEEE Journal of Robotics and Automation, RA-2*, 14-23.

[17] Shaughnessy, Larry. (2012, January 26). Budgeting for a New military Vision. *CNN*. Retrieved February 7, 2012, from http://security.blogs.cnn.com/2012/01/26/budgeting-for-a-new-military-vision

[18] Hirsch, Jen. (2011, December 12). MIT researchers unveil autonomous oil-absorbing robot. *MIT Media Relations*. Retrieved December 12, 2011, from http://web.mit.edu/press/2010/seaswarm.html

[19] Saenz, Aaron. (2009, November 16). Swarm of underwater Drones To Help Explore Ocean. *Singularity Hub*. Retrieved December 12, 2011, from http://www.singularityhub.com/2009/11/16/swarm-of-underwater-drones-to-help-explore-ocean

[20] Onyx Autonomously Guided Parachute Systems. *Atair Aerospace*. Retrieved December 11, 2011, from http://www.atair.com/onyx

[21] Nave, R. Rayleigh Scattering. *Department of Physics and Astronomy, Georgia State University*. Retrieved April 2, 2012, from http://hyperphysics.phy-astr.gsu.edu/hbase/atmos/blusky.html

[22] Davis, Robert. (2005, May 20). The price of just a few seconds lost: People die. *USA Today New.* Retrieved April 3, 2012, http://www.usatoday.com/news/nation/ems-day2-cover.htm

[23] (1998, April). Review of Emergency Response Statistics. *A Report by the Audit Services Division, Report #237, Office of the City Auditor, Portland, Oregon*.

[24] Yogeswaran M. and Ponnambalam S. G. (2010). Swarm Robotics: An Extensive Research Review, *Advanced Knowledge Application in Practice*, *Igor Fuerstner (Ed.), Sciyo, Published: November 02, 2010*, 259-263.

[25] Table reproduced from Swarm Robotics: An Extensive Research Review, *Advanced Knowledge Application in Practice* (p. 263), by Yogeswaran M. and Ponnambalam S. G. (2010). Igor Fuerstner (Ed.)*, Sciyo, Published: November 02, 2010.*

[26] (2012). Cuts, Consolidations, and Savings, Fiscal Year 2013. *Budget of the U.S. Government*, *Office of Management and Budget*.

[27] Public Law 106-398 – Oct. 30, 2000. National Defense Authorization, Fiscal Year 2001.

[28] Gertler J. (2012, January 3). U.S. Unmanned Aerial Systems, *Congressional Research Service Report for Congress 7-5700, R42136.*

[29] Changying W., Dezhong W., Li C., Xianglin M., Zhengping H. (2010). Study on improved ant colony algorithm of swarm intelligence algorithm, *3rd International Conference on Advanced Computer Theory and Engineering (ICACTE)*, *5,* 639-641.

[30] Dorigo M., Gambardella L. M. (1997). Ant colony system: A cooperative learning approach to the traveling salesman problem, *IEEE Transactions on Evolutional Computation, 1(1),* 53-66.

[31] Fisher, Lens. (2009). *The Perfect Swarm*. Basic Books, New York, NY.

[32] Martinez, Michael. (2009, March 26). Swarm Theory and Web Communities. *SEO Theory and SEO Theory and Analysis Blog*. Retrieved March 3, 2012, from http://www.seo-theory.com/2009/03/26/swarm-theory-and-web-communities

[33] (2011, May 31). 'Digital ants' could be key to protecting power grid, networks. *WRALtechwire*. Retrieved May 31, 2011, from http://wraltechwire.com/business/tech_wire/news/blogpost/9663771

[34] UAV SWARM Health Management Project Information. *Aerospace Controls Laboratory at MIT*. Retrieved 15, 2012, from http://vertol.mit.edu/prjinfo.html

[35] Lloyd, T. (2001, June 6). When swarm intelligence beats brainpower. *Telegraph*. Retrieved March 3, 2012, from http://tuvalu.santafe.edu/~vince/press/swarm-intelligence.html

[36] Abe K, Sugawara K., Sano M., Watanabe T., Yoshihara I. (1999). Foraging behaviour of multi-robot system and emergence of swarm intelligence, *Conference Processings, IEEE International Conference on Systems, Man, and Cybernetics, Conference Proceedings. 3,* 257-262.

[37] Ducatelle, F., Di Caro, G. A., Gambardella, L. M. (2010). Cooperative self-organization in a heterogeneous swarm robotic system. *In Proceedings of the 12th*

annual conference on Genetic and evolutionary computation, GECCO, ACM, New York, NY, 87-94.

[38] Lee D., K, Seo S., Sim, K. (2008, April). Online Evolution for Cooperative Behavior in Group Robot Systems, *International Journal of Control, Automation, and System, 6(2)*, 282-287.

[39] Montes De Oca, Roldan M. A. (2011, July). Incremental Social Learning in Swarm Intelligence Systems, *Doctoral Dissertation, Université libre de Bruxelles, Brussells, Belgium*.

[40] Beaver J., Cui X., Pullum L., Klump B., Stiles E., Treadwell J. (2010, August). The Swarm Model in Open Source Software Developer Communities, *IEEE Second International Conference on Social Computing*, 656-660.

[41] NXT IRSeeker V2. *HiTechnic*. Retrieved April 30, 2012, from http://www.hitechnic.com/cgi-bin/commerce.cgi?preadd=action&key=NSK1042
Note: Reproduced figure is from same source.

[42] Vacuum Formed Sphere Transparent. *Barnard, Ltd*. Retrieved April 30, 2012, from http://www.barnardltd.com/product.jsp?prodId=3874&catId=703

[43] (2009, February 16). NXT Matlab Bluetooth Router. *BrickEngineer: LEGO design*. Retrieved April 4, 2012, from http://www.brickengineer.com/pages/tag/nxt

[44] NXT EOPD. *HiTechnic.* Retrieved April 30, 2012, from http://www.hitechnic.com/cgi-bin/commerce.cgi?preadd=action&key=NEO1048
Note: Reproduced figure is from same source.

[45] LEGO® 9V Technic Motors compared characteristics. Retrieved April 4, 2012, from http://www.philohome.com/motors/motorcomp.htm

[46] Istenes, Z., Kovacs, T., Pasztor, A. (2009, May). Swarm intelligence simulation with NXT robots using Piconet and Scatternet, *SACI '09 5th International Symposium on Applied Computational Intelligence and Informatics,* 199-204.

[47] Brigandi, S.; Field, J., Yunfeng Wang. (2010, July). A LEGO Mindstorms NXT based multirobot system, *IEEE/ASME International Conference on Advanced Intelligent Mechatronics,* 135-139.

[48] NXT SuperPro Prototype Board. *HiTechnic*. Retrieved April 29, 2012, from http://www.hitechnic.com/cgi-bin/commerce.cgi?preadd=action&key=SPR2010

[49] Beni, G., and Wang, J. (1989). Swarm Intelligence. *In Proceedings of the Seventh Annual Meeting of the Robotics Society of Japan, Tokyo, Japan*, 425 428.

[50] Hsu, Jeremy. (2011, September 7). Military Battles Information Overload from Robot Swarms. *Innovation News Daily*. Retrieved February 28, 2012, from http://www.innovationnewsdaily.com/553-battlefield-drones-information-overload.html

[51] Corrin, Amber. (2010, February 4). Sensory Overload: Military is Dealing with a Data Deluge. *Federal Computer Week*. Retrieved February 4, 2010, from http://fcw.com/articles/2010/02/08/home-page-defense-military-sensors.aspx

[52] Dille, M., Grocholsky, Moseley, M., B.P., Nuske, S.T., Singh, S. (2011, August). Air-Ground Collaborative Surveillance With Human-Portable Hardware, *AUVSI Unmanned Systems North America Conference.*

[53] Ultrasonic Sensor. *LEGO Group*. Retrieved April 30, 2012, from http://shop.lego.com/en-US/Ultrasonic-Sensor-9846
Note: Reproduced figure is from same source.

[54] Light Sensor. *LEGO Group*. Retrieved April 30 April, 2012 http://shop.lego.com/en-US/Light-Sensor-9844
Note: Reproduced figure is from same source.