Communication Free Robot Swarming

THESIS

Zachary C. Gray, Captain, USAF

AFIT/GCE/ENG/09-03

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the United States Government.

AFIT/GCE/ENG/09-03

Communication Free Robot Swarming

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Computer Engineering

Zachary C. Gray, B.S.E.E., B.S.C.E

Captain, USAF

March 2009

AFIT/GCE/ENG/09-03

Communication Free Robot Swarming

Zachary C. Gray, B.S.E.E., B.S.C.E

Captain, USAF

Approved:

/signed/	26 Feb 2009
Dr. Gilbert L. Peterson(Chairman)	date
/signed/	26 Feb 2009
Dr. Gary B. Lamont (Member)	date
/signed/	26 Feb 2009
Dr. John F. Raquet (Member)	date

Abstract

As the military use of unmanned aerial vehicles increases, a growing need for novel strategies to control these systems exists. One such method for controlling many unmanned aerial vehicles simultaneously is the through the use of swarm algorithms. This research explores a swarm robotic algorithm developed by Kadrovach implemented on Pioneer Robots in a real-world environment. An adaptation of his visual sensor is implemented using stereo vision as the primary method of sensing the environment. The swarm members are prohibited from explicitly communicating other than passively through the environment. The resulting implementation produces a communication free swarming algorithm. The algorithm is tested for performance of the visual sensor, performance of the algorithm against stationary targets, and finally, performance against dynamic targets. The results show expected behavior of the swarm model as implemented on the Pioneer robots providing a foundation for future research in swarm algorithms.

Table of Contents

Page

Figure Page

Figure Page

List of Tables

List of Abbreviations

Abbreviation		Page
UV	Unmanned Vehicles	1
UAV	Unmanned Aerial Vehicles	1
USAF	United States Air Force	1
SEAD	Suppression of Enemy Air Defenses	1
UCAV	Unmanned Combat Aerial Vehicles	1
2D	two-dimensional	4
AFIT	Air Force Institute of Technology	5
AFRL/RY	Sensors Directorate, Air Force Research Laboratory	5
AFB	Air Force Base	5
3D	3-Dimensional	30
2D	2-Dimensional	30
CCD	Charge-Coupled Device	32
MOS	Metal Oxide Semiconductors	32
HSI	Hue Saturation Intensity Color Space	42
RGB	Red Green Blue Color Space	42
RLE	Run Length Encoding	45

Communication Free

Robot Swarming

I. Introduction and Overview

Future conflicts will see an increased role for Unmanned Vehicles (UVs) as we strive to improve the safety, accuracy, and robustness of our weapons systems while limiting the risk to service personnel. The development of Unmanned Aerial Vehicles (UAVs) is one focus of the United States Air Force (USAF) striving to meet this difficult military requirement. Additionally, Congress has mandated that one third of the deep strike force of the USAF be unmanned by 2010 [50]. To meet this ambitious goal requires a large effort in the field of autonomous control to make these vehicles better military options for commanders.

Contemporary operations in both the Iraq and Afghanistan theaters have demonstrated the effectiveness of the current UAV capability. The successes of the Predator UAV have been well documented including reconnaissance and limited strike capabilities. By 2010, the USAF plans to extend these capabilities to include suppression of energy air defenses (SEAD) using unmanned combat aerial vehicles (UCAV) [50].

Currently, UAVs are operated remotely by a ground controller. This is sufficient if the number of UAVs is small; however, as the siz e of our unmanned force increases the demand for well-trained support personnel and equipment will increase. One solution to this problem is to reduce this dependency by incorporating autonomous control behaviors. By allowing the UAVs to handle low-level details such as navigation, a single human controller can monitor several UAVs simultaneously.

This research focuses on the control of a large group of UAVs by a single user. The user control consists of communicating a mission plan (telling the system where to go), and the vehicles follow this mission plan without communicating between one another.

The goal of this research is to design a physical swarm robotic controller based on the simulation model presented in [30] and to evaluate the performance of the implementation. The target platform is the Active Media Pioneer P2-AT8 robotic platform running the ARIA software package. Our sensor for this work is a Videre STH-MDCS-C Stereo Camera system. Since we are interested in military applications we further restrict the problem by not allowing communication between the swarm members and restrict sensing to the passive camera system only. The only information available to any member of the swarm is what is provided through its vision sensor.

1.1.1 Goal. The primary goal of this research investigation is to design, implement, and evaluate the performance of a "swarm" of robots controlled by Kadrovach's algorithm on a Pioneer P2-AT8 robot. The objectives outlined in the next section yield define the approach for achieving these goals.

1.1.2 Objectives. In order to achieve the goal of this investigation, the effort is decomposed into three objectives. These objectives are further decomposed into research tasks to ensure completion. The first objective is the understand the design and implementation of Kadrovach's swarm model on the Pioneer robot. The next objective is to understand the Videre camera hardware and software and to model the visual sensor in [30]. The final objective is to design experiments and to analyze the performance of the physical swarm implementation.

1.1.2.1 ***Objective 1: Design and Implement Kadrovach's Swarm Model.*** The design of the physical implementation of the Kadrovach's swarm behavior is the focus of the first portion of this research. To that end, Table 1.1 outlines the sub-objectives necessary to satisfy the first objective. Several tasks are needed to support the first objective. The first task requires understanding the fundamentals of Kadrovach's swarm model, the Videre cameras and software, and the ARIA robotic system. With that knowledge the visual sensor is implemented and is tested. After settling on a visual sensor design the swarm factors are decomposed into behaviors, implemented and tested.

Table 1.1: Swarm Model Design and Implementation Objective.

***Objective 1**: Design and Implement Kadrovach's Swarm Model.*
- Understand the swarm model
- Understand the Videre camera hardware and software
- Implement the visual sensor model
- Test the visual sensor implementation
- Understand the ARIA robot control system
- Decompose the swarm factors into independent behaviors
- Implement the swarm behaviors
- Test the swarm behaviors on a single robot

1.1.2.2 ***Objective 2: Design and Implement Kadrovach's Visual Sensor.*** The design of the physical implementation of the Kadrovach's swarm behavior requires an implementation of his visual sensor for sensing the environment around swarm members. Table 1.2 outlines the sub-objectives necessary to satisfy the second objective. To begin, the visual sensor assumptions of the design presented in [30] are reviewed and understood. Similarly, the camera system and software provided must be understood. Next, the interactions and integration of the swarm vision sensor into the ARIA robot control architecture must be understood. Armed with a fundamental understanding of the model, hardware, and software the method of identifying swarm members is selected. Finally, the visual sensor is implemented and tested.

Table 1.2: Visual Sensor Model Design and Implementation Objective.

***Objective 2**: Design and Implement Kadrovach's Visual Sensor Model.*
- Understand the Kadrovach's visual sensor model
- Understand the Videre camera hardware and software
- Understand the ARIA robot control system
- Identify method of identifying swarm members
- Implement the visual sensor model
- Test the visual sensor implementation

1.1.2.3 ***Objective 3: Performance Evaluation of the Robotic Swarm.*** The third objective of this research is to evaluate the performance of the implemented robotic swarm. Table 1.3 outlines the sub-objectives necessary to satisfy the third objective. The evaluation of the swarm begins with designing a set of experiments to evaluate the visual sensor. Once the performance of the visual sensor is understood, parameters are set to test

the performance of the swarm model in a variety of situations. Once the experiments are designed and conducted, the results are analyzed and conclusions presented.

Table 1.3: Performance Evaluation Objective.

***Objective 3**: Evaluation of the Robotic Swarm.*
- Design experiments to evaluate visual sensor
- Design experiments to test performance vs static target
- Design experiments to test performance vs dynamic target
- Conduct the experiments
- Analyze the results

This section outlined the three objectives used to ensure the completion of the goal outlined above. First, the swarm model described in [30] is designed. Then it is implemented on the Pioneer P2-AT8 robots. The second objective supports the first by designing, implementing and testing the visual sensor composed of the Videre cameras. The final objective conducts a performance evaluation of the entire system in static and dynamic scenarios. The next sub section describes the assumptions of this research.

1.1.3 Assumptions. The robots are not allowed to communicate explicitly except through the environment. This simulates the operation in a hostile environment where emitting energy into the environment could compromise one's location and ultimately mission success. Furthermore, this research assumes the robots are provided with an a priori mission plan, and assumes is that the robots may only move in a two dimensional (2D) environment. This means that the research focuses on a team of ground vehicles or a static altitude team of aerial vehicles. This reduces our motion models.

1.1.4 Risks. The Pioneer P2-AT8 robots have limited processing, memory, and sensing capabilities which poses risks to the successful fielding of a swarm behavior in real-time with visual sensing. The computational requirements of the visual sensor poses the greatest risk to the completion of this work. Other risks include the integration and capability of the various vendor provided software packages and the refinement of the notional swarm vision model of [30] to the actual hardware. A fundamental risk is the lack of a telemetry data for the evaluation of performance of the implemented system.

1.1.5 Contributions. This work provides two main contributions. The first is a physical platform for future swarm robotics research and applications at the Air Force Institute of Technology (AFIT). The second is a performance analysis of the implemented system.

1.2 Research Sponsor

This research is sponsored by the Sensors Directorate, Air Force Research Laboratory (AFRL/RY) Wright Patterson Air Force Base (AFB), Ohio. AFRL/RY strives to be the "Eyes and Ears of the Warfighter." The mission of the Sensors directorate is "is to ensure unequaled reconnaissance, surveillance, precision engagement and electronic warfare capabilities for Americas air and space forces, by conceiving, demonstrating and transitioning advanced sensors and sensor technologies." [1] Its core technology areas include: radar, active and passive electro-optical targeting systems, navigation aids, automatic target recognition, sensor fusion, threat warning and threat countermeasures. This thesis supports the this mission by researching methods to improve the control of autonomous vehicles and exploring methods of visually sensing one's environment. This work provides the warfighter with the ability to extend his own eyes and ears into the battlefield gathering critical information with minimal risk.

1.3 Thesis Outline

The remainder of this thesis is presented in four subsequent chapters. Chapter 2 presents relevant background information on swarm algorithms and computer vision. Additionally, Chapter 2 discusses in detail Kadrovach's swarm algorithm. Chapter 3 discusses the hardware platform and software platforms in detail and continues with a detailed description the software design of the implementation. Chapter 4 outlines the design of the experiments and provides the results and analysis of the experiments. Chapter 5 presents conclusions and recommendations for future research.

1.4 Summary

This chapter provides an overview of and necessary background information for understanding the research conducted. It formulated the problem statement. An outline of the goals, assumptions, objectives, risks, and contributions is presented. A discussion of the sponsoring organization is provided. The chapter concludes by outline the structure of the remaining document.

II. Background

This chapter outlines the background information necessary to establish the research foundation for the work presented. This research draws from two main areas: swarm robotics and computer vision. The first section discusses the area of swarm robotics, its foundations, related research in the field and culminates in a detailed description of the robotic swarm control algorithm used in this work. The second section outlines the foundations of computer vision as they apply to this thesis stressing image formation, image segmentation, and stereo vision. This chapter concludes with a summary of key points from this chapter leading into the discussion of the implementation hardware and software design in Chapter 3.

2.1 Swarm Robotics

Nature provides several models for cooperative behavior researchers study and build upon. A colony of ants, a flock of birds, a school of fish, and swarms of locusts all exhibit what researchers term *swarm behavior*. A swarm behavior refers to the globally observable shape or formation of individual emerging from local decisions by members of the swarm [9].

This section presents the foundations of swarm robotics, ongoing current research in swarm control algorithms stressing physical implementations, and finally presents a detailed description of the swarm control algorithm of Kadrovach which is the foundation for this work.

2.1.1 Foundations. Swarm robotics finds its inspiration from flocks of birds, schools of fish, colonies of ants, and hives of bees. Nature provides many examples of social animals and insects cooperating together with many benefits including protection, hunting, foraging and migration [9]. Swarm robotics, swarm intelligence and swarm algorithms all harness these observations in nature to produce mimicry in robotic and computer systems. These systems all exhibit a desired collective behavior which emerges from simple interactions between robots or agents and their environment.

Reynolds [44] noted swarms have two overarching diametrically opposite behaviors governing their interaction, separation and cohesion. In other words, swarms seek to remain a small closely knit grouping while at the same time avoiding collisions with other

members during motion. To model this, Reynolds proposed three characteristics of swarms: collision avoidance, velocity matching and flock centering. In demonstrating his *'boids* flocking simulation Reynolds seeded both swarm robotics and artificial life research fields.

Swarm robotics focuses heavily on large numbers of entities. The benefits of this approach are scalability, robustness, and ease of design [3]. Swarm algorithms seek to be scalable. That is adding more robots, agents, etc is expected to increase effectiveness of the swarm as whole. In the application of mine clearance [13, 14] for example, the scalability property allows more area to searched by simply increasing the number of robots performing the task without complex communication or other behaviors. Another beneficial aspect of swarms is that they tend to be extremely fault tolerant. A failure of one robot does not hinder mission success as the members are highly interchangeable. This is an extremely beneficial result for applications in complex, hostile or remote environments [41]. Finally, due to the large numbers of robots required to form a swarm the design of the individual robots is usually much more simple than an individual robot design to accomplish the same task. This speeds the design process for solutions.

The next section discusses the taxonomy which is used to guide the development of the related work. This is important for comparing and contrasting the current swarm robots, algorithms and applications found in literature.

2.1.2 Robotic Systems Taxonomy. For the discussion which follows, it is meaningful to define a taxonomy for the purposes of comparing different robotic systems. Dudek et al. [21] proposed a taxonomic system which he uses to characterize the assumptions and engineering tradeoffs when determining which system is best suited for a specific task. The Dudek system defines seven dimensions: size, communication range, communication topology, communication bandwidth, reconfigurability, processing ability, and composition. A summary of the taxonomy is shown in Table 2.1. Each of these axes is briefly discussed in the following paragraphs.

Size is the first category and essentially refers to the number of agents operating cooperatively in an environment. There are four size classifications namely: SIZE-ALONE, SIZE-PAIR, SIZE-LIM, and SIZE-INF [21]. A single robot acting alone within the environment is an example of a SIZE-ALONE system. Similarly, two robots acting cooperatively in

Table 2.1: Summary of Dudek's Taxonomy for Multi-Agent Robotic Systems.

Axis	Description
Size	Number of autonomous agents in the system.
Comm Range	Maximum sustainable comm distance.
Comm Topology	Number of in-range robots one can communicate with.
Comm Bandwidth	How much information can they transmit to each other?
Reconfigurability	How fast can the organization be changed?
Processing Ability	Computation model used by the agents.
Composition	Are all elements of the collective the same?

an environment constitutes a SIZE-PAIR system. A SIZE-LIM system contains more than two robots but the overall size of the collective is small when compared to the overall environment. Finally, the SIZE-INF descriptor indicates a system with a large number of robots when compared to the size of the overall environment. The distinction between SIZE-LIM and SIZE-INF classification is a property of the task . SIZE-INF refers to collectives in which the number of robots dedicated for the task is unbounded while SIZE-LIM indicates a bounded number of robots in the collective [21].

Communication range limitations are an important consideration when classifying robotic systems. Range is a function of the communication medium, equipment and the robot location. The are three classes in this dimension: COMM-NONE, COMM-NEAR and COMM-INF. If the robots do not communicate with other members explicitly they have a COMM-NONE communication range. Conversely, if the range of communication is limited to a local area (either in the Euclidean sense or network topological sense) the communication range is in the COMM-NEAR class. COMM-INF refers to systems which any robot may communicate with any other robot regardless of their separation.

The way in which the collective communicates with the other members is captured by Dudek's third dimension, Communication Topology. The classes of this dimension are: TOP-BROAD, TOP-ADD, TOP-TREE and TOP-GRAPH. TOP-BROAD captures the classic broadcast network in which any member may communicate with any other member but prohibits communication with a specific member only. TOP-ADD allows for both the communication to any robot and to a specific robot in the collective through the robot naming or network addressing. The TOP-TREE communication topology limits the communication flows to neighbors in the communications hierarchy. Systems in this class are

more prone to problems when a communication link fails than the previous two categories. TOP-GRAPH provides a measure of robustness by connecting members of the collective in a general graph structure in which redundancy can improve robustness.

Communication Bandwidth is another critical resource in multi-robotic systems and thus an important dimension in Dudek's taxonomy. BAND-INF, BAND-MOTION, BAND-LOW, and BAND-NONE make up the four classes in this dimension. In a BAND-INF system, the cost and overhead of communication is small enough to be ignored entirely. BAND-MOTION captures systems in which the cost of moving the robot is equal to the cost of communication but is limited or in which the communication is signaled through the environment as is observed in bee collectives [21]. BAND-LOW indicates a system where the cost of communication is higher than the cost of moving the robot. A BAND-NONE system has no communication between members and members an incapable of sensing one another in the environment.

The fifth axis is Reconfigurability. This dimension captures the collectives speed and ability to change shape, formation or structure. The three classes in this dimension are ARR-STATIC, ARR-COMM and ARR-DYN. If the structure of the collective is fixed throughout the task, the system displays ARR-STATIC reconfigurability. Systems which change structure or shape in predetermined systematic procedures display ARR-COMM reconfigurability. Finally, systems which adapt their structure arbitrarily are ARR-DYN.

The computational model which a collective uses is an important classification captured with the sixth axis, Processing Ability. The categories of processing ability are PROC-SUM, PROC-FSA, PROC-PDA, and PROC-TME. PROC-SUM refers to the simple linear summation unit as in [21]. PROC-FSA allows for finite state automaton computation, PROC-PDA systems have push-down automaton computation models, while PROC-TME is the most predominant form of computation model used in contemporary research, Turing Machine Equivalent.

The final taxonomic dimension, composition, is used to characterize the diversity of the system much like *social entropy* [7]. Social entropy is a numeric metric based on Shannon's Information theory which captures the diversity of a robot system. The three labels of composition are COMP-IDENT, COMP-HOM and COMP-HET. In a COMP-IDENT system

all members have the same physical and behavioral characteristics. COMP-HOM members possess the same physical characteristics; however, the programmed behaviors may differ. COMP-HET systems have members of diverse physical and behavioral characteristics.

The remainder of the discussion uses this taxonomy to describe and contemporary systems in swarm algorithms and formation control. For the purpose of organization, it is necessary to define the ordering of the axes in the taxonomy. Systems are first classified by the primary axis, the secondary axis, and so on down the ordering. The primary axis is the Reconfigurability, followed by Communication Bandwidth, Communication Topology, Communication Range, Size, and Composition. Since most of the contemporary literature have PROC-TME characteristics, the Processing Ability dimension is eliminated.

2.1.3 Related Work. Das et al. [18], developed a visual controller algorithm for maintaining rigid relative positions of a group of Clodbuster robots, see Figure 2.1. They address a related area of research known as *formation control*. The fundamental difference between a swarm of robots and formation of robots is the rigid relative location. Swarm robotics does not concern itself with the precise definition of relative positions of the swarm members while in formation control this is the objective. Their algorithm assumes an a priori leader whose motion defines the motion of the entire formation.

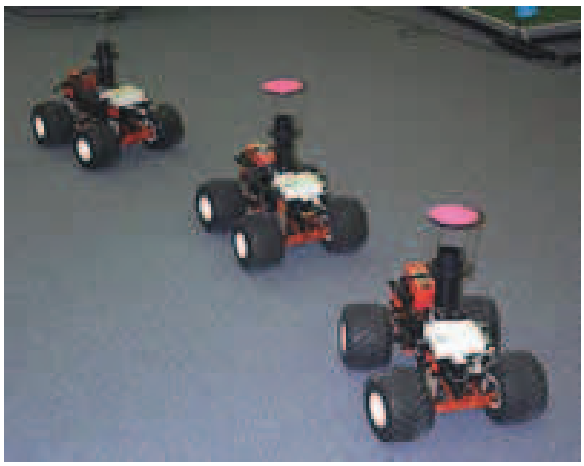

Figure 2.1: *Clodbuster Robot.* [18].

The Clodbuster is a modified commercial radio-control truck kit. The modifications have been made to improve shock absorption and to house an omnidirectional vision system,

a 2.4 GHz wireless video transmitter, and a battery pack. The robot has a servo controller on board for steering and a digital proportional speed controller for forward/backward motion. Up to eight Clodbuster robots are controlled remotely by a computer for experiments. Off board image processing allows for 30 Hz visual frame rates yielding real-time performance. An omnidirectional camera consisting of a parabolic mirror and a lens assembly mounted on a color CCD camera is used for navigation and control.

Each of the members have four controllers control theory based controllers. Only one controller is active at a time and the arbitration of these controllers is defined by an overall control graph. Although this work is similar in the overall goal of controlling the group robots, the methods differ considerably. First, all of the computation is handled by a centralized computer which communicates the heading and velocity changes to the robots via a wireless network. Secondly, inter-robot communication is allowed. The final difference is the sensor for this endeavor is an omnidirectional camera rather than the limited field of view camera on the physical system of our research. The classification of this work is COMM-INF,BAND-INF,ARR-COMM,SIZE-LIM,COMP-INDENT,TOP-ADD.

Hayes and Martinoli [26, 27] describe a swarm control algorithm designed to seek out an simulated chemical or "odor" source. The "odor" source problem explored here is a basic search application. Their focus is on the fusion of traditional sensing techniques to locate odor sources using moorebots, see Figure 2.2. In this work they demonstrate their plume odor detection algorithm up to 8 meters away as well as detect the source of the plume using low-level behaviors. Furthermore, they found that the integration of individual robot information increases the efficiency of the robotic system's performance.

Moorebots were originally designed by Owen Holland at the University of West England, Bristol, U.K. The robot is 24 cm in diameter and is equipped with two DC motor-driven wheels, a castor wheel, a 2 Mbit wireless LAN transceiver, and 12-bit A/D and D/A converters. See [2] for a more detailed robot description. Modifications from the basic configuration include: four infra-red range sensors for collision avoidance, a single odor sensor for plume detection, and a hot wire anemometer for wind speed. On-board high-level control is provided by a PC104 based Intel 386 processor running Linux. Low level control such as motor speed regulation is executed by dedicated hardware interfaced to the PC104 bus.

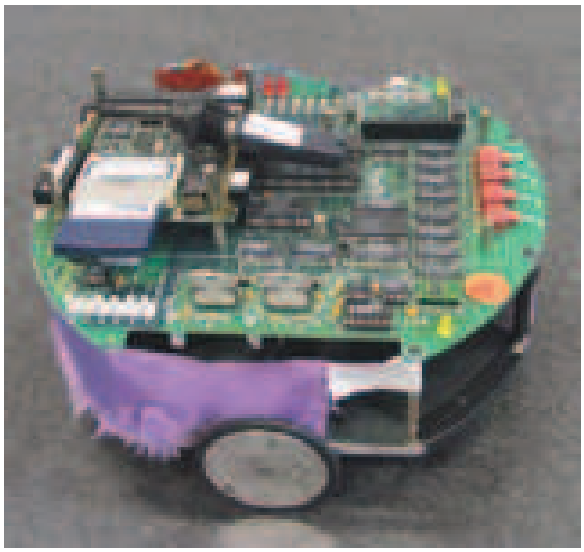

Figure 2.2: *Moorebot Robot* [27].

In [25], Hayes describes a leaderless flocking behavior with simulated sensor inputs to moorebots due to the lack of hardware implementation. The algorithm uses the perceived flock center to determine the alignment behavior of the flock without explicit knowledge of robot headings. They successfully demonstrated the leaderless flocking algorithm with simulated sensory inputs. The classification of this work is COMM-INF,BAND-INF,ARR-DYN,SIZE-LIM,COMP-INDENT, TOP-ADD.

Mataric and Fredslund [23] explore *formation control* with minimal communication and local sensing. Using the Pioneer AT DX similiar to the robots used in this work(see Section 3.1.1, they demonstrate formation control that is a predetermined geometric shape relative to one another. To accomplish this emergent behavior each robot is given an unique identification and a leader or conductor is assigned. The conductor's motion defines the heading for all other robots. Each robot communicates their heartbeats to everyone. The conductor communicates the current formation and its ID. Every other robot follows a partner robot called its *friend* maintain a distance and bearing to it. It does this by panning its camera to a known angle based on the position and maintaining its *friend* in the center of the image. Formations are then defined by the *chain of friendships*. Members are prohibited from communicating heading and velocities to members. In this work, they demonstrate a formation following algorithm that is very robust. Angles around 90

degree; however, proved troublesome. The classification of this work is COMM-INF,BAND-LOW,ARR-COMM,SIZE-LIM,COMP-HOMO,TOP-BROAD.

A group, from the Middle East Technical University has built *kobots* for use in their swarm robotic research. Shown in Figure 2.3, each kobot is 12 cm in diameter and weighs 350 grams. Its uses differential drive DC motors for locomotion. The basic short range IR sensing system provides both swarm member detection and obstacle avoidance through 8 senors equally distributed around the circular body. Kobots also possess wireless support through the IEEE 802.15.4/ZigBee protocol. This protocol provides a low-power networking capability that can support point-to-point, point-to-multipoint and peer-to-peer communication. This protocol, was preferred to IEEE 802.11 over its power efficiency, and to Bluetooth over its ability to address 65536 modules instead of 7 as supported by Bluetooth. Kobot's also are able to have a basic omnidirectional vision system comprised of a camera facing an omnidirectional mirror. This system provides range information up to 0.9m radius around the robot.

Figure 2.3: *Kobot.* A group of seven *kobots* designed by a group from the Middle East Technical University in Ankara, Turkey [3].

In [4], Ali Emre Turgut et al. explore the self-flocking behavior using *kobots*. Using a novel sensing system called the virtual heading system (VHS) which uses a digital compass and a wireless communication module for sensing the relative headings of neighboring robots they demonstrate the flocking behavior based on heading alignment and proximity control behaviors. Furthermore, they investigate the performance of their algorithm when subjected to variations in the VHS parameters. Their system is robust for noise perturbations of their VHS and increases with the swarm size. The classification of this work is COMM-INF,BAND-INF,ARR-DYN,SIZE-LIM,COMP-INDENT,TOP-ADD.

A group at the Massachusetts Institute of Technology has produced a swarm of over 100 iRobot SwarmBots like the one shown in Figure 2.4. The iRobot SwarmBot is a five inch cube with a 32-bit RISC ARM Thumb microprocessor. It possesses a wide array of sensors including bump sensors, light sensors, a camera, drive-wheel encoders, and the ISIS infrared location and communication system. Robots in close proximity (less than three feet) can communicate through ISIS. Additionally, through ISIS swarm members are able to determine the locations and headings of those around them. Each robot has an array of twelve IR emitters, grouped into four quadrants. Data can be transmitted from these quadrants independently or in any group. There are four receivers on each robot, which allow it to determine neighbor positions by comparing the signal strengths of one message that is received on two different receivers. Messages are passed through a gradient-based multi-hop messaging protocol. All of these features create a platform ideal for swarm robotics researchers.

Each robot is equipped with the Swarm Operating System (SwarmOS) which provides an API for swarm researchers writing applications for the robots. It controls low-level input and output, motor control, ISIS, power management, sensor drivers, and wireless software updates [36].

Finally, these robots are supported by HIVE and the Robot Ecology. This system takes much of the maintenance burden off the swarm researcher. The system charges all of the robots and provides a hands-free interface to the swarm for software development, debugging, and analysis. This greatly reduces the amount of management needed to maintain the swarm. Thus, allowing for rapid algorithm development and experimentation.

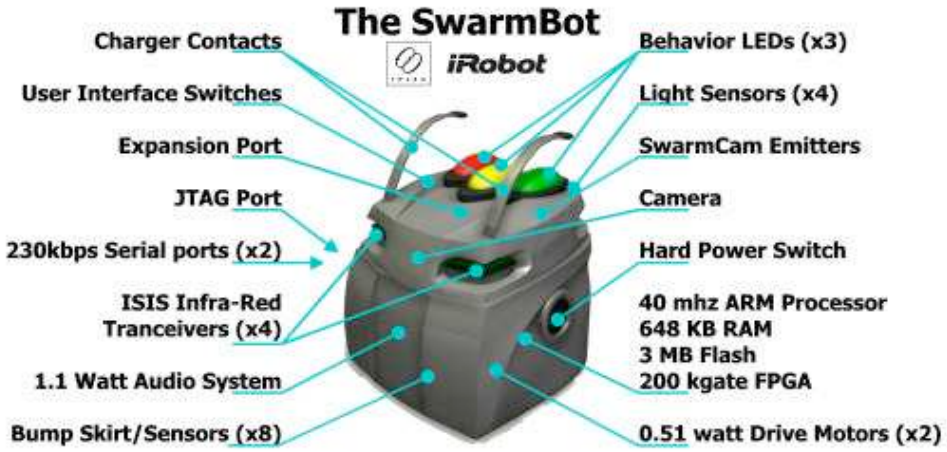

Figure 2.4: *iRobot SwarmBot* [36].

Using the SwarmBots, see Figure 2.5, McLurkin et al has produced algorithms and has explored the application of swarm disperson [40]. The goal of his dispersion algorithm is to disperse a large group of autonomous mobile robots efficiently throughout an indoor environment. This could be used to in any search and rescue, exploration, or map building application. McLurkin has also explored dynamic task assignment for large robotic swarms in [37, 38]. In [39] McLurkin explores the question of accuracy of communications within a swarm. He evaluates several multi-robot algorithms which rely on broadcast spanning trees against the speed of configuration changes in the swarm. He found the accuracy of these algorithms decrease as the robot speed, and the rate of topology change increases. This results confirms the interrelation of communications bandwidth, mobility, and algorithm accuracy. The classification of this work is COMM-NEAR,BAND-INF,ARR-DYN,SIZE-INF,COMP-INDENT, TOP-TREE.

Rothermich et al. [45], using the *iRobot Swarm*, devised an application for swarms in a distributed localization and mapping scenario. They constrain the swarm's movement to ensure no communication links are lost among any members. Robots uses local odometry readings to determine localization. As with single robot implementations this is often inaccurate and loses credibility with time. Rothermich proposes using swarm members as mobile beacons to improve localization. As the swarm moves *anchor robots* are formed at

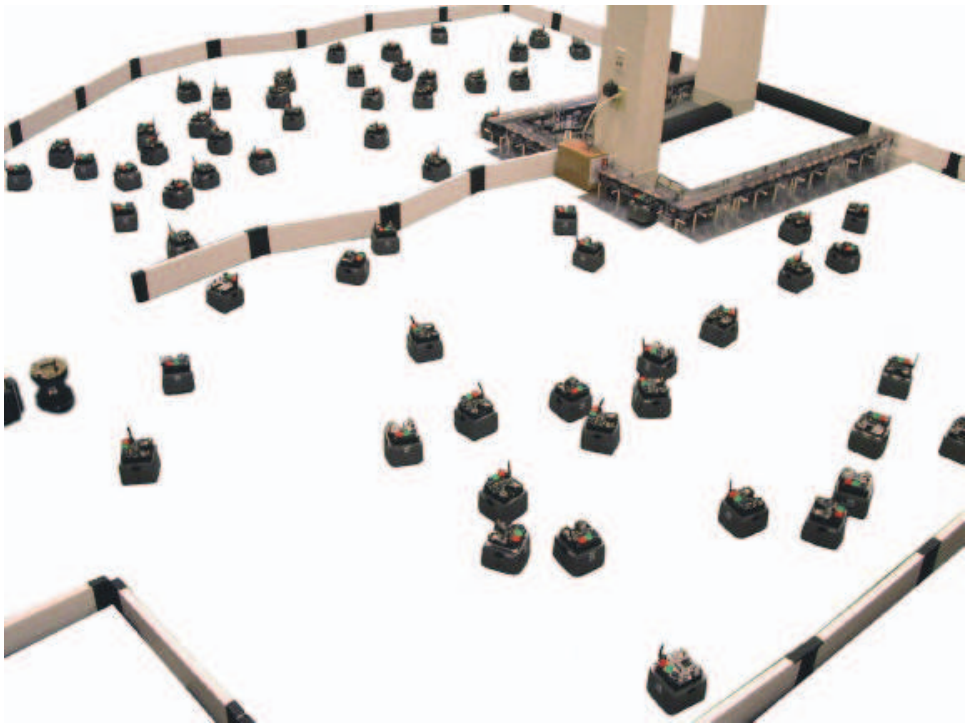

Figure 2.5: *Sample iRobot Swarm* - The iRobot Swarm consists of over 100 robots working together to accomplish common goals. In the upper right hand corner you can see the HIVE and Robot Ecology which allow the researcher to eliminate the maintenance tasks or recharging and allows for rapid code dispersion to the swarm [36].

the front on the swarm and releases *anchors* from the rear of the swarm. In this manner the swarm leap frogs its way through the environment while maintaining map accuracy.

This behavior emerges from three basic low-level controller decisions. Each controller is either, mapping, beacon or returning to mapping. Mapping means move with in the general swarm direction while maintaining a relative position (X,Y) based on local beacons. In beacon mode, the robot senses the number of beacons in the area. If that beacon number falls below a threshold the swarm member becomes a beacon broadcasting its position information to the swarm. The final behavior senses when the number of dependent swarm members goes below a threshold at which time the robot returns to the mapping behavior described above.

While moving through the environment each robot generates individual maps. Since each member is using the sample frame of reference, all maps are the same scale and orientation allowing for simple superimposing to reveal the overall swarm map. This method

has advantages over single robot mapping. First, since multiple viewpoints are used corners and edges are accurately captured. A second advantage is since the swarm maintains the same reference frame as it moves the localization problem is greatly reduced. The classification of this work is COMM-NEAR,BAND-INF,ARR-DYN,SIZE-INF,COMP-INDENT, TOP-TREE.

A group from the Universite Libre de Bruxelles [20] has built a large swarm project entitled, swarm-bots [49]. This research group has built a robotic swarm platform of small *S-bots*. Smaller than MIT's iRobots discussed earlier, these robots possess strong gripper with allow them to be physically linked.

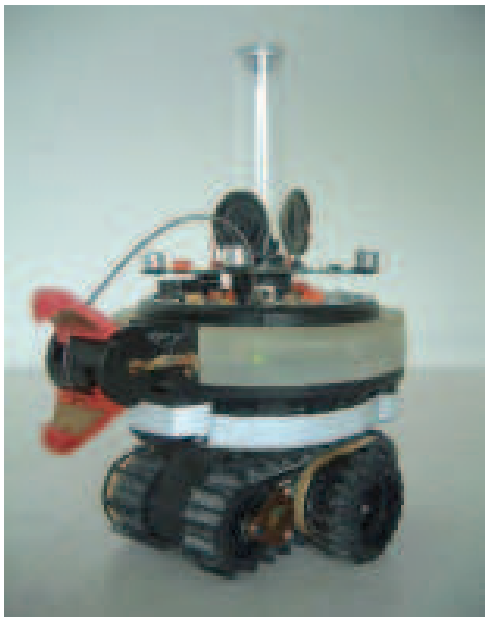

Figure 2.6: *s-Bot.* s-Bots are autonomous robots which can perform simple tasks [20].

Each *s-bot* (see Figure 2.6) is a fully autonomous robot capable of performing simple tasks including autonomous navigation, perception of the environment, and gripping objects. A differential drive system allows for efficient rotation and improves mobility. Each robot is also equipped with sensors and communication devices to detect and communicate with other *s-bots*, such as an omnidirectional camera, colored LEDs around the robots turret, and sound emitters and receivers. In addition to a large number of sensors for perceiving the environment, several sensors provide each s-bot with information about physical contacts, forces, and reactions at the interconnection joints with other s-bots. These include torque

sensors on most joints as well as traction sensors to measure the pulling/pushing forces exerted on the s-bots turret.

The ability to physically connect to other members is a unique feature of this system. When connected Dorigo refers to the collective as a forming a *swarm-bot*. An example is shown in Figure 2.7. A swarm-bot can perform tasks in which a single s-bot has major problems, such as exploration, navigation, and transportation of heavy objects on rough terrain.

Figure 2.7: *Swarm-Bot.* - A swarm-Bot working together to navigate rough terrain [20].

Using these *s-bots*, this group has produced a wide-variety of algorithms stressing emergent behavior rather than complex control algorithms. They have demonstrated successful swarms in applications of hole-avoidance [49], pattern formation [17], search and rescue [42], object transportation [24], and chain-based path formation [43]. The classification of this work is COMM-NEAR,BAND-INF,ARR-COMM,SIZE-LIM,COMP-INDENT, TOP-ADD.

This section presented an overview of physical swarm robotics. In the next section Kadrovach's simulation swarm model is described. Following this description, a basic overview of computer vision is given.

2.2 Kadrovach's Swarm Model

Kadrovach designed his novel swarm model primarily to provide a foundation for his research on optimal communication protocols for a swarm of UAVs. While this thesis investigation is not concerned with the structure of the swarm for communication purposes;

it is interested in the performance of this algorithm using an physical visual sensor on actual hardware. This subsection discusses the general algorithm including parameters, the vision sensor model, and the swarm entity models.

2.2.1 General Algorithm. The general algorithm is based on the foundations provide by Reynolds in [44] and Crombie in [16]. Essentially, the motion of swarm particles are defined by three main interaction factors or rules illustrated in Figure 2.8,

1. *Alignment*. Each agent's motion should match the average direction of its neighbors.
2. *Cohesion*. Each agent moves to the average location of its neighbors.
3. *Separation*. Each model should move to avoid collisions between members by maintaining a minimum separation distance.

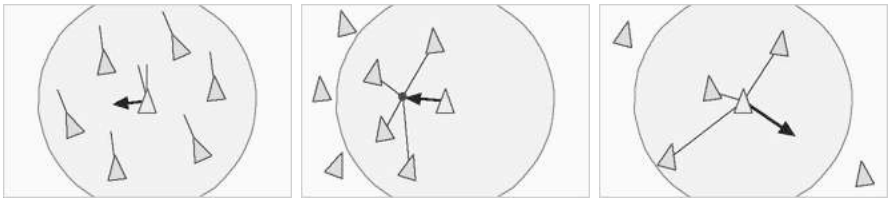

Figure 2.8: *Illustration of the basic swarm/flocking rules*. From left to right: alignment, cohesion, and separation [44].

These basics rules produce the emerging behavior akin to flocks of birds or schools of fish. The removal of one of these interactions destroys the emergent behavior [16].

The general algorithm for local motion control is given in Figure 2.9 and definitions of variables in Table 2.2. For each mobile particle, a new velocity vector is calculated based on the three rules described above and the basic interactions of the entities.

Table 2.2: *Swarm Algorithm Variable Definitions* [30].

Variable	Description
S	The set of all mobile entities
N	The number of swarm members, $\|\|S\|\|$
p_i	The i^{th} particle of S
N_i	The number of particles in the neighborhood of p_i

```
Loop ∀p_j ∈ S_i, j = 1 : N_i
   Process boundary objects
   Loop ∀p_j ∈ S_i, j = 1 : N_i
      Process neighbor p_j
      Calculate direction vector
   End Loop
   Move for a time step
End Loop
```

Figure 2.9: *General Motion Algorithm for Mobile Particle.* The descriptions of the variables can be found in Table 2.2 [30].

2.2.2 Vision Sensor Model. This section discusses the visual sensor model assumed by Kadrovach, its assumptions, and its effects. The adaptation of this model to the physical sensor on the Pioneer P2-AT8 robots is discussed in Section 3.2.1.

The purpose of Kadrovach's visual sensor is two-fold. First, it is used to define the entity's neighbors which ultimately define its motion. Secondly, it ensures the scalability of his control algorithm by limiting the size of the neighborhood thus producing an upper bound on the computational complexity of his algorithm. The key concepts of his visual model are *shadow*, *blocking*, and *visibility*. These concepts are defined in the following three paragraphs.

The shadow of a particle is the set of points which lie in a particle's visual field of view. Mathematically, the shadow of a particle is shown in Figure 2.10 and is defined by the following equations.

$$vis(\cos\theta_{ab}) = \begin{cases} true, & \cos\theta_{ab} < \cos\theta_{vis} \\ false, & \cos\theta_{ab} \geq \cos\theta_{vis} \end{cases} \tag{2.1}$$

$$\mathbf{v_a} = \mathbf{p_j} - \mathbf{p_i} \tag{2.2}$$

$$\mathbf{v_b} = \mathbf{p_k} - \mathbf{p_j} \tag{2.3}$$

$$\theta_{ab} = \cos^{-1}\left(\frac{\mathbf{v_a} \cdot \mathbf{v_b}}{\|\mathbf{v_a}\| \cdot \|\mathbf{v_b}\|.}\right) \tag{2.4}$$

A particle p_i is said to be *blocked* by particle p_j if it lies in the shadow (field of view) of p_j. A particle p_i is *visible* if it is not *blocked*.

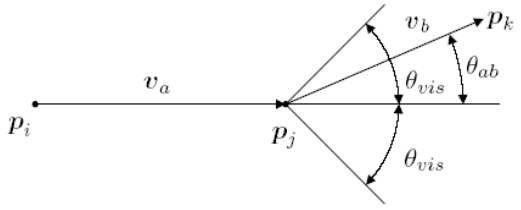

Figure 2.10: *Visibility Model* [30].

Figure 2.10 depicts a situation in which particle p_k is not visible by p_i because it lies in the field of view of the p_j and p_j is not blocked. This makes p_j a member of S_i while p_k is not a member. The net effect of this visual model is the only the nearest particles in the field of view determine the motion of the swarm particle. This behavior requires knowledge of other swarm members neighborhoods. Our limitation on communication makes this behavior impossible without a method for estimating the neighborhoods of other swarm members. This is ignored for the purposes of this thesis.

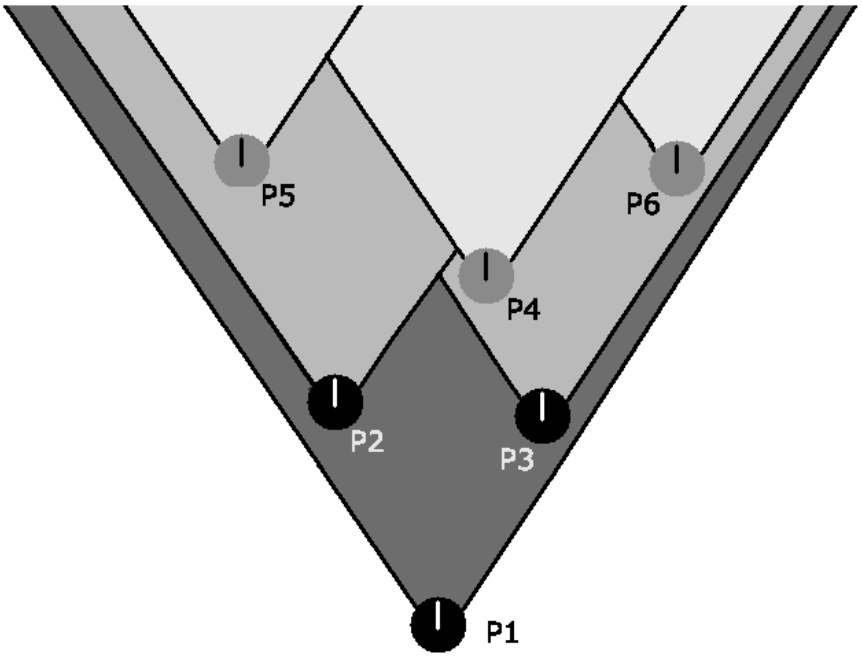

Figure 2.11: *Visibility Model Example.* The neighborhood of P_1 consists of P_2 and P_3. P_4 is no included because it is *blocked* by particle P_3. Similarly, particles P_5 and P_6 are *blocked* from P_1 although all three blocked particles lie within P_1 field of view, $+/-\theta_{vis}$.

Figure 2.11 illustrates further the net effect of the visual model. That is only the nearest swarm particles become elements of the neighborhood sets, S_i, which determine the overall motion of the swarm particle. In Figure 2.11, P_1's neighborhood S_{P_1} consists of only P_2 and P_3 because each of the remaining particles is *blocked* from P_1's view by either P_2 or P_3.

Kadrovach's motivation for selecting θ_{vis}, (for his work $\theta_{vis} = \frac{\pi}{3}$), is based on the ideal separation distance for his communication models and the goal of maintaining scalability of the swarm control algorithm by reducing the size neighboring set, that is $\|S_i\|$. Furthermore, he assumes each vision sensor has infinite range. That is a particle is visible without regard to separation distance. This assumption proves invalid when attempting a physical implementation of a vision sensor since all have a limited range of accuracy; however, it is safe to a assume a maximum distance, d_{max}, of influence for swarm particles.

2.2.2.1 Peripheral Vision Model. In effort to improve the fidelity of his simulation model Kadrovach modeled the notional vision sensor as one which weighted the estimates based on location in the field of view. Objects lying directly in front of the swarm particle have more influence on the motion of that particle than do ones to the periphery. This effect is captured by the w_{periph} weight given in equation 2.7. The formula for computing this weight is given by,

$$w_{periph}(\theta) = C_{periph} \cos^n(\frac{\theta}{2}), \qquad n = 1, 2, 3, \ldots. \tag{2.5}$$

The value of C_{periph} was chosen to be unity in Kadrovach's simulations. Figure 2.12 shows the graph of this function.

Kadrovach gives an alternative form of equation 2.5 through trigonometric manipulation it is given by,

$$w_{periph}(\theta) = C_{periph}[.5(1 + \cos\theta]^{n/2.} \tag{2.6}$$

2.2.3 Swarm Entity Model. This subsection discusses the swarm entities model derived and used by Kadrovach. Including his three main entities, their parameters and the

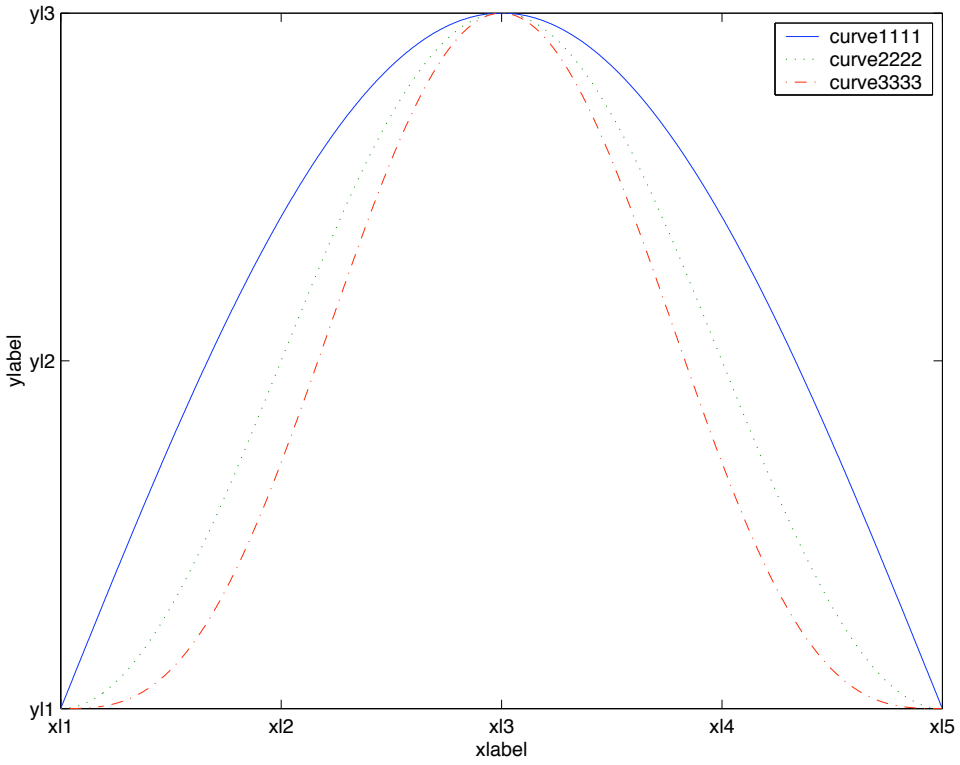

Figure 2.12: *Peripheral Weight Illustration* [30].

control equations. The overall motion of an entity is determined through the interaction of particles in a neighborhood of influence surrounding each mobile particle. The result is a distributed control algorithm for each member without explicit communication. The goal of Kadrovach's model is to approximate the flocking of birds with the three main control rules of *alignment*, *cohesion*, and *separation*. The biological process is a continuous phenomena with instantaneous feedback; however, the approximation model of Kadrovach is a discrete-time model which lends itself well to implementation in simulation.

Table 2.3: *Swarm Member Parameter Definitions* [30].

Parameter	Scope
Max Speed	Local
Turning Radius	Local
Separation Distance	Local
Neighborhood Size	Local
Swarm Size	Global
Environment Size	Global

2.2.3.1 Positional and Velocity Updating. All of the entities in Kadrovach's model are assumed to be points with negligible size with respect to the separation distance. The classes of entities can further decomposed into mobile (swarm member) and static particles (waypoints and boundaries). Swarm particles sense their environment through the notional vision sensor with peripheral vision modelings. This vision sensor allows the swarm particles to update their position based on the update equation given in equation 2.7. $(\mathbf{v_{update}})_\mathbf{i}$ gives the positional update for swarm particle i. The $\mathbf{v_{attract}}$ vector component of equation 2.7 encapsulates the *separation* and *cohesion* rules. The $\mathbf{v_{align}}$ vector encapsulates the alignment rule.

$$(\mathbf{v_{update}})_i = \sum_{p_j \in S_i} [(w_{periph})_{ij}(w_d)_{ij}((w_{attract})_{ij}(\mathbf{v_{attract}})_{ij} + C_{align}(\mathbf{v_{align}})_{ij})]. \tag{2.7}$$

The indices of equation 2.7 are to clearly identify the quantities are computed with respect to particles $\mathbf{p_i}$ and $\mathbf{p_j}$ where $\mathbf{p_j}$ is a neighbor of $\mathbf{p_i}$ and thus a member of S_i. w_{periph} is the weighting coefficient for the Kadrovach's peripheral vision model described in Section 2.2.2.1. Similarly, w_d is the distance weighting coefficient. The $\mathbf{v_{attract}}$ is computed by the following equation.

$$(\mathbf{v_{attract}})_\mathbf{ij} = \mathbf{p_j} - \mathbf{p_i}. \tag{2.8}$$

Where $\mathbf{p_j}$ is a neighbor of $\mathbf{p_i}$. Mathematically, $\mathbf{p_j} \in S_i$. The C_{align} quantity is a constant which determines the weighting of the alignment rule. The $(\mathbf{v_{align}})_\mathbf{ij}$ component is computed by the following equation.

$$(\mathbf{v_{align}})_\mathbf{ij} = (\mathbf{v_{dir}})_\mathbf{j}. \tag{2.9}$$

where $(\mathbf{v_{dir}})_\mathbf{j}$ is the heading of the particle p_j which is a neighbor of p_i. This is global knowledge in Kadrovach's simulation; however, for our physical implementation requires the vision sensor to estimate these headings.

The particle direction and position vectors are updated by the following two equations. Note the value of Δt is assumed to be unity for the purposes of simulation implementation.

$$\mathbf{v'_{dir}} = \frac{\mathbf{v_{dir}} + \mathbf{v_{update}}}{\|\mathbf{v_{dir}} + \mathbf{v_{update}}\|}. \tag{2.10}$$

$$\mathbf{p_i'} = \mathbf{p_i} + (\mathbf{v_{dir}} + \mathbf{v_{update}})\Delta t. \tag{2.11}$$

$\mathbf{v'_{dir}}$ and $\mathbf{p'_i}$ are the updated positions of the particle. To more accurately model the behavior of physical swarm members, Kadrovach bounds the positional and direction updates by bounding the maximum distance and heading changes in a time step.

2.2.3.2 Neighborhood Model. The neighborhood model defines the influence of other swarm members on the current member's motion. Figure 2.13 illustrates this model. The neighborhood model partitions the space surrounding a particle into four regions. The resulting effect is that the particle's effect on another particle is dependent on the region the particle resides in during the update cycle. The neighborhood models for each particle effect the the distance weight, w_d, and the attraction weight, $w_{attract}$, of equation 2.7.

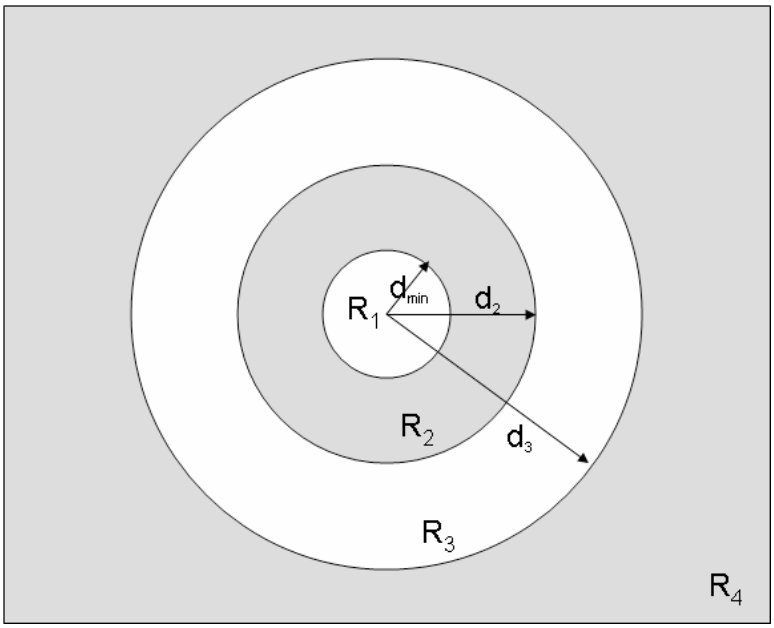

Figure 2.13: *Neighborhood Model* [30].

The first region represents the *close* region. The region is defined to be a measured separation distance, d, between zero and d_{min}. Kadrovach's value for d_{min} is unity.

The second region is the *comfort zone* or no effect region. This region is the ideal distance for dynamic particles to reside in Kadrovach's model. The value d_2 is specified by the parameter c_{zone}. As this value increases the region of no effect increases providing

a larger R_2. Kadrovach uses this parameter to allow more freedom in the velocity and directional controls of the swarm particles in this region.

The third region is the far region. Particles in this region have a scaled effect and tend to pull their neighbors towards the *comfort zone* defined above.

Finally the fourth region limits the effect on the neighbors if a particle is too far away. This region contains everything beyond d_3 and the next section shows that particles in this region have much less effect on a swarm members motion than the previous three regions.

This subsection introduced the neighborhood model. The neighborhood is subdivided into four distinct regions and the effect on the swarm members motions is defined by the regions it falls in. In the next section the attraction and repulsion weights in Kadrovach's model are discussed.

2.2.3.3 Attraction and Repulsion. Attraction and repulsion behaviors are necessary to ensure the swarm members do not collide and maintain their positions relative to each other. Kadrovach's neighborhood model defines the weights, w_d and $w_{attract}$ which control the attraction and repulsion of particles. In Kadrovach's swarm calculations for attraction and repulsion are made with the following equations.

$$w_d = \begin{cases} \sqrt{1-d}, & p_j \varepsilon R_1 (d < d_{min}) \\ 0, & p_j \varepsilon R_2 (d_{min} \leq d < d_2) \\ (\frac{d-d_2}{d3-d_2})^2, & p_j \varepsilon R_3 (d_2 \leq d < d_3) \\ e^{\frac{-(d-d_3)}{d_3}}, & p_j \varepsilon R_4 (d \geq d_3) \end{cases} \tag{2.12}$$

$$w_{attract} = \begin{cases} -C_{repulse}, & p_j \varepsilon R_1 (d < d_{min}) \\ 0, & p_j \varepsilon R_2 (d_{min} \leq d < d_2) \\ C_{attract}, & p_j \varepsilon R_3 (d_2 \leq d < d_3) \\ C_{attract}, & p_j \varepsilon R_4 (d \geq d_3). \end{cases} \tag{2.13}$$

The values of $C_{repulse}$ and $C_{attract}$ control the coherent behavior or incoherent behavior of the global swarm. Coherent behavior refers to a swarms tendency to move as a unit. Thus, incoherent swarm would tend to favor spreading out. Coherent and incoherent behavior is favored depending on the application of the swarm. For example, a coherent swarm is well suited for formation type activities. Incoherent swarms are better used in search type activities where spreading out through one's environment is desirable.

Recalling from equation 2.7 the total attraction weight is given by $w_d * w_{attract}$. Figure 2.14 shows this aggregate weighting coefficient. The neighborhood regions are labelled R_1 through R_4.

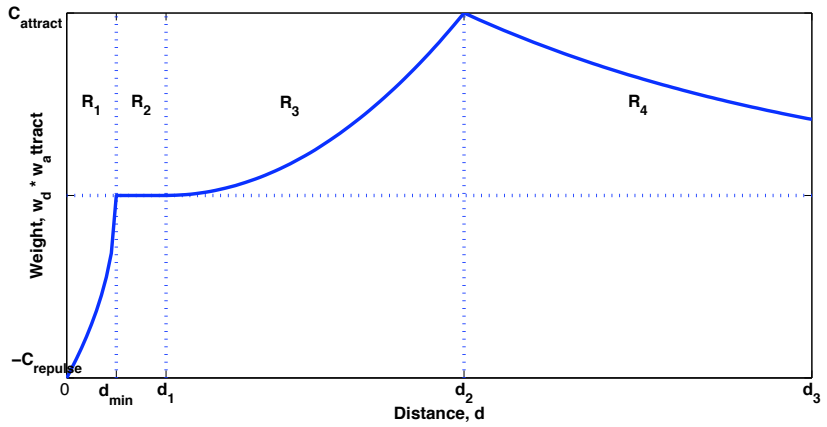

Figure 2.14: *Attraction Weight* [30].

2.3 Computer Vision

The goal of computer vision is to reverse the imaging process so we can reason about objects in the real world. The task is complicated by the fact that a given 2D image can arise from a number of 3D scenes. Every vision task begins with the acquisition of images from a camera. This process results in an unavoidable loss of information. Traditionally, the goal of vision, as far as robotics or artificial intelligence is concerned, is construction of a symbolic representation of the surrounding environment so that the entity can reason about the world. This task has proven to be very difficult for researchers to tackle. The shear volume of information is staggering [28].

Consider the information contained in a single black and white camera with a resolution of 512 by 512 pixels. Each of these pixels are eight bits encoding 256 intensity levels. If these images are processed at a normal frame rate of 30 Hz, the arrival data rate is in excess of 7 megabytes per second. The situation is three times worse when the images are color because of the three color bands.

With the wealth of information available in the context of an image, how does one extract the pertinent information? Moreover, the how does one use the information to construct a high level model of the world around you? Indeed these have proved to be difficult problems. Tanimoto identifies several barriers to computer vision to reason about the world in [48]:

1. The inherent loss of information from the transformation of the 3D world to a 2D image.
2. Each pixel's value is a product of many interacting phenomena which are difficult to decouple.
3. The shear size of the data in a single image frame is large.

While vision holds the most promise, computational complexity of vision algorithms limits their effectiveness in mobile robotics. Computational resources available on most robotic platforms are not yet capable of processing streams of images in real-time. In many cases algorithms reduce the size of images which reduces the quantity and accuracy of the information that can be extracted.

Using computer vision to solve problems in robotics has advantages over more traditional perception methods such as sonar, radar, infrared, and lasers. First, vision is inherently passive. It does not emit anything into the environment and is thus advantageous in applications requiring a level of stealth. Second, given proper optics vision has no range limitations unlike lasers, infrared and sonar. Thirdly, biological vision systems provide researchers goals and examples to mimic and learn. Finally, the information yield is larger from vision sensors than from sonar, radar, infrared and lasers.

This section provides the basics of computer vision as they apply to the application. Image formation, image segmentation, and stereo vision are stressed. The image formation

section provides a foundation of the overall process of computer vision. This provides a good solid foundation for understanding the algorithms in Chapter 3. Image segmentation is important in this application because a color segmentation algorithm provides the ability to locate other members of the swarm. Finally, the stereo vision section provides the background on the method for extracting distance and velocity estimates of the neighboring swarm members.

2.3.1 Image Formation. Whether it is our eyes or a computer's cameras, vision works by gathering light information from the objects in a 3-dimensional (3D) world, or *scene*, and creating a 2-dimensional (2D) representation called an *image*. An *image* is simply a 2D pattern of brightness. To properly study any image algorithm one must understand the factors which govern image formation. This section provides an overview of the process addressing two important issues. First, the geometric model of the correspondence of brightness patterns of an image with points in the scene. Second, the properties which govern the brightness of a point in a image are addressed. For further development see [28, 31, 46].

2.3.1.1 Geometric Model For Image Formation. Figure 2.15 depicts the pinhole camera model for image formation. The model is composed of an ideal pinhole at a constant observation distance,f, in front of an image plane. The perpendicular line from the image plane to the pinhole aperture defines the *optical axis*. Since light travels in straight lines each point in the image corresponds to a point in the scene by the line through the pinhole connecting the two points. Imagine a normal right-handed coordinate system with an origin at the pinhole with the positive z-axis in front of the camera.

Using this model, the relationship between an object point P_o, with coordinates (x_o, y_o, z_o) and the point on the image plane P_i, with coordinates (x_i, y_i, z_i) is derived. Using similar triangles the *projection equations* are discovered.

$$x_i = \frac{-fx_o}{z_o} \; and \; y_i = \frac{-fy_o}{z_o}. \tag{2.14}$$

The *Depth Range* of a scene is the range of distances of surfaces from the camera. Objects at different distances have different magnifications in the image. If the depth range of the scene is small compared to the average distances of the surfaces to the camera then

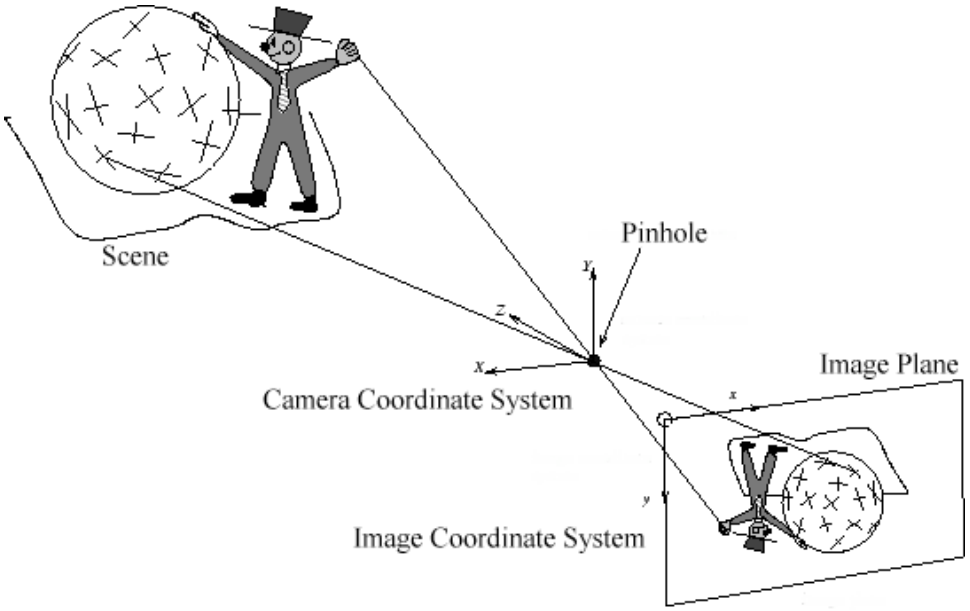

Figure 2.15: *Pinhole Camera Model Illustration* [35].

the projection equations simplify. That is if the depth, Z, of points on an object lie in a range $Z_0 \pm \triangle Z$ with $\triangle Z \ll Z_0$, then the scaling factor $\frac{f}{Z_o}$ in 2.14 with the constant $s = \frac{f}{Z_0}$ [46].

Some limitations of the pinhole camera should be discussed. First the pinhole model assumes an aperture of zero diameter. This is a purely theoretical construct as all physical apertures have some non-zero diameter. Secondly, the pinhole camera model assumes light through a small aperture travels only in straight lines and thus ignores wave nature of light namely, diffraction.

2.3.1.2 Lenses. Cameras make use of lenses. In humans, the eyes change their shape to focus on objects in the scene. Cameras, however, change the focal length of the camera to bring objects into focus. An ideal lens produces the same projection on the image plane as the pinhole camera; however, it captures only a finite quantity of light. Moreover, the entire scene is not in focus. An image of an object a distance Z_{object} in front of the lens is projected on a fixed distance Z_{pro}. The two are related by the lens equation:

$$\frac{1}{Z_{object}} + \frac{1}{Z_{pro}} = \frac{1}{f}. \tag{2.15}$$

where f is the focal length of the lens system. If the image plane is located at Z_{pro} then the object is in focus. If not the objects' points are imaged as blur circles instead of points. The *Depth of Field* is the range of distances which are focused such that their blur circles are less than the image resolution of the device. Figure 2.16 illustrates this process.

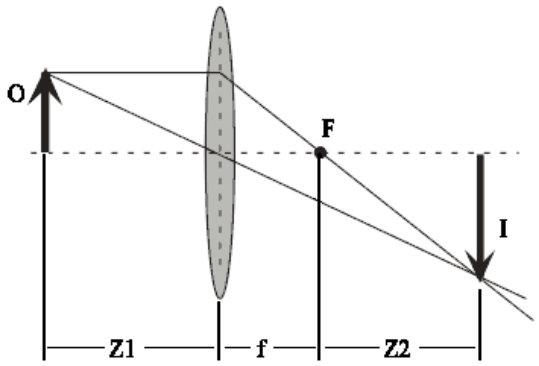

Figure 2.16: *Lens Illustration* [28].

2.3.1.3 Brightness Factors. The brightness is an ambiguous term to describe two distinct concepts: *image brightness* and *scene brightness* [28]. *Image brightness* or *irradiance* is defined as the power per unit area of falling on a surface, in our case the image plane. It is light received by a detector. *Scene Brightness* or *radiance* is defined as the power per unit foreshortened area emitted by a surface into a unit solid angle cone by a surface. It is light emitted from a source in a give direction. Surfaces emit light into a hemisphere of angles thus, in general, the radiance will vary with the observation position. The definition of radiance allows us to capture only the light emitted in a finite cone in the direction of observation.

2.3.1.4 Image Sensing. As discussed above cameras acquire a tremendous amount of information. This subsection discusses the basics of the imaging devices which capture the images. The most popular digital imaging device is the Charge-Coupled Device (CCD). A CCD is a matrix of photoelements connected to metal oxide semiconductor (MOS) capacitors. When light hits the matrix a number of the incidental photons are transformed into electrons. Electrons are stored in the MOS capacitors. An analog to digital (A/D) converter then quantifies a number of electrons into a value to be read into a computer.

2.3.2 Stereo Vision. Stereo Vision attempts to extract 3D information about a scene from using multiple 2D images taken from different known locations. Using the information from different viewpoints allows us to extract some depth clues about the observed scene. This section describes the basics of stereo vision. For more information see [22, 28, 31, 46]. For surveys of contemporary techniques see [8, 19, 29, 34, 47].

To interpret a pair of images obtained from two different viewpoints one must first recover the transformation between the two camera coordinate systems. There are two fundamental problems in stereo vision. First, one must establish a relationship between the pixels in the two images. This is called *correspondence*. After the correspondence has been established the next step is to use the disparity between the images to establish an estimation for the depth of the point. This is called *reconstruction*. We discuss each of these problems in the next two paragraphs.

2.3.2.1 Correspondence Problem. For a point P_1 in image I_2, determine its corresponding point P_2 in image I_2. The term *correspond* means that they are the images of the same physical point M. This is what is commonly known as the correspondence problem. Approaches to solving this problem are normally grouped into two categories: Correlation Based and Feature Based. Researchers have begun exploring stereo vision without correspondence [5, 15]; however, correspondence is used here to solve the reconstruction problem.

Correlation based or matching methods for establishing correspondence use correlation among brightness patterns in the local neighborhood of a pixel in the other image. Since, these methods use the intensity values directly they are sensitive to changes in the intensity values due to ambient lighting and perspective. Finally, the presence of occlusions in the scene present problems which lead to erroneous distance estimates.

Feature based techniques use features derived from the images rather than the image intensities directly. This makes them less susceptible to changes in image intensities. The features most commonly are edges, edge points, or edge segments. Additionally, feature based techniques are typically faster than correlation based methods because the comparisons tend to be simpler [8, 19].

2.3.2.2 Reconstruction Problem. After correspondence has been established, the reconstruction problem must be solved. Given two corresponding points, P_1 and P_2, the problem is to estimate the 3-D coordinates of the object point, P_{obj}. Suppose a basic two camera system with parallel optical axes and separated by a distance, b. The line connecting the connecting the two lens centers is the *baseline*. For simplicity, assume the baseline lies perpendicular to the optical axes and parallel to the image planes. Let the global camera coordinate system have origin at the midpoint of the baseline. Figure 2.17 depicts this situation.

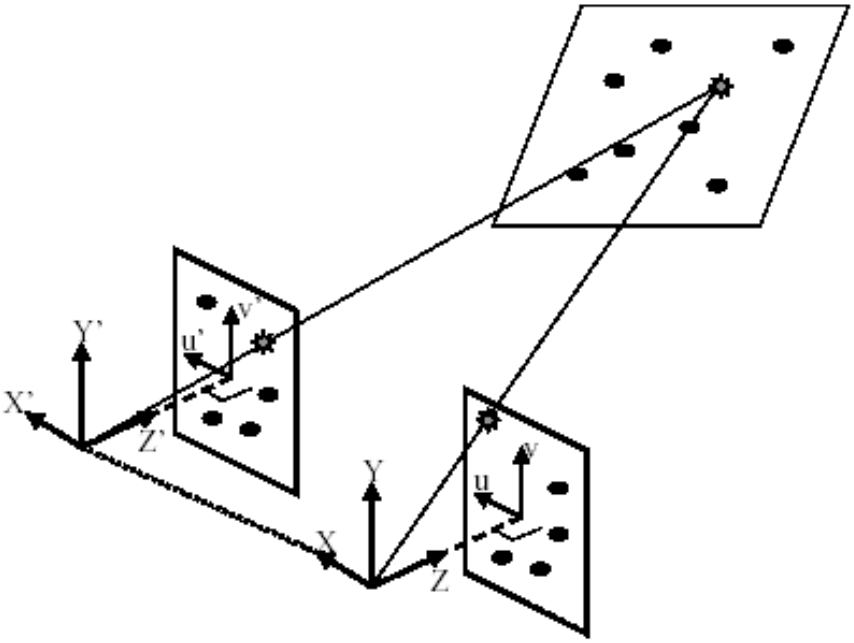

Figure 2.17: *Stereo Camera Model Illustration* [28].

It should be noted that a more general model exists for stereo vision allowing for cameras of different focal lengths and any orientation. In this more complex case one defines a so called *fundamental matrix* to handle transformations between the two cameras [19, 28]. However, since the camera in the implementation is of fixed geometry it is discussed here.

Let (x_1, y_1) correspond to the object point's $P_{obj}(x, y, z)$ projection onto image plane I_1, and (x_2, y_2) be the projection onto I_2. The relationships between the two points are given by,

$$\frac{x_1}{f} = \frac{x + \frac{b}{2}}{z} \; and \; \frac{x_2}{f} = \frac{x - \frac{b}{2}}{z}. \tag{2.16}$$

while

$$\frac{y_1}{f} = \frac{y_2}{f} = \frac{y}{z}. \tag{2.17}$$

where f is the focal length of the two cameras and (x, y, z) are the unknown coordinates of P_{obj}. To solve for these coordinates note,

$$\frac{x_1 - x_2}{f} = \frac{b}{z}. \tag{2.18}$$

The difference in the image coordinates $(x_1 - x_2)$ is known as the *disparity*, d.

Finally, solve for the object points,

$$\begin{aligned} x &= b\frac{\frac{x_1+x_2}{2}}{x_1 - x_2} \\ &= \frac{b(x_1 + x_2)}{2d} \end{aligned} \tag{2.19}$$

$$\begin{aligned} y &= b\frac{\frac{y_1+y_2}{2}}{x_1 - x_2} \\ &= \frac{b(y_1 + y_2)}{2d} \end{aligned} \tag{2.20}$$

$$\begin{aligned} z &= \frac{bf}{x_1 - x_2} \\ &= \frac{bf}{d}. \end{aligned} \tag{2.21}$$

Distance is inversely proportional to disparity. The accuracy of distance measurements is therefore better for near objects than far ones. Disparity is also directly proportional to the baseline distance, b. Thus, the accuracy of distance measurements increases with an

increased baseline. This effect is bounded however, because as one increases the baseline of the cameras less and less of the scene overlaps. That is the disparity field of view decreases. Finally, disparity is also directly proportional to f. This is because images are magnified with increased focal length.

Another important aspect of reconstruction is *epipolar geometry*. The epipolar geometry is a function of the relative orientation of the two cameras. A point in the scene which is visible in both cameras gives rise to a pair of corresponding image points called a *conjugate pair*. The conjugate pairs must lie on a line between the two images because they have the same y-coordinate by convention. This line is known as the *epipolar line*. For this particular geometry all epipolar lines are parallel to the x-axis; however, this is not true in the general camera model. The important thing to note is that if a point in the right image has a conjugate pair it must lie on the epipolar line; it cannot lie elsewhere [46].

Stereo algorithms do not in general search the entire image for a correspondence match as described above due to the computational complexity of such a process. Instead, they search a small local search of around a pixel. The result of this process is that the disparity values are bounded to a three dimensional space known as the *horopter*. Figure 2.18 depicts this situation. The correspondence match must lie in the horopter or it will not be found.

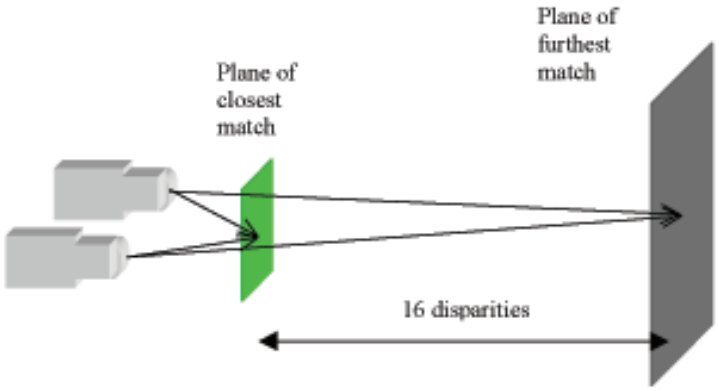

Figure 2.18: *16-pixel Horopter Illustration.* The nearest plane corresponds to the highest disparity value(15) while the farthest plane corresponds to the lowest disparity value (0) [33].

The placement and size of the horopter depends on the application. Typically, the search space is limited to 16 or 32 pixel disparities for computational reasons. For this

thesis' implementation the range will be 32 pixels. The horopter size can be increased by the following methods,

1. Decrease the camera baseline.
2. Decrease camera focal length.
3. Increase pixel width
4. Increasing the disparity bounds.

The first three of these options deal with changing the camera geometry itself which has a effect of the range resolution of the algorithm. Only, changing the disparity search range increases the horopter size without an adverse effect of range resolution.

Range resolutions refers to the minimum change in range that the stereo algorithm can differentiate. This is different from range accuracy which is the measure of how well the range estimates correspond to the actual ranges. Range resolution is governed by the following function,

$$\Delta r = r^2/bf \cdot \Delta d. \tag{2.22}$$

where Δr is the range resolution, r is the range, b is the camera baseline, f is the focal length, and d is the disparity. This equation show that range resolution degrades quadratically with increasing range. Camera geometry (focal length and camera baseline) also have an inverse effect. That is increasing either results in better (smaller) image resolution. Finally, the pixel size of the imaging devices has a proportional influence on the range resolution. Smaller pixel sizes yield better resolution.

2.4 Summary

This chapter presents background information necessary to establish the foundation for the work herein. The first section discusses the area of swarm robotics, its foundations, related research in the field and culminated in a detailed description of the robotic swarm control algorithm used in this work. The second section outlines the foundations of computer vision as they apply to this thesis investigation stressing image formation, image segmentation, and stereo vision. Having discussed this background information the next Chapter outlines the implementation of a physical swarm model.

III. System Design and Implementation

The development of any complex software system requires a structured, well-defined development plan. The robotic software system of this effort is no exception. A bottom-up behavior based approach [6] is used in the system beginning with the camera controller. The tight coupling of sensory inputs to motor outputs is the defining characteristic of this design paradigm which lends itself to the design and implementation of highly reactive systems.

This chapter covers the hardware and software components of the swarm model. Section 3.1 discusses the hardware components in the system. The details of the software components of the robotic system are presented in Section 3.2.

3.1 Hardware

This section discusses the hardware systems and robotic platforms used in this research. Section 3.1.1 the ActivMedia Pioneer 2^{TM}-AT8 robotic platform is discussed. Next, the Videre stereo camera is discussed. Figure 3.1 depicts the hardware and the interconnections between the hardware for a single implemented swarm controller. The swarm controller is implemented in two algorithms as described in Chapter 2.

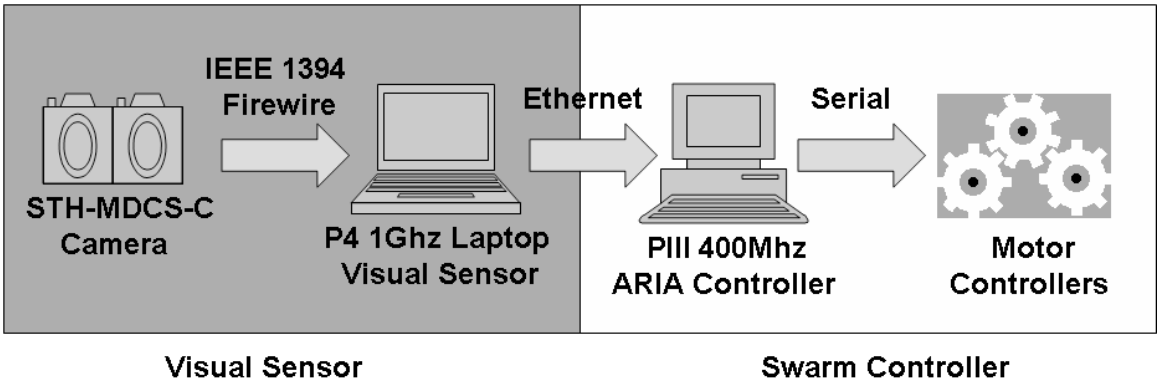

Figure 3.1: *Swarm Controller High Level Hardware with interconnections.*

3.1.1 Pioneer Robotic System. The ActivMedia Robotics Pioneer P2-AT8 is the platform used to conduct this research. The Pioneer P2-AT8 has an onboard 400Mhz Pentium processor and 64MB of onboard RAM. This onboard computing power proved to be insufficient for doing onboard segmentation and disparity processing so for this thesis the

onboard processor handles only the updates to the motor controllers defined by the swarm behaviors. The onboard behaviors take position information from the video processing laptop and use it to update heading and velocity of the robot.

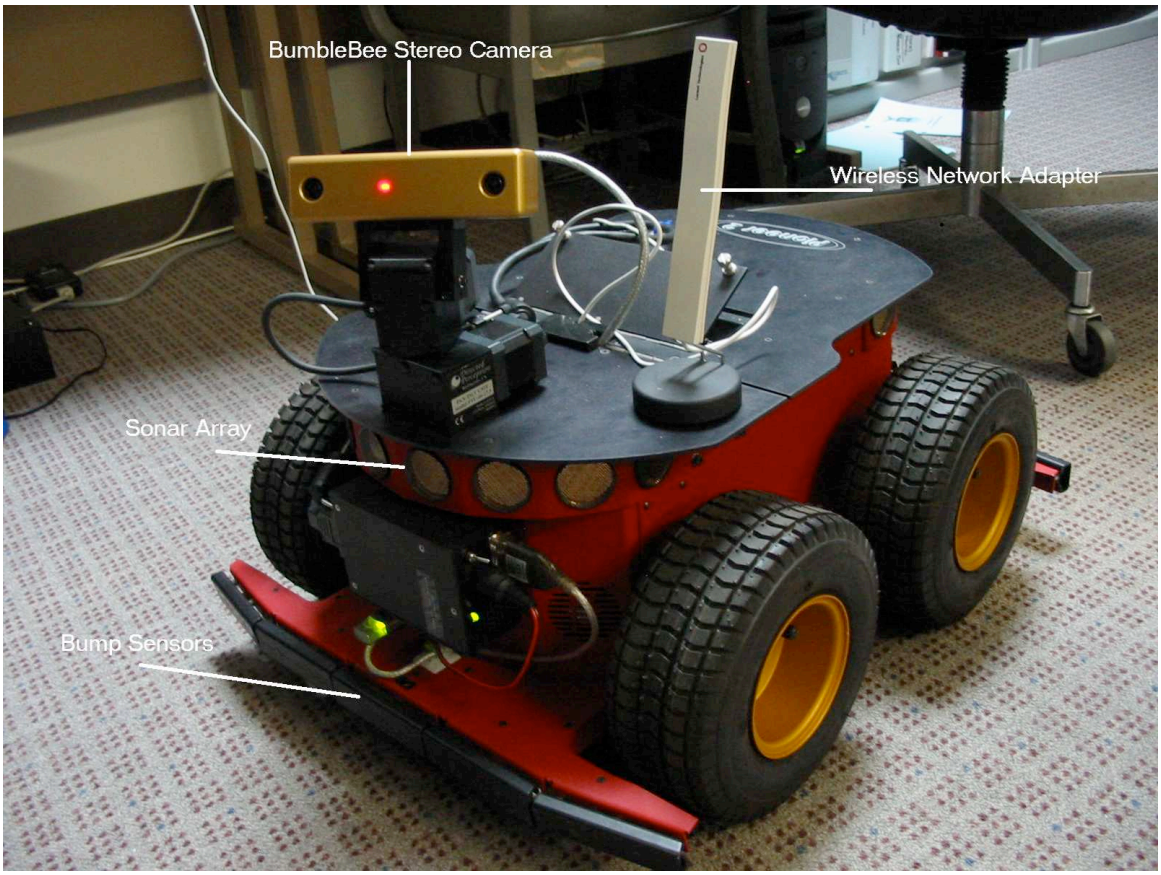

Figure 3.2: The Active Media Robotics Pioneer 2^{TM}-AT8.

3.1.2 Videre Stereo Camera System. For image capture and stereo processing, the STH-MDCS Stereo Camera System by Videre Design is used. As can be seen in Figure 3.2 the camera sits a atop a pan-tilt head; however, for this implementation the pan-tilt head is in a fixed position looking forward because of the assumptions of Kadrovach's visual sensor model. The complexity of the stereo vision processing lead to implementing the vision sensor on a 1Ghz Pentium P4 laptop which connects to the Pioneer robot via a standard ethernet connection. The laptop's specifications are listed in Table 3.1. The stereo head contains two CMOS imagers with a resolution of 1280x1024 pixels with a maximum frame rate of 12 megapixels per second. The camera interfaces with a laptop computer over a

Table 3.1: Summary of the Computational Specifications of the Vision Laptop.

Spec	Description
Processor Clock	1GHz
Memory	512 MB
Ethernet Speed	100 Mbs

IEEE 1394 "firewire" serial bus. This bus is capable of transferring the information over the bus without the need for buffering on the PC side. The lenses have a focal length of 6mm which gives the imagers a field of view of 65.2 degrees in the horizontal and 51.3 degrees in the vertical. The largest image size supported by the camera is 1280x960; however, for this thesis the application is a real-time implementation. Because a reduction in the image size results in faster run time performance of all algorithms, the image size is reduced from 1280x960 to 320x240 by "binning." Binning averages adjacent pixels into one pixel. The *'x4' binning* technique averages every four pixels is into a single pixel. The result of this process is an image of H/4 x V/4 where H is the original width of the image in pixels and V is the original height of the image in pixels. A summary of the STH-MDCS stereo camera specifications is given in Table 3.2.

Table 3.2: Summary of the Specifications for the STH-MDCS Stereo Vision Camera.

Spec	Description
Size	1.5X5X1in
Focal Length	6mm
Baseline	9mm
Horizontal Field of View	65.2°
Vertical Field of View	51.3°
Max Image Resolution	1280X960
Frame Rate at Max Res	7.5Hz
Implementation Resolution	320x240
Implementation Frame Rate	30Hz

3.2 Software

This section discusses the software components of the system. First the general algorithm of the visual sensor implementation is presented. This is followed by the segmentation algorithm used to identify swarm members. Finally, the method of determining estimated distances is outlined. Then, the low-level swarm behaviors are described.

3.2.1 Visual Sensor Model Implementation. The physical implementation of the notional visual sensor model of outlined in Section 2.2.2 begins by recalling its requirements. The visual sensor is required to provide positional and velocity estimates for the neighbors of the swarm member and pass this information to the controller for velocity updating. To do that requires a method to identify the other swarm members, to estimate their velocities, and to return this information to the controller for motion and positional updating. The vision sensor is implemented through the use of several classes which are discussed beginning with fast color segmentation which provides a means to locate the swarm members in the camera image. The fast color segmentation bounds the search space for the stereo processing algorithm. Using the SRI smallv Stereo Vision API [33] classes the visual sensor calculates an average estimated distance for the segmented area. This information is passed to the ARIA robot via a TCP/IP connection for motor control input. Figure 3.3 shows the pseudocode for this process.

```
Capture Color Stereo Images
Segment the Color Image to ID Orange cone
Build Regions Data Structure (Areas of Orange)
For Each Region
   Compute Disparity Image
   Estimate Distance from Disparity
   End Loop
   Pass Distance Estimate and Neighbors to ARIA Robot
```

Figure 3.3: *High Level Algorithm for the Visual Sensor.*

3.2.2 Fast Color Segmentation. The segmentation algorithm used in this research is an implementation of the Balch et al. fast color segmentation algorithm described in [12]. The main benefit of using this algorithm is its speed. Balch has tested the system on robots obtaining close to real-time performance (30 frames per second) on nominal computational hardware [12]. The overall algorithm is depicted in Figure 3.4. Only monocular vision, using the left camera, is used in segmentation.

The basic approach of this segmentation algorithm is to segment the image by defining up to 32 colors using thresholds in three dimensional color space. These thresholds define

```
For all pixels
   Project the Color Space
   Classify each pixel as one of up to 32 colors
   Run Length Encode the Classified Image
   Group runs into regions of like color
   Collect Regions Statistics
   Sort by Color and Size
```

Figure 3.4: *Algorithm for Fast Color Segmentation [12].*

volumes of the specified color in the color space. Three color spaces are popular in computer vision namely, Hue Saturation Intensity (HSI), YUV, and Red Green Blue (RGB). The selection of color space depends on the utility of the color space for the particular application and the color space provided by the vision hardware. Since the Videre imager produces (RGB) images, the RGB color space is used in this work.

The RGB color space is popular in many image processing fields; however, Balch et al [12] notes that for robust segmentation in robotic vision applications the RGB color space suffers from a major disadvantage. Ideally, robust performance of the segmentation algorithm in a wide range of lighting conditions is desirable. This leads to defining colors as ratios of the intensities of red, green, and blue bands in the color scheme. While this can be done in the RGB color space quite easily, the resulting volume in the color space is conical and thus cannot be represented by simple thresholds in each of the color bands [12]. The HSI and YUV color spaces encoded color information in two bands and intensity in the third. The relationships between the three are given by the following equations,

$$
\begin{aligned}
Y &= \frac{R+G+B}{3} && (3.1)\\
U &= R - \frac{G+B}{2} && (3.2)\\
V &= \frac{(B-G)\sqrt{3}}{2} && (3.3)\\
I &= \frac{R+G+B}{3} && (3.4)\\
S &= \sqrt{U^2+V^2} && (3.5)\\
H &= \begin{cases} Arc\cos(\frac{V}{S}), Y \geq 0 \\ 2\pi - Arc\cos(\frac{V}{S}), Y < 0. \end{cases} && (3.6)
\end{aligned}
$$

To mitigate this problem, the algorithm begins by modifying the RGB color space by subtracting the monochromatic intensity and storing the absolute difference back into the color bands. This transformation allows for a better color approximation by a defining rectangular region in the color space [12]. This decision sacrifices some computation time in order to improve the robustness of the segmentation algorithm in variable lighting conditions.

```
if((R_ij >= R_LOW) AND
   (R_ij <= R_HIGH) AND
   (G_ij >= G_LOW) AND
   (G_ij <= G_HIGH) AND
   (B_ij >= B_LOW) AND
   (B_ij <= B_HIGH))
     pcolor_ij = color;
```

Figure 3.5: *Naive Approach to Thresholding.* R_{ij}, G_{ij},and B_{ij} represent the pixel color values in each of the Red, Green, and Blue color bands respectively for the pixel located in row i and column j. R_{LOW}, G_{LOW}, and B_{LOW} are the low thresholds. R_{HIGH}, G_{HIGH}, and B_{HIGH} are the high thresholds [12].

As stated the segmentation algorithm relies on simple thresholds to define the colors for segmentation. The naive approach to define colors in this manner would use an if-then construct to test for a color. This approach is illustrated in Figure 3.5. After compiling, this approach proves inefficient due to each pixel requiring up to six conditionals to be evaluated

before the pixel's classification is determined. Furthermore, each additional color requires six additional conditionals.

Balch et al., instead, decompose the multidimensional thresholds into three boolean valued functions along each axis of the color space. The decomposition is stored in three static arrays one for each color component. Thus class memberships are determined using a bitwise AND of the elements in each color band. The result of the operation is a boolean with a *true* value representing the pixel is a member of the color class and $false$ value if it is not a member of the color class. The resulting algorithm is much faster on modern processors.

The process becomes clearer when considering an example. Consider a 10 level discrimination of the each of the 256 value color values. Assume that the color of interest is defined by the following ranges in each color band (6-2,7-3,0-1). The class membership arrays for this case would be,

$$RedClass = \{0,0,1,1,1,1,1,0,0,0\} \tag{3.7}$$

$$GreenClass = \{0,0,0,1,1,1,1,1,0,0\} \tag{3.8}$$

$$BlueClass = \{1,1,0,0,0,0,0,0,0,0\}. \tag{3.9}$$

Suppose now that a pixel has color values of (6,4,1). Determining, this pixel's membership in the defined pixel class requires the evaluation of the following.

$$pixelclass(6,4,1) = RedClass[6]\ AND\ GreenClass[4]\ AND\ BlueClass[1]. \tag{3.10}$$

For this example the result would be *true* indicating this pixel's membership in the notional class.

The main advantage of this approach, although not used in this investigation, is simultaneous pixel membership. To handle multiple colors the algorithm uses each of the n bits of an integer of the membership arrays to denote membership in n different classes. For example, to add a second color with the following ranges in the membership arrays

(2-1,5-3,8-9) requires the modification of membership arrays.

$$RedClass = \{00, 10, 11, 01, 01, 01, 01, 00, 00, 00\} \tag{3.11}$$

$$GreenClass = \{00, 00, 00, 11, 11, 11, 01, 01, 00, 00\} \tag{3.12}$$

$$BlueClass = \{01, 01, 00, 00, 00, 00, 00, 00, 10, 10\}. \tag{3.13}$$

Recall the example pixel has the color values of (6,4,1). The evaluation of the pixelclass function yields '01'. The most significant bit denotes that the pixel is not the second color but is the first color. This method can be extended to n bits where n is determined by the computational architecture.

After each of the pixels in the image has been classified into one of the pixel classes, the images are grouped into connected regions under the four-connectedness heuristic. This is accomplished in two stages. First, the image is run length encoded Red Green Blue (RLE). This accomplishes merging under horizontal connectedness. Then, the second stage must determine only vertical connectedness. This is accomplished in place in the RLE image. This is done by having each of the runs have a pointer to a parent run in the image. That is each run stores the length of the run, the run pixel, and pointer to the parent run. The parent is the upper-left most run of a region. The merging procedure is a tree-based union find with path compression [12]. It scans each pair of adjacent rows and merges runs which are the same color class and overlap vertically. The result of this is a disjoint forest with each run pointing upward in the image towards the global parent of the region. A second path compresses the paths so that all runs point directly to the global parent. Figure 3.6 illustrates this procedure.

The next step in the segmentation is the region statistic extraction. In this stage, the algorithm constructs a region table which includes a bounding box, centroid, and size of each region. This requires a single pass over the run length encoded image as each of the statistics can be calculated incrementally. After the statistics have been calculated the regions are divided into separate color regions sorted by decreasing size.

For this effort, the algorithm needs to recognize the color "orange." An orange cone marks the location of a swarm member. The first step of this implementation is to define

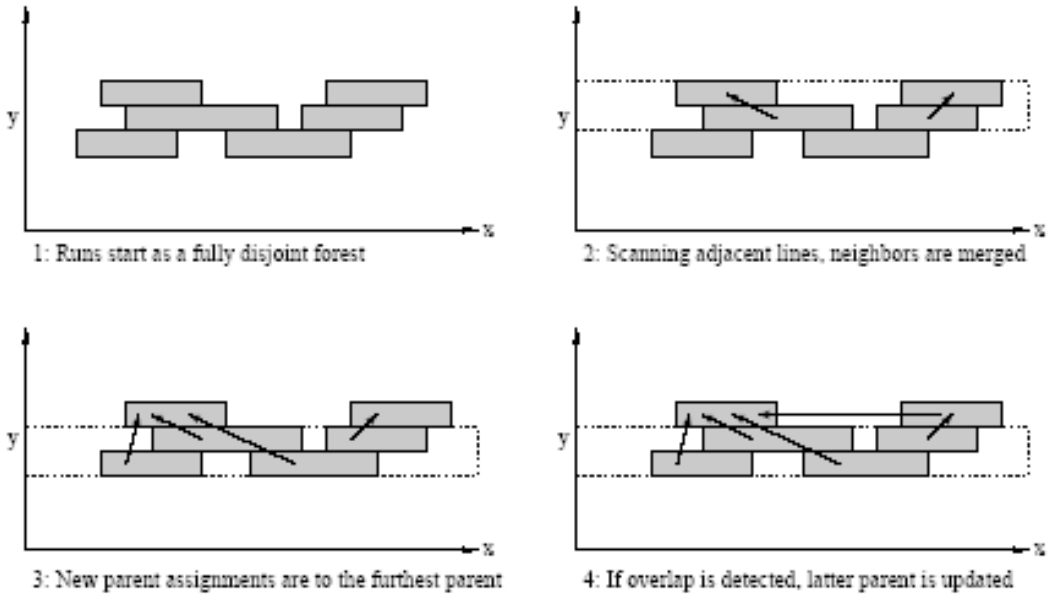

Figure 3.6: *Example of Union Find Path Compression* [12].

the color "orange" in each of the color bands. First, images are taken of the orange cones in two different lighting conditions, one indoor and one outdoor. After the subtracting the intensity information as described above the images are loaded into a graphics editor (MS Paint) and samples of the images are taken in different locations.. Ten samples from each image (20 samples in total) are loaded into Microsoft Excel and the mean (μ) and standard deviation (σ) of the samples are calculated. The summary results are listed in Table 3.3. The color orange is defined as $\mu + / - \sigma$. Figure 3.7 shows a sample pair of images with the fast segmentation.

Table 3.3: Summary of Orange Sampling Data.

metric	Red	Green	Blue
μ	112.10	20.80	72.40
σ	24.58	7.93	14.17
$\mu + \sigma$	136.68	27.73	86.66
$\mu - \sigma$	87.52	12.86	58.13

After segmenting the image the algorithm calculates a bounding box of the segmented image to pass to the stereo vision algorithm to estimate the distance to the target.

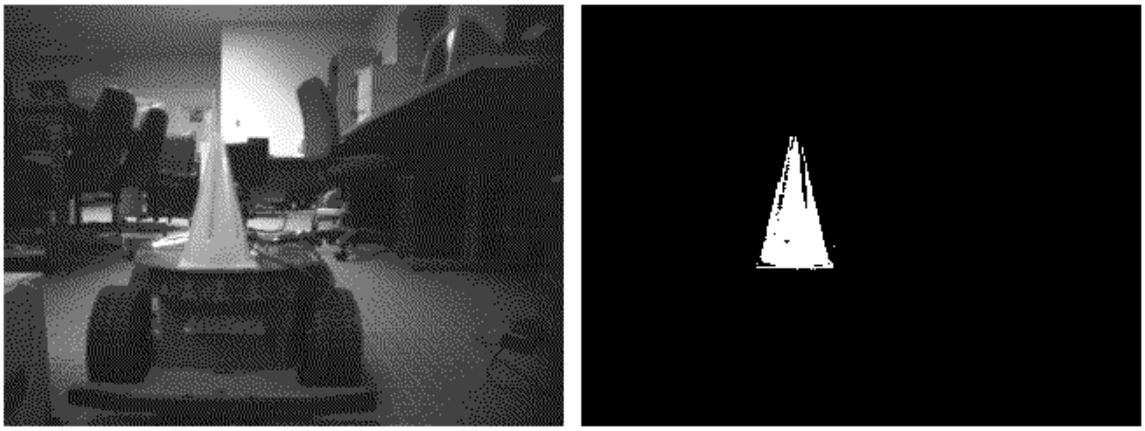

Figure 3.7: *Sample Segmentation Pair.* The image on the right shows the left color image obtained from the Videre camera. The image on the right shows the resulting color segmented "orange" image. This segmented region is the primary method of identification of swarm members.

3.2.3 Recovering Distance Measurements with Stereo Vision. The SRI Small Vision system [33] provides C++ classes and API for calculating distance estimates of objects with the stereo cameras. Recall from the background discussion Stereo algorithms compute range information using triangulation. Two images at different viewpoints see the object at slightly different positions: this difference in location of in the two imagers is called *disparity*. Furthermore, the range they can determine is also restricted by the disparity search range or horopter. For our process the horopter was set to 32 for maximum information. A sample image with disparity and point cloud is show in Figure 3.8.

As stated in the previous section the fast color segmentation algorithm passes a bounding box to the stereo vision algorithm. Using this information the smallv classes allow us to estimate the distance the object is by averaging the distances returned by querying the stereo processor over all points in the bounding box. Points without disparity information are ignored in this process. This information is stored in the neighbors data structure and passed to the motor controllers described in the next section to calculate the velocity and heading update for the swarm member.

3.2.4 Behavior-Based Swarm Robotic Control Implementation. This subsection outlines the behavior-based robotics controller implementation on the Pioneer robot. It begins with general discussion of behavior based robotics. A high-level design model is then

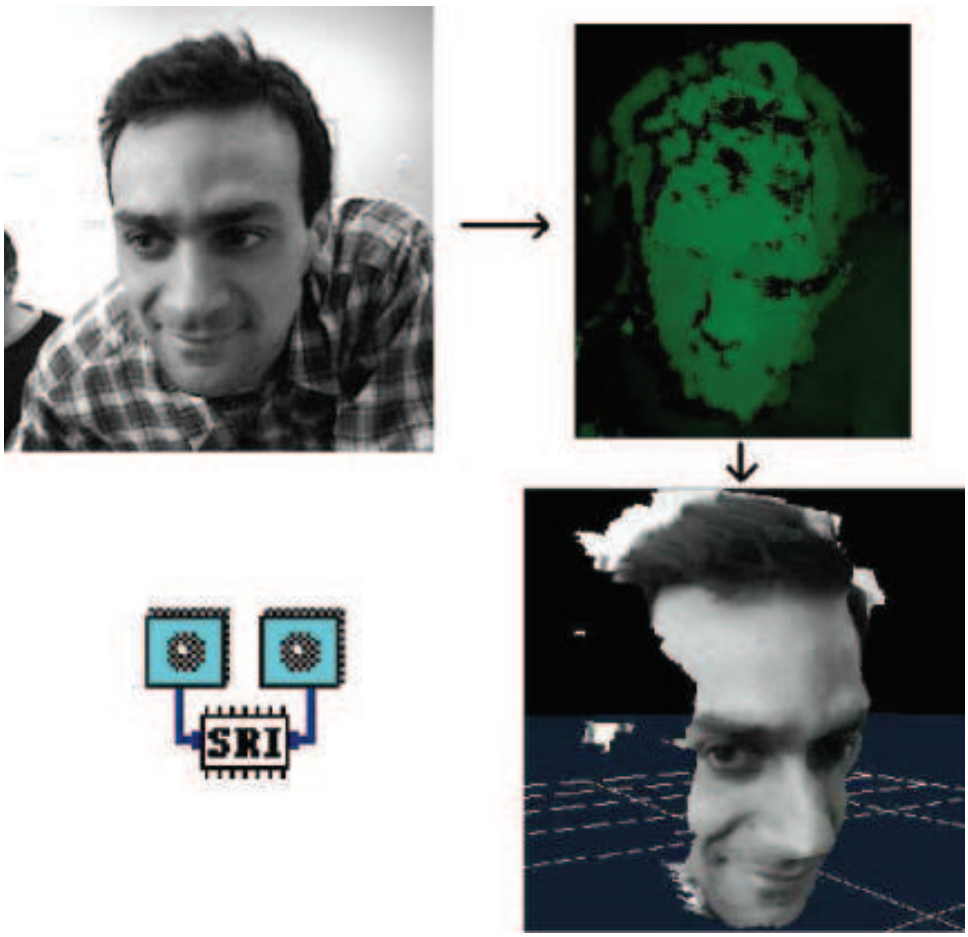

Figure 3.8: *Sample Small Vision System Stereo Processing Class Output* [33].

presented in which the Kadrovach swarm model is decomposed into two behaviors which are implemented on the robot. Finally, a detailed discussion of the separation and alignment behaviors concludes the subsection.

3.2.5 Introduction to Behavior Based Robotics. Championed initially by Rodney Brooks, the behavior based robotics field focuses on design robots whose macro behavior is an aggregation of several, often simple, behaviors [11]. These systems tend to be very reactive in nature and contain very little state information. Furthermore, behavior based approaches rarely have complex models to define objects in their environment for the purpose of reasoning. Rather these systems take the robots sensor information and perform quick almost reflex reactions to their environment. Interested readers are referred to [6, 10, 11] for more development.

3.2.6 High-Level Design. The system for implementing this swarm controller begins with a high-level model. The swarm controller is decomposed into two behaviors separation and alignment behaviors. These behaviors are implemented in the ARIA robot control architecture [32]. Figure 3.9 shows a high-level design diagram. Figure 3.9 A shows the overall design paradigm. While Figure 3.9 B shows the two behaviors necessary to implement the swarm model described by Kadrovach. The stimuli are provide by the vision sensor described above and the responses are given sent to the ARIA controller for execution. The behaviors *alignment* and *separation* are implemented in ARIA as Actions derived from the *ArAction* class provided in ARIA. They are then combined using vector addition to produce the overall action by the two behaviors. In the instantiation of the ARIA control environment requires that the priorities of the two behaviors be specified. Since there is no preference given in the simulation model the behaviors are given equal priorities of 100.

3.2.7 Separation/Cohesion Behavior Design. The separation and cohesion behavior requires two basic functions. First, the information passed to the ARIA architecture is read in via a TCP/IP connection. The visual sensor provides information on the number of neighbors and an estimate of the neighbors location and heading. Using this information the separation/cohesion behavior determines the region each neighbor lays in and computes the update vector as defined below in Figure 3.10.

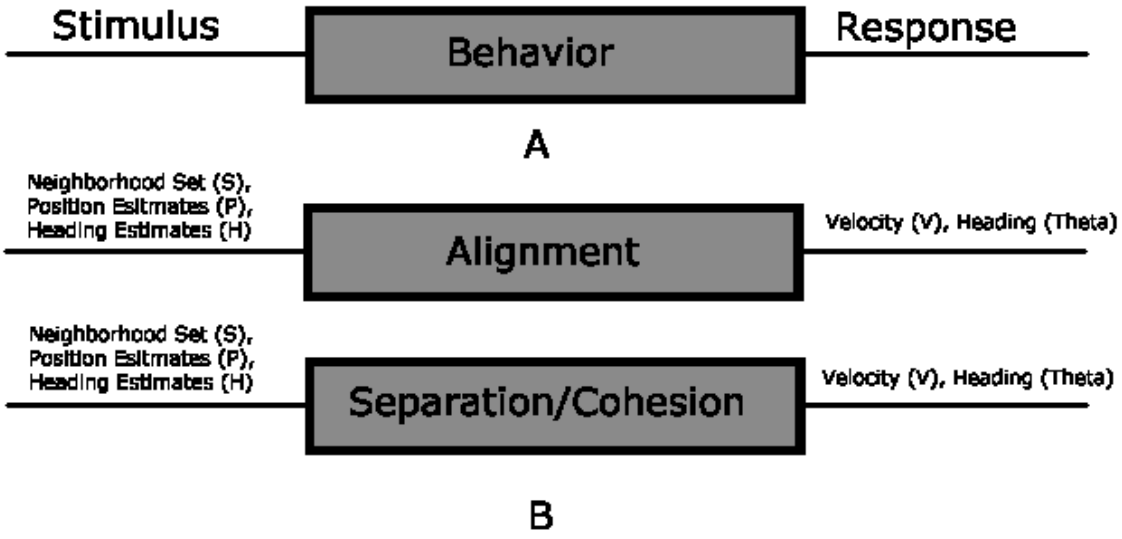

Figure 3.9: High Level View of the Stimulus/Reponse of the two swarm behaviors.

Recall, Equation 2.7 repeated here for clarity.

$$(\mathbf{v_{update}})_i = \sum_{p_j \in S_i} [(w_{periph})_{ij}(w_d)_{ij}((w_{attract})_{ij}(\mathbf{v_{attract}})_{ij} + C_{align}(\mathbf{v_{align}})_{ij})]. \tag{3.14}$$

For the behavior implementation the v_{update} is split into two pieces to facilitate the two behavior implementation described above. Namely,

$$(\mathbf{v_{attract}})_i = \sum_{p_j \in S_i} [(w_{periph})_{ij}(w_d)_{ij}((w_{attract})_{ij}(\mathbf{v_{attract}})_{ij})]. \tag{3.15}$$

and,

$$(\mathbf{v_{align}})_i = \sum_{p_j \in S_i} [C_{align}(\mathbf{v_{align}})_{ij})]. \tag{3.16}$$

Recall, that,

$$(\mathbf{v_{attract}})_{\mathbf{ij}} = \mathbf{p_j} - \mathbf{p_i}. \tag{3.17}$$

Loop $\forall Neighbors$
  Determine Neighbors Relative Location
  Determine Region for the Neighbor
  Calculate W_{perph}, W_d and $W_{attract}$
  Calculate $v_{attract}$
$v_{update} = v_{update} + w_{perhp} * W_d * w_{attract} * v_{attract}$
End Loop
return v_{update}

Figure 3.10: *Cohesion/Separation Vector update algorithm.*

To simplify the position of the neighbor is defined relative to the swarm member yielding,

$$(\mathbf{v_{attract}})_{\mathbf{ij}} = \mathbf{p_j}. \tag{3.18}$$

Using the disparity measurements the relative location of the swarm member can be determined. This information yields the $v_{attract}$.

3.2.8 Alignment Behavior Design. Similar to the previous section the Alignment behavior receives the same information about its neighbors and their approximate relative locations. Using this information the alignment behavior determines the region each neighbor lays in and computes the update vector as defined below in Figure 3.11.

Loop $\forall Neighbors$
  Determine Neighbors Relative Location
  Determine Region for the Neighbor
  Calculate v_{align}
  $\mathbf{v_{update}} = \mathbf{v_{update}} + \mathbf{v_{align}}$
  End Loop
  return v_{update}

Figure 3.11: *Alignment Vector update algorithm.*

Recall from Chapter 2,

$$(\mathbf{v_{align}})_{\mathbf{ij}} = (\mathbf{v_{dir}})_{\mathbf{j}}. \tag{3.19}$$

Kadrovach's swarm simulation has global knowledge of the headings of all members. The swarm model algorithm does not have this luxury since communication is prohibited. Instead the equation below is used to update the heading. The old heading information is stored for each iteration if no old heading information is available the heading is simply the unit vector of the relative position vector.

$$(\mathbf{v_{align}})_{\mathbf{ij}} = \mathbf{p_{jnew}}/||\mathbf{p_{jnew}}||. - \mathbf{p_{jold}}/||\mathbf{p_{jold}}||. \tag{3.20}$$

3.3 Summary

This chapter describes the physical implementation of the swarm model for use in the tests that follow. The Pioneer P2-AT8 robot hardware, Videre cameras, and visual processing laptop are discussed providing the reader an understanding of the computation abilities and limitations of our design. Next the software is discussed beginning with the visual sensor. The fast color segmentation algorithm is described as the means of identifying swarm members. The implementation details including the method definition of the color *orange* are described. The stereo vision implementation and distance estimates are discussed. Finally, the design and implementation of the two behaviors to implement the physical swarm are outlined. The next chapter outlines the experiments to test the performance of the implemented swarm controller.

IV. Experiments, Results, and Analysis

This chapter describes the design of experiments and metrics for the characterization and evaluation of the swarm control algorithm as implemented on the Pioneer AT robots described in Chapter 3. There are three separate experiments each designed to evaluate a different portion of the system. First, two experiments are conducted to evaluate the performance of the vision sensor implementation. Next, a series of stationary target tests are conducted to evaluate the performance of the swarm controller following a stationary target. Finally, a set of tests are conducted to evaluate the performance of the system following a dynamic target.

4.1 Visual Sensor Experiments

This section describes the two visual sensor calibration tests conducted with the system. The goal of these tests is to establish a baseline for the performance of the Stereo Cameras with the SVS C++ classes and the visual sensor as a whole. This section begins by evaluating the stereo vision cameras alone. Next, the fast color segmentation algorithm is combined with the stereo processor and is tested.

4.1.1 Videre Camera Disparity Test. This experiment determines the performance of the Videre cameras in the absence of the fast color segmentation algorithm. The cameras were placed known distances ranging from [20in-128in] from another robot and 100 samples of the estimated distance were recorded. This distance was varied as shown in Table 4.1 along with average estimated distance (mm), standard deviation, average absolute error (mm) and standard deviation.

Table 4.1: Summary of the Videre Camera Performance Test.

Test	act dist (mm)	avg est dist (mm)	std dev	avg error (mm)	std dev
1	508	N/A	N/A	N/A	N/A
2	990.6	1009	37.7	24.6	34.0
3	1143	1153	14.3	10.9	13.4
4	1371.6	1381	8.9	11.1	6.6
5	1803.4	1842	28.4	38.5	28.4
6	2336.8	2389	22.5	52.3	22.5
7	2870.2	2898	43.3	40.5	31.7
8	3251.2	3268	19.0	20.0	15.2

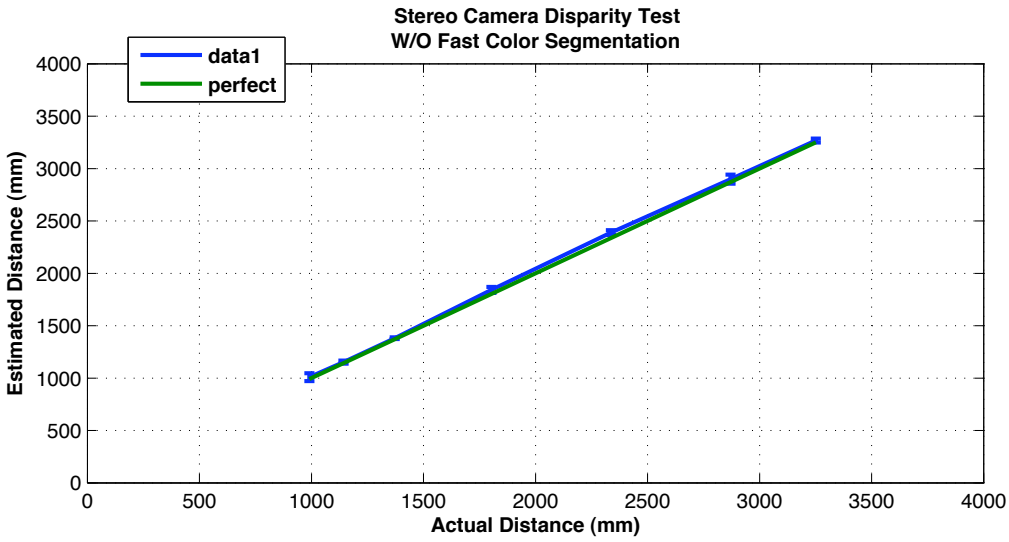

Figure 4.1: *Disparity Test Results.*

It is important to note the first test failed because the disparity information was too noisy due to the target being too close to the observer to calculate an accurate estimate of distance. That is the target fell outside 32 pixel horopter. From Figure 4.1.1 it is clear that the SVS stereo system provides a very accurate estimate of the distance from the cameras in the ranges tested. The target used in these tests was the entire robot outline not just the orange cone as will be for the fast color segmentation tests described in the next subsection. The entire robot gives a better correlation of the images in the two camera because of the differences in intensities and colors over the entire robot body. Since a better correlation can be established the resulting distance estimates are more accurate. This is evident in the next section.

4.1.2 Fast Color Segmentation with Disparity Test. The second visual sensor experiment goes a step further and evaluates the average error of the fast color segmentation algorithm when combined with the Videre Camera's disparity measurements. This section begins with a discussion of the test setup used to evaluate the performance of the visual sensor. Finally, the results of the tests are presented and analyzed.

The test is setup by placing the follower robot with orange cone swarm marker in view at a known distance from varied [1ft - 20ft] in increments of one foot. At each test distance 100 estimated ranges are recorded. The cone is kept directly in the center of the imager

throughout the test series. Off-angle performance is not tested. The fast color segmentation algorithm segments the orange regions of the image and passes the image coordinates of the cone to the stereo processor. The stereo processor then computes the average distance of the objects in this image region ignoring any undefined regions. This average distance is recorded and compared the known test distance. This is a slightly different the test in the previous subsection in that only the region returned by the fast color segmentation algorithm is used to estimate distance. Recall, the previous test included the entire robot as a target. The summary of the results is shown in Table 4.2 and graphically depicted in Figure 4.2.

Table 4.2: Summary of the Visual Sensor Disparity with Fast Color Segmentation.

Test	act dist (mm)	avg est dist (mm)	std dev	avg error (mm)	std dev
1	304.8	5322.2	1476.9	5017.2	1476.8
2	609.6	1512.7	914.4	903.1	198.1
3	914.4	1217.2	50.7	302.8	50.7
4	1219.2	1458.1	10.2	238.8	10.2
5	1524	1843.3	3.74	319.3	3.7
6	1828.8	2232.1	25.9	403.3	26.0
7	2133.6	2659.8	9.5	526.2	9.5
8	2438.4	3074.3	14.3	635.9	14.3
9	2743.2	3503.9	19.0	760.7	19.0
10	3048	3986.9	25.0	938.2	25.0
11	3352.8	4378.5	29.0	1025.7	29.0
12	3657.6	4907.5	38.0	1249.9	38.0
13	4572	5458.7	46.9	886.7	46.9
14	4876.8	6015.9	63.3	1139.1	63.3
15	5181.6	6521.0	62.2	1339.4	62.2
16	5486.4	7015.9	70.8	1529.5	70.8
17	5791.2	7509.73	81.7	1718.5	81.7
18	6096	8247.3	74.3	2151.3	74.3
19	6400.8	8777.6	59.2	2376.8	56.2
20	6705.6	9513.6	75.7	2808.0	75.7
21	7010.4	10095.2	72.6	3084.8	72.6
22	7315.2	11137.0	128.0	3821.8	128.1
23	7620	12081.8	133.7	4461.8	133.6
24	7924.8	12868.7	119.2	4943.9	119.2
25	8229.6	13411.7	101.5	5182.1	101.5
26	8534.4	14160.7	192.0	5626.3	192.0
27	8839.2	15480.9	228.7	6641.7	228.7
28	9144	17660.0	246.6	8516.0	246.6

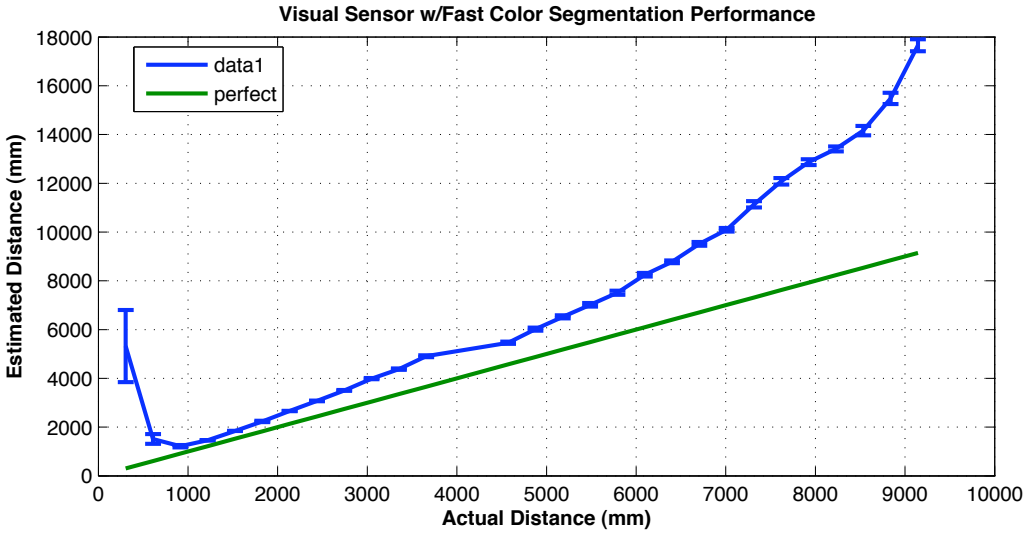

Figure 4.2: *Fast Color Disparity Test Results.*

From Table 4.2 and Figure 4.2 show again that objects too close to the visual sensor present unreliable data to the swarm system. Therefore, it is important to keep Kadrovach's regions out of the area of poor performance. Again these objects lie outside the horopter of the stereo processor. Around 900mm the information is reliable and scales approximately linearly until about 3000mm. As expected, the error increases as a function of distance due to decreased range resolution noted in Section 2.3.2.2. That is as you get further away from the sensor the readings are less accurate due to reduced resolution.

Using these test results the neighborhood distances parameters are defined for the subsequent tests. The parameters are selected to keep the swarm algorithm within the range of good performance of the visual sensor. Figure 4.3 shows the neighborhood model again for clarity. Table 4.3 shows the summary of the values defining the neighborhoods for our swarm controller for the subsequent tests. Thus, d_{min} was selected to be 1200mm giving some room for the visual sensor to detect the swarm member was in R_1. The comfort zone is defined to be 120mm wide by setting $d_2 = 1320$mm. Finally, d_3 is set to 3000mm for the tests that follow.

Recall from Kadrovach's model the swarm members desire to remain in the *comfort zone* defined by $d_{min} <= d <= d_2$ or between 1200mm and 1320mm. Therefore, in the following tests it is expected the swarm member should attempt to settle into this *comfort*

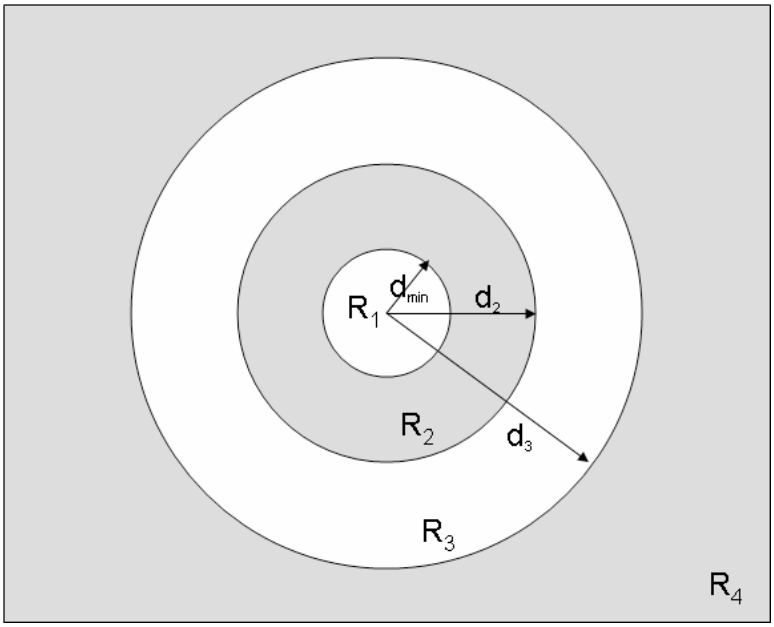

Figure 4.3: *Neighborhood Model* [30].

Table 4.3: Neighborhood Parameters.

Parameter	Distance (mm)
d_{min}	1200
d_2	1320
d_3	3000

zone and maintain it for the remainder of the test sequence. For the tests that follow the correct behavior of the swarm with respect to the regions defined in this section defines success of the test.

4.2 Stationary Target Tests

This subsection describes the stationary target test environment, experimental setup and provides a summary of the results. The goal of the stationary target tests is to see if the swarm model performs as expected against a simple stationary target. To accomplish this goal, the swarm model is tested in three different initial configurations shown in Figure 4.4. In these three configurations the swarm member's starting position is varied from directly behind (direct approach), offset to leader's left (left approach) and offset to the leader's right (right approach). These tests provide the confidence necessary to proceed with testing with a dynamic target. This section begins with a description of the test environment.

4.2.1 Environment. The environment for stationary tests is the Wright Field Gymnasium, Wright Patterson AFB, Ohio. The tests are conducted on the baseline of the Wright Field basketball court. The test area was marked with white tape marks every two feet to provide a ground truth reference for the data collection camera. The video data is recorded from an observation deck approximately 20 feet above the test area. No obstacles are presented to the system in these tests. Only the leader robot with an orange cone for identification and the follower swarm member occupy the test area. The next section discusses the setup for the stationary target tests.

4.2.2 Experimental Setup. Each experiment begins in one of the three initial configurations shown in Figure 4.4. A target robot is placed a known distance and remains stationary. The swarm robot approaches the target robot a position offset to right, offset to the left or directly behind the target robot. During the test the robot approaches the target and settles into the *comfort zone* behind the target robot. The swarm robot's velocity is limited to 100mm/s for safety. During each test both robot's motion was recorded at 320x240 resolution using a data collection camera operated from an observation deck. The test area was lined on both sides with white tape marks every two feet to provide a reference point for post test data analysis.

The parameters of the test are summarized in Table 4.4.

Table 4.4: Stationary Target Parameters.

Parameter	Value
d_{min}	1200
d_2	1320
d_3	3000
C_{periph}	1
C_{align}	1
$C_{repulse}$	1
$C_{attract}$	1
$V_{maxrobot}$	100 mm/s
MaxTurn	5 degrees
update parameter	.25

Five tests were conducted for both right and left approach tests and ten were conducted for the direct approach tests. These tests provided confidence in the algorithm as designed.

Since manual image analysis is required, no estimated distances are presented in this section only a qualitative discussion of the tests.

Figure 4.4: *Initial Configurations for Stationary Target Tests.* From left to right: Left Approach Test Configuration, Direct Approach Configuration, and Right Approach Test Configuration.

4.3 Stationary Target Test Results

All of the tests conducted for the stationary target tests demonstrated expected behavior of the swarm model. The results are discussed in the following three sections. Each sub section describes the results of each series of stationary target tests, provides general observations and concludes with a rudimentary analysis of a sample video sequence of the test. Complete video analysis is omitted because all analysis is obtained via manual analysis of individual frames.

4.3.1 Left Approach Test Results. This section describes the qualitative results of the left approach tests, describes general observations, and concludes with a rudimentary analysis of a sample test video sequence.

Table 4.5: Left Approach Test Summary

Test	Duration (s)	Back up (s)	Forward (s)	Oscillations	Notes
1	71	48	53	1.5	Oscillates in c_{zone}
2	75	45	50	2.0	Oscillates in c_{zone}
3	62	44	N/A	N/A	Doesn't Oscillate
4	71	48	53	2.5	Oscillates in c_{zone}
5	71	45	51	2.5	Oscillates in c_{zone}

During all the left approach tests the follower begins by turning to the left to directly approach the target robot. This demonstrates the ability of the robot to correct heading provided the orange identification cone can be seen in the visual sensor. After making the initial turn, the robot approaches until it reaches distance near the *comfort zone*. That is

around d_2 or 1320mm. In all but the third test, the robot begins a series of oscillations while attempting to settle into the *comfort zone*. When the robot senses it has left the *comfort zone* it correctly backs up to reestablish position within the *comfort zone*. The swarm member then begins a series of oscillations in an attempt to stay in the *comfort zone*. From Table 4.5, it can be seen that the time of the first back up are around 46 seconds. The differences in these times can be accounted for by variations in the sensor readings and by the error in the manual video analysis. Furthermore, the times of the first forward movement after the backup all begin around 52 seconds. The differences are attributed to variation in sensor readings and error in detecting the first resumption of forward motion by the video analyst. Clearly, the swarm model as implemented performs as expected.

The third left approach test differs slightly from the other four tests and warrants further discussion. During the third test, the robot approaches the target and appears to halt in the *comfort zone*. A quick manual image analysis estimates this distance to be 1316.94mm or within the *comfort zone*. Initially, it appeared that this test ended with the robot not oscillating; however, further image analysis showed the robot did oscillate in the *comfort zone*. This is difficult to detect with just the eye due to camera motion and the small amplitude of these oscillations.

Figure 4.5 shows a sample image sequence of the left approach test one. Beginning in the upper left hand corner, you can see the initial configuration of the robots with the swarm member offset to the top of the image. The second image shows the swarm member turning to correct its heading and approach the *comfort zone* as expected. Images three through seven show the continued approach of the swarm member until it passes beyond d_{min}. The last two images show the robot backing up to within the *comfort zone*.

This section described the results of the left turn tests conducted to confirm the performance of the algorithm implemented. The left turn tests all showed the expected behavior with the swarm member approaching a stationary target and attempting to settle into the *comfort zone* of Kadrovach's model. Initial analysis indicated left turn test three may have concluded with the swarm member settling into and maintain itself in the *comfort zone*; however, further detailed analysis showed the robot actually did oscillate with smaller

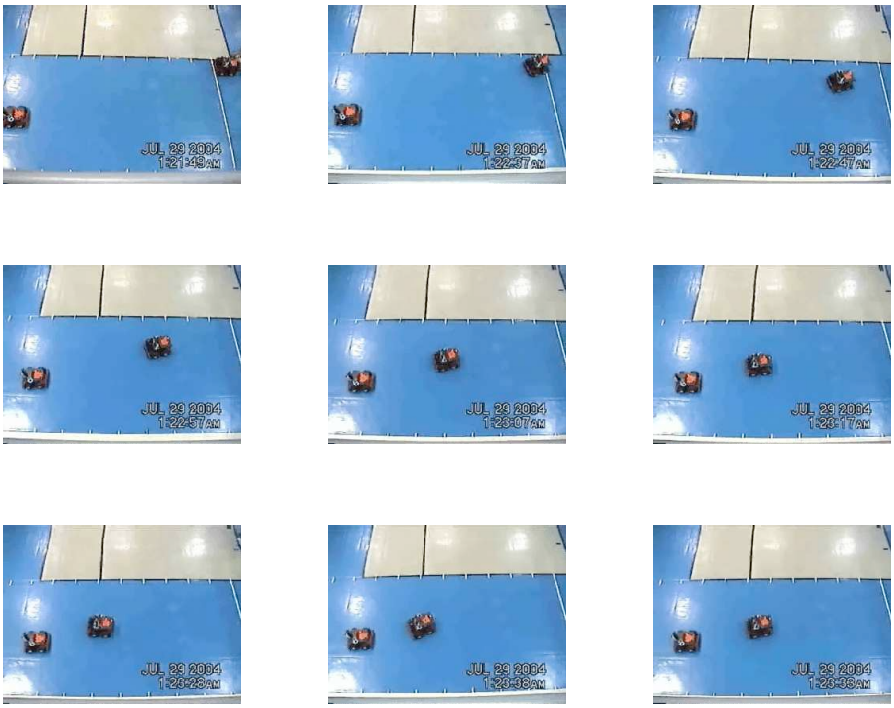

Figure 4.5: *Sample Image Sequence of Left Approach Tests.* Images progress in time from left to right and top to bottom.

amplitudes than the other tests. This was difficult to see on initial analysis due to camera jitter.

4.3.2 Right Approach Test Results. This section describes the qualitative results of the right approach tests, describes general observations, and concludes with a rudimentary analysis of a sample test video sequence.

Table 4.6: Right Approach Test Summary.

Test	Dur(s)	Back (s)	Forward (s)	Oscillations	Notes
1	83	38	46	4.0	Oscillates in c_{zone}
2	75	46	53	3.0	Oscillates in c_{zone}
3	74	46	53	3.5	Oscillates in c_{zone}
4	67	41	45	3.5	Oscillates in c_{zone}
5	75	44	47	4.5	Oscillates in c_{zone}

During all the right approach tests the follower begins by turning to the right to begin approaching the target robot. This demonstrates the ability of the robot to correct heading provided the orange identification cone can be seen in the visual sensor as was seen in the previous series of left approach tests. Just as in the left approach tests, the robot approaches until it reaches distance near the *comfort zone* around d_2 or 1320mm. The robot begins a series of oscillations while attempting to settle into the *comfort zone*. When the robot senses it has left the *comfort zone* it correctly backs up to reestablish position within the *comfort zone*. The swarm member then begins a series of oscillations. From Table 4.6, the time of the first back up are around 42 seconds. Similar to the previous tests the differences in these times can be accounted for by the variation in the sensor readings and by the video analysis. Furthermore, the times of the first forward movement after the backup all begin around 50 seconds. The differences are attributed to variation in sensor readings and error in detecting the first resumption of forward motion. Clearly, the swarm model as implemented performs as expected.

Figure 4.6 shows a sample image sequence of the right approach tests. This image sequence shows right approach test one. Beginning in the upper left hand corner, you can see the initial configuration of the robots with the swarm member offset to the bottom of the image this time. As in the left approach tests, the second image shows the swarm member turning to correct its heading and approach the *comfort zone* as expected. Images three through five show the continued approach of the swarm member until it passes beyond d_{min}. Images six through eight show the backup portion of an oscillation. The final image shows the swarm member beginning to close the distance with its leader again.

This section described the results of the right approach tests conducted to confirm the performance of the algorithm implemented. The right approach tests all showed the expected behavior with the swarm member approaching a stationary target and attempting to settle into the *comfort zone* of Kadrovach's model. The next section evaluates the performance of the final configuration of stationary target tests.

4.3.3 Direct Approach Test Results. This section describes the qualitative results of the right approach tests, describes general observations, and concludes with a rudimentary analysis of the a sample sequence of a sample test.

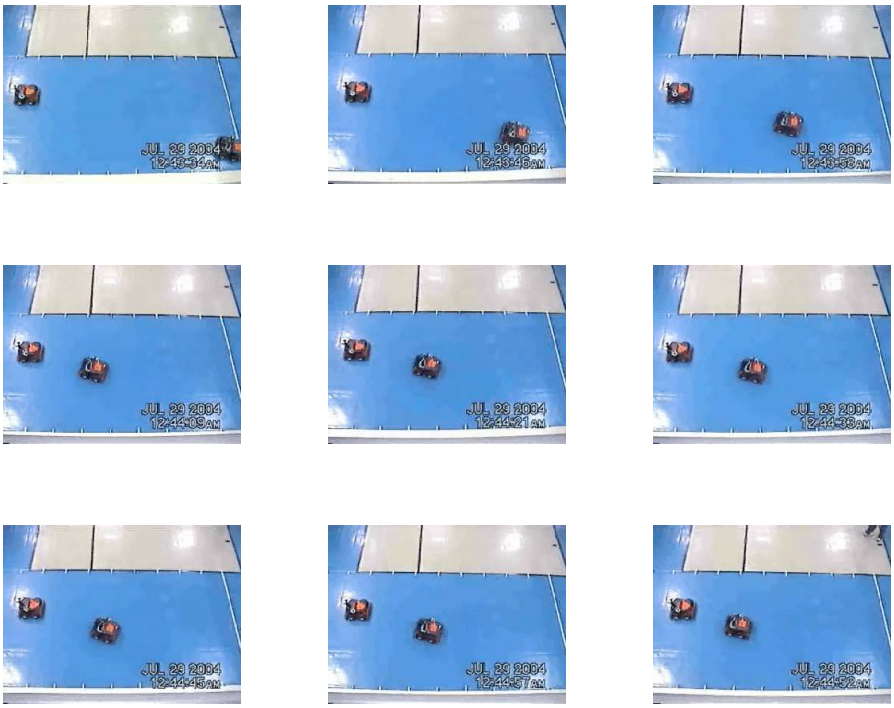

Figure 4.6: *Sample Image Sequence of Right Approach Tests.* Images progress in time from left to right and top to bottom.

All ten direct approach begin with the swarm member directly behind the target robot. The robot then approaches the *comfort zone*. After translating through the *comfort zone* the swarm member backs up to maintain its position and in all but test 4 and 5 oscillates until the end of the tests. Test 6 and 7 have oscillations that decrease in amplitude with time. Test 9 and Test 10 both have exhibit oscillations followed by a temporary correction and then a resumption of oscillatory behavior around the *comfort zone*. From Table 4.7, all tests have an initial backup of the robot around 40 seconds. Additionally, in the tests which exhibit oscillatory behavior the forward motion resumes around 45 seconds. Like the left approach test two tests appear to have the robot settle into the *comfort zone* without oscillating.

Test 3 and Test 5 of the direct approach tests initially appeared to exhibit no oscillatory behavior. As with Test 3 of the left approach tests, a more detailed image analysis is used

Table 4.7: Direct Approach Test Summary.

Test	Dur (s)	Back (s)	Forward (s)	Oscillations	Notes
1	99	29	31	12	Oscillates in c_{zone} amp decreases
2	99	44	51	5	Oscillates in c_{zone}
3	73	38	N/A	N/A	Doesn't Oscillate
4	105	39	47	6	Oscillates in c_{zone}
5	56	41	N/A	N/A	Doesn't Oscillate
6	104	42	50	8	Oscillates in c_{zone} amp decreases
7	85	38	44	11	Oscillates in c_{zone} amp decreases
8	70	40	45	4	Oscillates in c_{zone}
9	73	36	41	5	Oscillates; moves closer; oscillates
10	87	42	46	11	Oscillates; moves closer; oscillates

to further investigate this apparent behavior. Direct Approach Test 3 analysis showed behavior similar to that seen in left approach test 3. Namely, small oscillations around the *comfort zone* which are difficult to see without detailed analysis due to camera jitter. Further analysis of test 5 shows the same results. Therefore, further analysis shows all tests displayed oscillatory behavior around the *comfort zone*.

Figure 4.7 shows a sample images sequence of the direct approach tests. It depicts direct approach test one. The image in the upper left shows the initial configuration of the test with the swarm member directly behind the target robot. Images two through four show the initial approach of the robot in to the *comfort zone*. Images five through nine show the oscillatory nature of the behavior around the *comfort zone*.

This section described the results of the direct approach tests conducted to confirm the performance our algorithm. The direct approach tests all showed the expected behavior with the swarm member approaching a stationary target and attempting to settle into the *comfort zone* of Kadrovach's model. Two tests in this section appeared to exhibit no oscillatory nature; however, detailed image analysis revealed true oscillatory behavior for both Test 3 and Test 5 initially not detected due to camera jitter. The next section discusses the tests conducted against a moving target in three different configurations.

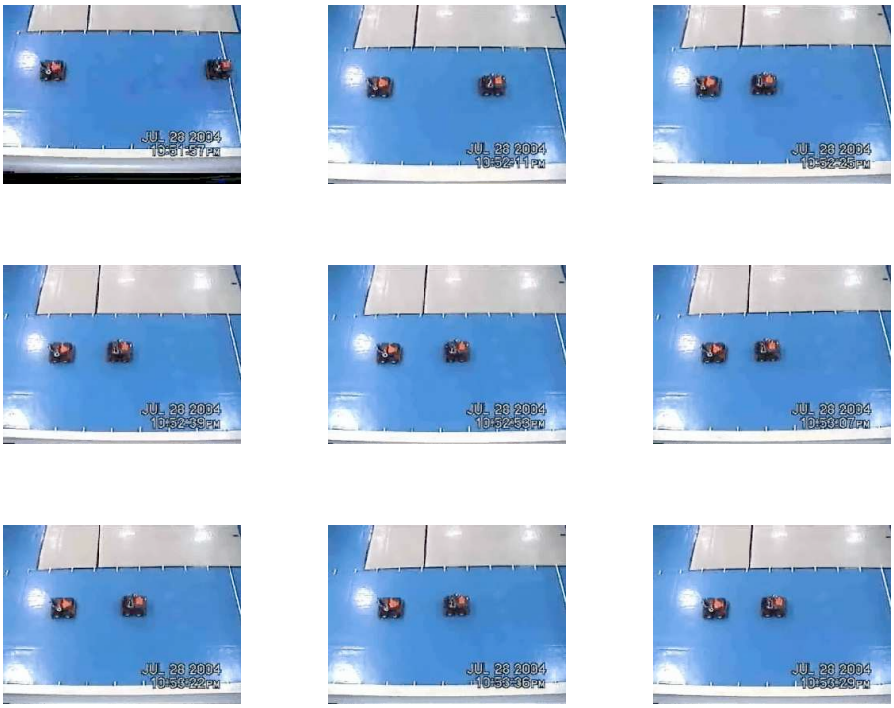

Figure 4.7: *Sample Image Sequence of Direct Approach Tests.* Images progress in time from left to right and top to bottom.

4.4 Dynamic Target Tests

This subsection describes the dynamic target tests, environment, setup and provides a summary of the test results. The goal of the dynamic target tests is to demonstrate performance of the implemented swarm control algorithm with moving targets. The test environment is described in the next sub section. Then, the experimental setup is outlined. Finally, the results and analysis of the dynamic target tests is presented.

4.4.1 Environment. The environment for stationary tests is the Wright Field Gymnasium, Wright Patterson AFB, Ohio. The tests are conducted on the baseline of the Wright Field basketball court. The test area was marked with white tape marks every two feet to provide a ground truth reference for the data collection camera video. The video data is recorded from an observation deck approximately 20 feet above the test area. No

obstacles are presented to the system in these tests. Only the leader robot with an orange cone for identification and the follower swarm member occupy the test area. Experimental setup is discussed in the next subsection.

4.4.2 Experimental Setup. The swarm model is evaluated against a dynamic target in three scenarios. First, similar to the direct approach tests the target robot is placed a set distance in front of the swarm member. This time the target robot, leader, translates at 50mm/s while the swarm member begins translating at 100mm/s. The speeds of the robots are kept small due to the limited space for conducting the tests and to limit the movement by the data collection camera. This setup allows the swarm member to approach the target, then settle into its comfort zone and continue translating until the end of the test. This test is run twelve times to evaluate the performance of the swarm model in a translation mode. This is similar to a formation joining and maintenance behavior as seen in other formation control experiments in literature.

Next, a series of tests evaluates the swarm model in another dynamic mode. The initial setup is similar to the translational tests described in the previous paragraph. The target translates at 75mm/s and the swarm model robot begins translating at 100mm/s to allow the swarm member to approach into the comfort zone. After translating a distance the leader robot performs a left turn and continues to translate at 75mm/s. The speeds of the robots are kept small to the limit space used to conduct the tests and to limit the movement of the data collection camera. The follower is expected to approach, establish its position in the comfort zone and then maintain its relative position while performing the left turn and following the leader. This test is run ten times to evaluate the performance of the swarm model following a dynamic target who turns. Again this is similar to formation joining and maintenance behaviors as seen in other formation control experiments in literature.

Finally, a series of right turn tests is conducted with setup identical to the previous left turn tests; however, this time the leader performs a right turn after translating at 75mm/s. Again the swarm member translates at 100mm/s to allow for the swarm member to approach, establish position in the comfort zone and maintain that position while tracking a leader turning to the right. Again speeds of both robots are limited due to space and data collection camera motion constraints.

The initial configurations of the tests are shown in Figure 4.8.

Figure 4.8: *Initial Configurations for Dynamic Target Tests.* From left to right: Left Turn Test Configuration, Translation Configuration, and Right Turn Test Configuration.

4.5 Dynamic Target Test Results

All of the tests conducted for the dynamic target tests demonstrated expected behavior of the swarm model implemented. The results are discussed in the following three sections. Each sub section describes the results of the translation, left turn, and right turn tests. Full data analysis of the estimated distances is included for five test sequences: two translation tests, two left turn tests, and one right turn test.

4.5.1 Translation Test Results. This section presents the results of the twelve translation tests. All of the tests conducted for the translation target tests demonstrated expected behavior of the swarm model. The section begins with the general observations of the series of tests. Then two detailed analyses of translation test 4 and translation test 8 conclude this section. Only these two tests are analyzed due to the manual analysis needed to recover estimated distance from video data. Due to the data collection camera motion the raw video data is sampled at 2Hz and then distance is estimated frame by frame using the white tape mark's known separation distance to calibrate the pixel length. Then the center of the leader's cone and front of the robot are used as reference points. Using these points a Euclidean pixel distance between the two robot's is calculated and multiplied by the estimated pixel length to obtain an estimate of the separation distance of the leader and the follower.

All of the twelve translation tests begin with the swarm member directly behind the lead robot. The tests begin with the follower closing the distance to the follower at approximately 50mm/s. Upon reaching the *comfort zone* the follower halts or backs up to maintain its position in the *comfort zone*. Oscillations begin and continue for the remainder

of the test sequence. All of the tests have the first backup around 37 seconds into the test sequence. The two outliers, tests 5 and 11 are explained by the follower starting its motion after the leader has begun translating. The first movement forward after the first stop begins around 4 seconds after the first backup. Then, a variety of oscillations occur for the remainder of the test sequence. The differences in the number of oscillations reported is due to sensor error and visual analysis error similar to what is reported during the stationary tests in the previous section.

Table 4.8: Translation Test Summary.

Test	Dur (s)	Back (s)	Forward (s)	Oscillations	Notes
1	196	35	38	12	
2	180	36	40	9	
3	118	39	43	6	
4	94	41	47	4	
5	110	57	62	3	Late Follower Start
6	96	43	46	5	
7	104	53	57	5	
8	97	32	35	9	
9	99	35	38	8	
10	93	35	39	5	Delay Restart after 4 Oscillations
11	105	68	70	5	Late Follower Start
12	95	40	40	11	Lead Batt dies @ 80s

Figure 4.9 shows the estimated distance in mm plotted against the time of test 4 in seconds. There are 4 horizontal lines depicting the boundaries of the four neighborhood regions of the Kadrovach model. d_{min} is the green line. d_2 is the red horizontal line. A blue horizontal line depicts d_3. Finally, start, in purple, denotes the known starting distance of 8ft or 2438.4mm. The first observation is the difference in estimated starting distance and known starting distance. This test sequence begins with the data collection camera appearing to be slightly angled towards the leader in order to maintain the leader in the frame. This angle distorts the image and results in the large error in starting position. Furthermore, during the test sequence the leader disappears from the frame intermittently as the data collection camera attempts to hold the robots in the frame.

Given the test setup and known speeds of the two robots the slope of the line in the graph should be -50mm/s as this is the difference in speeds, or closure rate of the follower.

Using Data Cursors in MATLAB at two points (10, 2767) and (30, 1606) yields an estimated closure rate of -58mm/s. Furthermore, the initial back up of the follower, noted in Table 4.8, at approximately 41 seconds. The forward motion begins again at approximately 47 seconds. Finally, the four peaks of the oscillations are easily seen. Assuming a starting distance of 2438.4, closure rate of the follower of -50mm/s, and a perfect sensor we would expect the swarm member to cross the d_2 line, entering the *comfort zone* at 22.368 seconds and exit the *comfort zone* at 24.768 seconds. This test produces inconclusive results due to errors. Section 4.6 attempts to detail sources of error in the system.

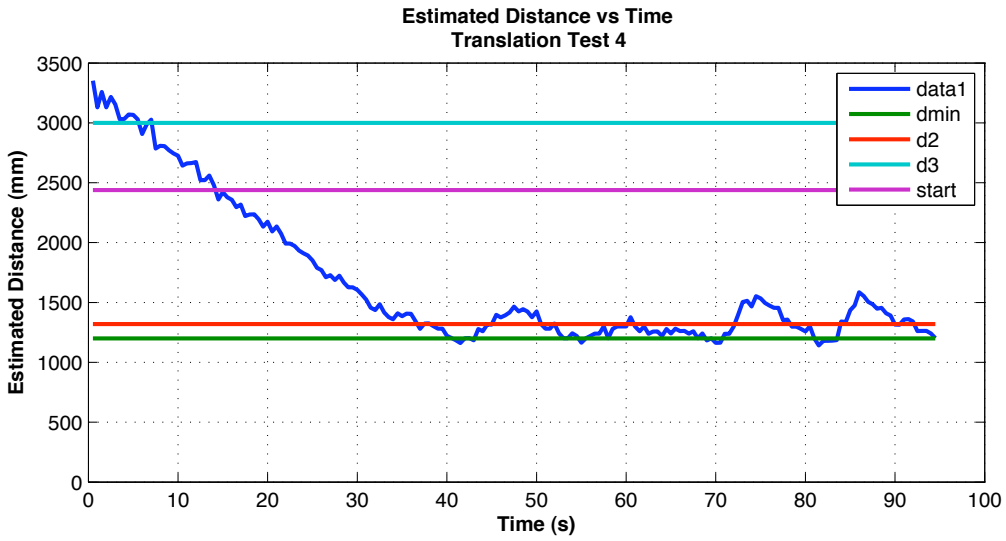

Figure 4.9: *Translate Test 4 - Estimated Distance vs Time.*

Figure 4.10 shows the estimated distance in mm plotted against the time of test 8 in seconds. Identical to translation test 4, there are 4 horizontal lines depicting the boundaries of the four neighborhood regions of the Kadrovach model. d_{min} is the green line. d_2 is the red horizontal line. A blue horizontal line depicts d_3. Finally, start, in purple, denotes the known starting distance of 8ft or 2438.4mm. Unlike the translate test 4 above, a very good approximation of the initial distance is seen to begin this test sequence. Recall, translation test 4 began with camera angled toward the leader. Translation test 8; however, begins with leader in the middle of frame. The data collection camera also appears to be more directly overhead throughout the test. Furthermore, the data collection camera keeps the robots in the middle of the frame for the duration of the tests. The estimated distance is 2498.72 for an error of only 60.32mm. The same technique described above for two points

(10, 2106) and (28.5, 1279) yields an estimated closure rate of -44.7mm/s. As described previously the expected crossover of d_2 into the *comfort zone* and out of the *comfort zone* are 22.368 seconds and 24.768 seconds, respectively. Figure 4.10 shows the follower entering the *comfort zone* at between 27 and 27.5 seconds an error of approximately 5 seconds. Exiting the *comfort zone* is estimated to occur between 29.5 and 30 seconds, an error of approximately 5 seconds. From Table 4.8 we note the initial data analysis indicates the robot began backing around 32 seconds which is near what the figure shows. Table 4.8 notes forward motion again at 35 seconds which is clearly reflected in Figure 4.10. Finally, the table notes nine oscillations of which only approximately eight are shown in the figure. This test shows a good depiction of the expected behavior of the swarm model.

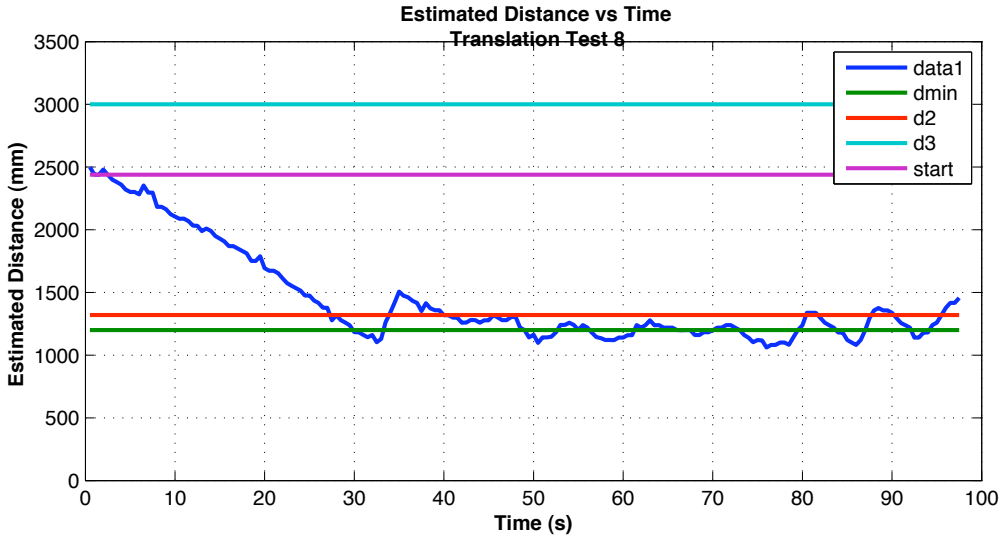

Figure 4.10: *Translate Test 8 - Estimated Distance vs Time.*

This subsection describes the first series of dynamic tests of the swarm model. All of the tests conducted for the translation target tests demonstrated the expected behavior of the swarm model. The section began with general observations of all 12 tests conducted including description of similar and outlier behavior. After the general observations a detailed analysis of the results of translation test 4 and translation test 8 is presented. While the results of test 4 proved inclusive due to errors in the video analysis, test 8 produced excellent behavior of the swarm model. The next section presents the analysis of the left turn tests.

4.5.2 Left Turn Test Results. This section presents the results of the ten left turn tests. All of the tests conducted for the left turn tests demonstrate the expected behavior of the swarm model. The section begins with the general observations of the test series. Then two detailed analyses of left turn test 1 and left turn test 8 conclude this section. Only these two tests are analyzed due to the manual analysis needed to recover estimated distance from video data. Due to the data collection camera motion the raw video data is sampled at 2Hz and then distance is estimated frame by frame using the white tape marks and known separation distance to calibrate the pixel length. Then the center of the leader's cone and front of the robot are used as reference points. Using these points a Euclidean pixel distance between the two robot's is calculated and multiplied by the estimated pixel length to obtain an estimate of the separation distance of the leader and the follower.

All of the ten left turn tests begin with the swarm member directly behind the lead robot at the top of the frame. After translating at 75mm/s for 30 seconds the leader performs a left turn and resumes translating at 75mm/s second. Given the difference in leader translation speed the closure rate for this test sequence is 25mm/s not 50mm/s as seen in the translation tests. Upon reaching the *comfort zone* the follower halts or backs up to maintain its position in the *comfort zone*. Oscillations begin and continue for the remainder of the test sequences. Table 4.9 shows the results of the ten tests. For all ten tests the leader begins turning around 37 seconds. Subsequently, the follower begins its turn around 46 seconds. The follower then backs between 60-70 seconds into the test. The variation in the back up times is due to the sensor noise and the manual video analysis. Test 9 is an outlier as the follower never backs up to maintain its position. This is due to the follower not overcoming its late start to begin the test sequence.

Figure 4.11 shows the estimated distance in mm plotted against the time of test 1 in seconds. As before, there are 4 horizontal lines depicting the boundaries of the four neighborhood regions of the Kadrovach model. d_{min} is the blue line. d_2 is the green horizontal line. A red horizontal line depicts d_3. Finally, start, in purple, denotes the known starting distance of 8ft or 2438.4mm. The data collection camera's position during the beginning of this test is approximately overhead as noted in the translation 8 test sequence; however the robots appear in the top of the frame and are further away from the camera position at the beginning of the test sequence. As observed in the translation tests

Table 4.9: Left Turn Test Summary.

Test	Dur (s)	Lead Turns (s)	Follow Turns (s)	Follow Backs (s)	Notes
1	85	35	47	72	Follow Backs x2
2	74	40	48	59	
3	73	39	46	66	
4	70	36	47	N/A	Never Backs
5	63	45	55	60	
6	66	38	48	70	
7	70	35	44	66	
8	73	39	49	59	Leader Late
9	88	36	43	N/A	Follow Late
10	81	31	41	65	

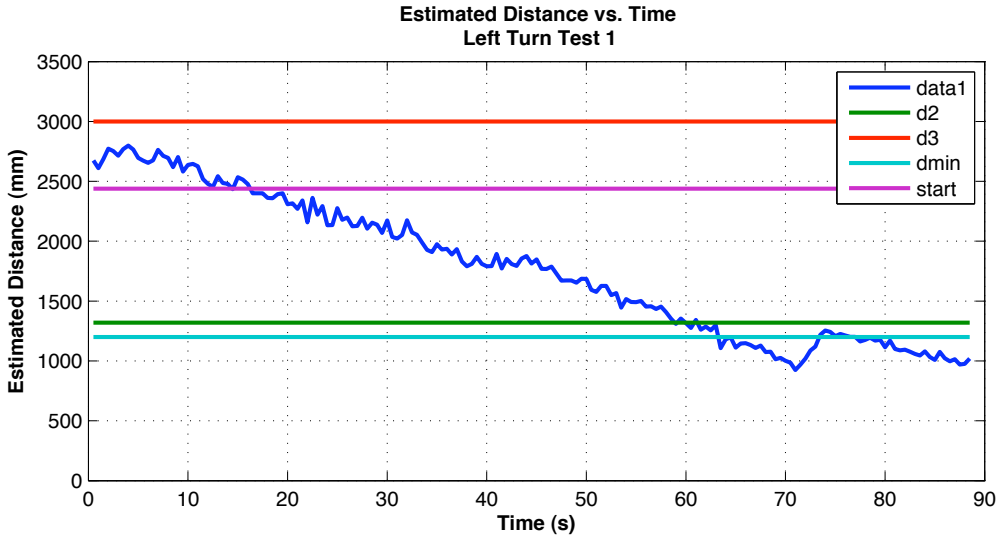

Figure 4.11: *Left Turn Test 1 - Estimated Distance vs Time.*

there is an error to begin the test sequence as the first data point measures the relative distance is estimated 2674.37mm for an error of 235.97mm.

Given the test setup and known speeds of the two robots the slope of the line in the graph should be -25mm/s as this is the difference in speeds, or closure rate of the follower. MATLAB data cursors are again used at two points (10, 2638) and (50, 1682) yielding an estimated closure rate of -23.9mm/s. While information on the turning times cannot be obtained from relative distance plots, the initial back up of the follower noted in Table 4.9 at approximately 72 seconds. Although the second backup noted in the table is not easily seen. Unlike the translation tests predicting the cross over times of the follower is not

straight forward because of the lower closure rate of 25mm/s. Assuming a starting distance of 2438.4, closure rate of the follower of -25mm/s, and a perfect sensor we would expect the swarm member to only close to 2438.4 - 30s*25mm = 1688.4 mm before the leader begins his turn. This makes estimation of the entry and exit times of the *comfort zone* difficult. It is possible to predict the swarm member to enter the *comfort zone* at approximately 44.736 seconds and exit the *comfort zone* at 49.536 seconds assuming a straight line closure rate of -25mm/s. From Figure 4.11 the entry and exit times of the *comfort zone* are between 58.5 and 59 seconds and between 63 and 63.5 seconds respectively.

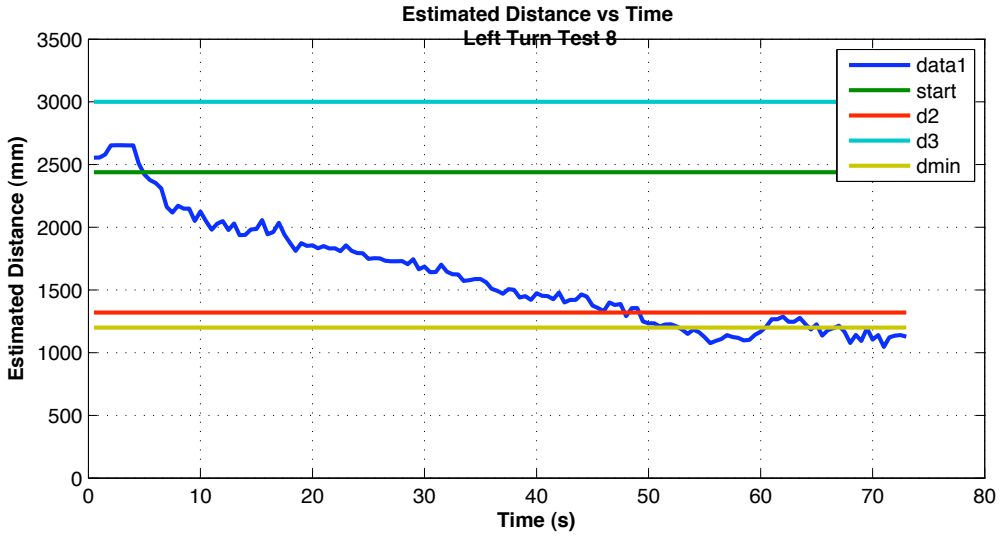

Figure 4.12: *Left Turn Test 8 - Estimated Distance vs Time.*

Figure 4.12 shows the estimated distance in mm plotted against the time of test 8 in seconds. As before, there are 4 horizontal lines depicting the boundaries of the four neighborhood regions of the Kadrovach model. d_{min} is the yellow line. d_2 is the red horizontal line. A blue horizontal line depicts d_3. Finally, start, in green, denotes the known starting distance of 8ft or 2438.4mm. The data collection camera's position is angled towards the follower as in translation test 8. The leader is late to start as noted in Table 4.9 this allows a more rapid initial closure rate as depicted in the first few seconds of Figure 4.12. Due to the late start the camera keeps the robots in the center of the frame for most of the test sequence. It is expected these results should be more accurate as is seen in translation test 8. As observed in the translation tests and similar to left turn test 1 above there is an error

to begin the test sequence as the relative distance is estimated at 2555mm for an error of 116.6mm.

MATLAB data cursors are again used at two points of the for estimation of the closure rates; however, due to the late start of the leader two estimates of closure rates are obtained, an initial and final. Due to the late start, its is expected the initial closure rate should be near the max velocity of the swarm member of 100mm/s. The two points used in the estimation are (4, 2653) and (6.5, 2310) yielding an initial estimated closure rate of 137.2 mm/s. To estimate the final closure rate points (10, 2125) and (20, 1836) are used yielding an estimated final closure rate of 26.9 mm/s. As discussed earlier, approximating *comfort zone* entry and exit times is more difficult in turning tests. This test is further complicated by the late start of the leader and different resulting closure rates. Therefore, the entry and exit times of the Figure 4.12 are only reported. They are between 47.5 and 48 seconds and 52.5 and 53 seconds, respectively.

This subsection describes the left turn tests of the swarm model. All of the tests conducted in this series demonstrates the expected behavior of the swarm model. The section begins with general observations of all ten left turn tests conducted including description the general observations of swarm member behavior. After these observations a detailed analysis of the results of left turn test 1 and 8 is presented. The next section analyzes the final series of dynamic tests, the right turn tests.

4.5.3 Right Turn Test Results. This section presents the results of the ten right turn tests. All of the tests conducted demonstrate the expected behavior of the swarm model. The section begins with the general observations of the right turn test series. Then concludes with a detailed analysis of right turn test 1. Only this test is analyzed due to the manual video analysis needed to recover estimated distance from video data. See Section 4.5.1 or Section 4.5.2 for detailed rationale.

All of the ten right tests begin with the swarm member directly behind the lead robot like the previous dynamic test sequences; however, this time the robots appear at the bottom of the video frame, closer to the data collection camera. This moves the camera into a better position and thus, more accurate distance estimates are expected. For this test sequence, the leader translates at 75mm/s for 30 seconds then performs a right turn before resuming

at 75mm/s second. The closure rate for this test sequence is 25mm/s again not 50mm/s as seen in the translation tests. Upon reaching the *comfort zone* the follower halts or backs up to maintain in the *comfort zone*. Oscillations begin and continue for the remainder of the test sequences. Table 4.10 shows the results of the ten tests. For all ten tests the leader begins turning around 36 seconds. Subsequently, the follower begins its turn around 42 seconds. The follower then backs between 50-70 seconds into the test. The variation in the back up times is due to the sensor noise and the manual video analysis technique.

Table 4.10: Right Turn Test Summary.

Test	Dur (s)	Lead Turns (s)	Follow Turns (s)	Follow Backs (s)	Notes
1	78	34	43	65	
2	91	40	50	51	
3	70	34	43	N/A	Follow Late
4	71	35	40	67	
5	70	34	40	67	
6	69	33	41	N/A	
7	68	34	42	N/A	
8	76	33	44	64	
9	71	38	45	56	
10	69	35	45	60	

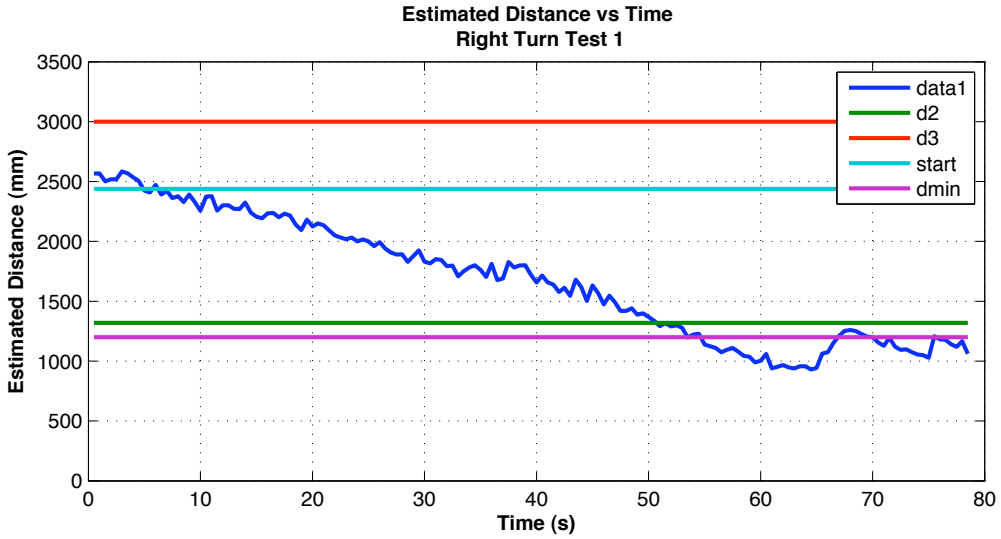

Figure 4.13: *Right Turn Test 1 - Estimated Distance vs Time.*

Figure 4.13 shows the estimated distance in mm plotted against the time of the right turn test 1 in seconds. As before, there are 4 horizontal lines depicting the boundaries of the four neighborhood regions of the Kadrovach model. d_{min} is the purple line. d_2 is the

green horizontal line. A red horizontal line depicts d_3. Finally, start, in blue, denotes the known starting distance of 8ft or 2438.4mm. The improved data collection camera position produces a better initial estimate of relative distance to begin right turn test 1. The initial distance in right turn test 1 is 2567mm which is only 128.6mm in error. For most of the tests the robots appear directly underneath the data collection camera's position. Using MATLAB data cursors to select points, points (5.5, 2424) and (30, 1831) are used to obtain a closure rate estimate of 24.2mm/s. As explained in Section 4.5.2, estimating ideal entry and exit times for the *comfort zone* is difficult but they are noted to be between 50.5 and 51 seconds and between 54.5 and 55 seconds, respectively. This is clearly the most accurate test of the three turning tests analyzed due to the superior camera position; however, the oscillations noted in Figure 4.13 are below the *comfort zone*.

This subsection described the results of the right turn test series. All of the tests conducted in this series demonstrated the expected behavior of the model. The section began with general observations of all ten right turn tests. These observations are followed by a detailed analysis of the results of right turn test 1. As has been noted throughout this chapter, there are many sources of error effecting the results. The next section identifies and analyzes these sources of error.

4.6 Error Analysis

This section discusses errors in both the system as implemented and the errors which could be introduced in the manual data extraction method. There are four possible sources of error affecting the results presented in this chapter. They are:

1. Stereo Vision Sensor Error
2. Error offset of the cameras
3. Error picking the center of cone and front of the robot.
4. Error in each pixel distance due to observer motion and camera angle

The next four paragraphs discusses each of these sources of error in detail and analyzes their effects on the results.

The stereo vision sensor introduces the first error. As we saw from Figure 4.1.1 the vision sensor is not a perfect visual sensor. Moreover from Figure 4.2 showed us the fast color segmentation algorithm introduces further error in the system. The increases with as the distance increases. From equation 2.22 we recall the resolution of the disparity measurements from a stereo camera scales quadratically with range. To combat this effect, the neighborhoods of the swarm model are kept under 3000mm. Figure 4.2 shows an average distance estimate of 4000mm at an actual distance of 3000mm. An error of over 1 meter. Furthermore, in the region around d_{min} and d_2 errors in are around approximately 300mm. This means the swarm member estimates it is about 300mm further away than the estimates from the data collection camera would show. This accounts for some of the oscillations occurring below d_{min} in the trials.

The second source of error in the results comes from the inability to observe the actual cameras and thus the position of the imagers in the data collection camera's video stream. Recall the distance in the results is estimated from the center of the leader's cone to the front of the follower robot. The pan-tilt camera head and thus the imagers of the stereo cameras are offset from the front of robot by 168mm. This means the data collection camera's reported distance is in error by 168mm. This is in the opposite direction of the error in the previous section. As the robot is measured to be closer than the robot perceives it is.

The third source of error is introduced by the manual selection of the center of the leader's cone and the front of the follower robot. The estimated error in the center of of the leader's cone is estimated to be one or two pixels. Given the average pixel distance from the estimations is between 15.6mm/pixel and 20.5 mm/pixel. Giving an estimated error of: $+/-2pixels*[15.6mm/pixel-20.5mm/pixel] = +/-[31.2mm-41mm]$. The second manual estimation is the front of the follower robot. Shadows and camera motion make this estimate harder. The error is estimated to be between two or four pixels. Again given the average pixel distance from the tests yields $+/-4pixels*[15.6mm/pixel-20.5mm/pixel] = +/-[62.4mm-82mm]$. In the worst case the errors are additive. This yields an estimated error in the $+/-(41+82) = +/-123mm$.

The final source of error in our data extraction is the data collection camera's motion and angle. The estimation process assumes the camera position is nearly directly overhead while this is true in some of the right turn and translational tests it is not true in general. Estimating this error is further complicated by the camera's motion throughout all of the tests. Furthermore, the error is not consistent as the camera angle varies with each test. Therefore this error is ignored for the analysis.

This subsection discusses the error sources in the system as implemented and the errors introduced by the manual data extraction method.

4.7 Summary

This chapter describes the design of experiments and metrics for the characterization and evaluation of the swarm control algorithm as implemented on the Pioneer P2-AT8 Robots described in Chapter 3. Three separate experiment sets are designed to evaluate a different portion of the system. First, two experiments are conducted to evaluate the performance of the vision sensor implementation. Next, a series of stationary target tests are conducted to evaluate the performance of the swarm controller versus a stationary target. Finally, a set tests are conducted to evaluate the performance of the system versus a dynamic target. The section concludes with an analysis of the errors in the system. The final chapter presents conclusions and future work.

V. Conclusions and Future Work

This research presented an adaptation of Kadrovach's Swarm Model [30] to a Pioneer P2-AT8 robot and demonstrated a real-time visual sensor model. This chapter gives final conclusions drawn from the research. Also, future research work is discussed.

5.1 Conclusions

This research sought to implement the swarm algorithm presented by Kadrovach in [30] on a Pioneer P2-AT8 robot. Three objectives are outlined in Chapter 1 and are summarized here:

1. Design and Implement Kadrovach's Swarm Model.
2. Design and Implement Kadrovach's Visual Sensor.
3. Evaluate the performance of the implemented swarm.

The robots are not allowed to communicate explicitly except through the environment. Limiting the implementation to *passive* sensing to simulate hostile environments. All of the objectives are successfully completed. The swarm model is implemented on the Pioneer AT robots using the Videre cameras and fast color segmentation algorithm [12] to produce position estimates to stationary and dynamic targets. This swarm algorithm uses no communication to produce the desired behavior. Complete analysis of the swarm algorithm is limited due to the complications of errors introduced as described in section 4.6. The limitations of the complex manual data extraction technique limits the results presented. The analysis showed the swarm algorithm works as designed for simple tasks and small swarms.

5.2 Future Work

This section gives ideas for further research to improve swarm applications at AFIT. The following topics warrant further investigation.

5.2.1 Oscillations. In all the results presented, the follower robot oscillated around the comfort zone. Sometimes the amplitude of the oscillations decreased with time; however,

a more robust behavior is desired. The errors in the visual sensor no doubt contributed to this oscillatory behavior. Given the errors of the visual sensor it is easy to see the oscillations are in large part due to the small *comfort zone* selected. Recall the comfort zone, is selected to be only 120mm wide in this design while the error of the sensor in this region is 300mm. A tuning of the behavior should be conducted to eliminate these oscillations beginning with the exploration of expanding the comfort zone.

5.2.2 Robust Ground Truth Data Extraction. This work relied on a manual method for extraction of ground truth data. The frame-by-frame video analysis is required due to camera movement and jitter. A more robust testing environment at AFIT is necessary for more in depth analysis of performance. Without a detailed automated data extraction evaluation of complex swarming algorithms is very cumbersome and difficult.

5.2.3 Robust Robot Identification. The method of determining swarm members in the passive environment was limited to color cones for identification. As is seen in ref Section 4.6 this introduced error into the control system because of the variability in 1) detecting the orange color due to lighting conditions and 2) lack of texture in the orange cones. Thus, a more robust robot identification algorithm is a naturally extension of this work. Many methods from object detection literature are possible candidates including [51].

5.2.4 Formation Tests. Another avenue of extension for this work is the investigation of different swarming tests. Increasing the number of members in the swarm from two warrants investigation. For this to occur, the sensor module will need to be modified to provide a wider field of view. Recall from Section 3.1.2 the field of view of the Videre camera's is only 65.2 degrees for each imager. The effective field of view is smaller still given that any object must be seen in both imagers in order for an estimate of distance to be accurate. Since we limit the robot's sensors to passive only a method needs to be determined to expand this field of view to ensure swarm members do not hit each other during trials.

Once this occurs the performance of a swarm of four to five robots could be conducted to determine the overall performance of Kadrovach's algorithm. One interesting observation in [30] is the concept of stable structures. Namely, Kadrovach found that swarm members

tended to settle in to simple lattice structures. The first such structure would be an equilateral triangle which could be explored using just three robots. Similiarly, in a four robot system the emergence of a diamond formation should be seen.

A final avenue of extension is the exploration of the connection between the fields of *formation control* and *swarm algorithms*. Can we manipulate the parameters of a swarm to exhibit different types of desired formations? What adaptations are necessary to create a swarm which travels in specific formations. This could yield algorithms for unmanned military vehicles in which the operator could select *formation control* or *swarm behaviors* during different portions of the mission.

5.3 Summary

This research presented an adaptation of Kadrovach's Swarm Model [30] to a Pioneer P2-AT8 robot. A real-time visual sensor model was presented using Videre stereo vision cameras , Balch's fast color segmentation [12], and the SRI smallv C++ API package to provide distance estimates to swarm members. This system is then analyzed for performance of the visual sensor, static targets, and dynamic targets. The adaptation of swarm simulation models to real-world swarms is a key component to the next revolution in modern warfare, robots.

1. "AFRL FactSheet". URL `http://www.afrl.af.mil/factsht/afrlfactsheet.htm`.

2. A.F.T. Winfield, O.E. Holland. "The application of wireless local area network technology to the control of mobile robots", 2000.

3. Ali E. Turgut, Hande Celikkant Levent Bayindir, Fatih Gokce and Erol Sahin. *Kobot: A mobile robot designed specifically for swarm robotics research*. Technical report, Middle East Technical University, Ankara, Turkey, November 2007.

4. Ali Emre Turgut, Fatih Gokce, Hande Celikkant and Erol Sahin. *Self-Organized Flocking in Mobile Robot Swarms*. Technical report, Middle East Technical University, Ankara, Turkey, August 2008.

5. Aloimonos, J. and J. Herv'e. "Correspondenceless stereo and motion: Planar surfaces", 1990.

6. Arkin, Ronald C. *Behavior-Based Robotics*. MIT Press, 1998.

7. Balch, Tucker. *Robot Teams*, chapter 4, 93–135. A K Peters, Ltd, 2002.

8. Barnard, Stephen T. and Martin A. Fischler. "Computational Stereo". *ACM Comput. Surv.*, 14(4):553–572, 1982. ISSN 0360-0300.

9. Bonabeau, Eric, Laurent Daghorn, and Pierre Fron. "Scaling in Animal Group-size Distributions". *Proceedings of the National Academy of Science*, volume 96, 4472–4477. April 1999.

10. Braitenberg, V. *Vehicles: Experiments in Synthetic Psychology*. MIT PRESS, 1984.

11. Brooks, Rodney A. "Intelligence without representation". Number 47 in Artificial Intelligence, 139–159. 1991. URL `citeseer.ist.psu.edu/brooks91intelligence.html`.

12. Bruce, James, Tucker Balch, and Manuela Veloso. "Fast and Inexpensive Color Image Segmentation for Interactive Robots". *Proceedings of IROS-2000*. Japan, October 2000.

13. Cassinis, R. "Landmines Detection Methods Using Swarms of Simple Robots". E. Pagello (editor), *International Conference on Intelligent Autonomous Systems 6*. Venice, Italy, 2000. URL `http://frank.ing.unibs.it:8080/papers/IAS2000SW.pdf`.

14. Cassinis, R, G. Bianco, A. Cavagnini, and P. Ransenigo. "Strategies for Navigation of Robot Swarms to Be Used in Landmines Detection". *EUROBOT '99 Third European Workshop on Advanced Mobile Robots*. Zuerich, Switzerland, 1999. URL `http://frank.ing.unibs.it:8080/papers/EURO99SW.pdf`.

15. Chung, Ronald. *Correspondenceless Stereo Vision under General Stereo Geometry*. Technical Report CUHK-ACAE-02-01, Chinese University of Hong Kong, 2002.

16. Crombie, Duncan. *The Examination and Exploration of Algorithms and Complex Behavior to Realistically Control Multiple Mobile Robots*. Master's thesis, Australian National University, 1997. AFIT/GE/ENG/97D-06.

17. Şahin, E., T.H. Labella, V. Trianni, J.-L. Deneubourg, P. Rasse, D. Floreano, L. Gambardella, F. Mondada, S. Nolfi, and M. Dorigo. "SWARM-BOTS: Pattern Formation in a Swarm of Self-Assembling Mobile Robots". A. El Kamel, K. Mellouli, and P. Borne (editors), *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*. Piscataway, NJ: IEEE Press, Hammamet, Tunisia, October 6-9, 2002.

18. Das, Aveek, Rafael Fierro, Vijay Kumar, Jim Ostrowskiand John Spletezer, and Camillo Taylor. "A Framework for Vision Based Formation Control". *IEEE Transactions on Robotics and Automation*, 2001.

19. Dhond, Umesh R. and J.K. Aggarwal. "Structure from Stereo - A Review". *IEEE Transactions of Systems, Man , and Cybernetics*, 19(6):1489–1510, December 1989.

20. Dorigo, M., E. Tuci, R. Groß, V. Trianni, T.H. Labella, S. Nouyan, C. Ampatzis, J.-L. Deneubourg, G. Baldassarre, S. Nolfi, F. Mondada, D. Floreano, and L.M. Gambardella. "The SWARM-BOTS project". E. Şahin and W. Spears (editors), *Proceedings of the First International Workshop on Swarm Robotics at SAB 2004*, volume 3342 of *Lecture Notes in Computer Science*, 31–44. Springer Verlag, Berlin, Germany, 2004.

21. Dudek, G., M. Jenkin, E. Milios, and D. Wilkes. "A Taxonomy for Multi-Agent Robotics". *Autonomous Robots*, 3:375–397, 1996.

22. Dudek, Gregory and Michael Jenkin. *Computational Principles of Mobile Robotics*. Cambridge University Press, Cambridge, United Kingdom, 2000.

23. Fredslund, Jakob and Maja J Mataric. "Robot formations using only local sensing and control". *In Proceedings, International Symposium on Computational Intelligence in Robotics and Automation (IEEE CIRA 2001*, 308–313. 2001.

24. Groß, R., E. Tuci, M. Dorigo, M. Bonani, and F. Mondada. "Object Transport by Modular Robots that Self-assemble". *Proc. of the 2006 IEEE Int. Conf. on Robotics and Automation*, 2558–2564. IEEE Computer Society Press, Los Alamitos, CA, 2006.

25. Hayes, A. T. and P. Dormiani-Tabatabaei. "Self-Organized Flocking with Agent Failure: Off-Line Optimization and Demonstration with Real Robots". *Proc. of the 2002 IEEE Int. Conf. on Robotics and Automation IROS-02*, 3900–3905. 2002.

26. Hayes, Adam T., Alcherio Martinoli, and Rodney M. Goodman. "Distributed odor source localization". *IEEE Sensors*, 2:260–271, 2002.

27. Hayes, Adam T., Alcherio Martinoli, and Rodney M. Goodman. "Swarm robotic odor localization: Off-line optimization and validation with real robots". *Robotica*, 21(4):427–441, 2003. ISSN 0263-5747.

28. Horn, Berthold K.P. *Robot Vision*. MIT Press, 1986.

29. Jones, Graeme A. "Constraint, Optimization, and Hierarchy: Reviewing Stereoscopic Correspondence of Complex Features". *Computer Vision and Image Understanding: CVIU*, 65(1):57–78, 1997. URL `citeseer.nj.nec.com/article/jones97constraint.html`.

30. Kadrovach, Brian A. *A Communications Modeling System for Swarm-based Sensors*. Ph.D. thesis, Air Force Institute of Technology, 2003.

31. Klette, Reinhard, Karsten Schlns, and Andreas Koschan. *Computer Vision*. Springer, Singapore, 1998.

32. Konolige, Kurt. "ARIA Reference Manual", 2002.

33. Konolige, Kurt and David Beymer. "SRI Small Vision System Users Manual: Software version 3.0a", 2003.

34. Lane, R.A. and N.A. Thacker. "Stereo Vision Research: An Algorithm Survey". URL `citeseer.ist.psu.edu/lane96stereo.html`.

35. McLauchlan, Philip F. "Gandalf: The Fast Computer Vision and Numerical Library". URL `http://gandalf-library.sourceforge.net/tutorial/report/node102.html`.

36. McLurkin, James. *Stupid Robot Tricks: A Behavior-Based Distributed Algorithm Library for Programming Swarms of Robots*. Master's thesis, Massachusetts Institute of Technology, 2004.

37. McLurkin, James. "Dynamic Task Assignment in Robot Swarms". *Proceedings of Robotics: Science and Systems*, 2007–214838. 2005.

38. McLurkin, James. *Analysis and Implementation of Distributed Algorithms for Multi-Robot Systems*. Ph.D. thesis, Massachusetts Institute of Technology, 2008.

39. McLurkin, James. "Measuring the Accuracy of Distributed Algorithms on Multi-Robot Systems with Dynamic Network Topologies". *Proceedings of the The 9th International Symposium on Distributed Autonomous Robotic Systems*. November 2008.

40. McLurkin, James and Jennifer Smith. "Distributed algorithms for dispersion in indoor environments using a swarm of autonomous mobile robots". *in 7th International Symposium on Distributed Autonomous Robotic Systems (DARS*. 2004.

41. Michael W. Trahan, et al. "Swarms of UAVs and Fighter Aircraft". *Proceedings of the Second International Conference on Nonlinear Problems in Aviation and Aerospace*, volume 2, 745–752. 1998.

42. Mondada, F., D. Floreano, A. Guignard, J.-L. Deneubourg, L. Gambardella, S. Nolfi, and M. Dorigo. *Search for Rescue: an Application for the SWARM-BOT Self-Assembling Robot Concept*. Technical report, LSA2 - I2S - STI, Swiss Federal Institute of Technology, Lausanne, Switzerland, September 2002.

43. Nouyan, S. and M. Dorigo. "Chain Based Path Formation in Swarms of Robots". volume 4150, 120–131. 2006.

44. Reynolds, Craig W. "Flocks, Herds, and Schools: A Distributed Behavioral Model". Maureen C. Stone (editor), *Computer Graphics (SIGGRAPH '87 Proceedings)*, volume 21, 25–34. July 1987.

45. Rothermich, Joseph A., M. Ihsan Ecemis, and Paolo Gaudiano. "Distributed Localization and Mapping with a Robotic Swarm". *Swarm Robotics*, 58–69. 2004.

46. Russell, Stuart and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 1995.

47. Scharstein, Daniel and Richard Szeliski. "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms". *International Journal of Computer Vision*, 47(1):7–42, May 2002.

48. Tanimoto, Stephen L. *The Elements of Artificial Intelligence*. Computer Science Press, 1987.

49. Trianni, V., S. Nolfi, and M. Dorigo. "Cooperative Hole Avoidance in a *Swarm-bot*". *Robotics and Autonomous Systems*, 54(2):97–103, 2006.

50. Vehicle, Unmanned Air. "TECHNICAL DOCUMENT 3042 September 1998".

51. Viola, Paul and Michael Jones. "Robust Real-time Object Detection". *International Journal of Computer Vision*. 2001.

Zachary C. Gray attended Mascoutah High School in Mascoutah, Illinois. He accepted an Air Force ROTC scholarship enrolled in the University of Florida, Gainesville, Florida pursuing undergraduate degrees in Electrical Engineering and Computer Engineering. While at Florida, he was awarded the Communications Group William H. Everitt Award and was selected as a Undergraduate Scholar. He graduated with Honors in May 2002 earning degrees in Electrical Engineering and Computer Engineering. Upon Graduation, Zachary accepted his commission in the United States Air Force on May 5^{th}, 2002 as an Air Force ROTC distinguished graduate. In August of 2002, he entered active duty at Wright Patterson Air Force Base, Ohio to pursue his masters degree at the Air Force Institute of Technology.

OMB No. 0704-0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
21–03–2005	Master's Thesis	Aug 2002 — Mar 2009

4. TITLE AND SUBTITLE

Communication Free
Robot Swarming

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Zachary C. Gray, Capt, USAF

5d. PROJECT NUMBER

JON09-219

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GCE/ENG/09-03

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Force Research Laboratory
ATTN: Dr. Jacob Campbell
2241 Avionics Circle
Wright-Patterson AFB, OH 45433
((937) 255-6127 x4154 jacob.campbell@wpafb.af.mil)

10. SPONSOR/MONITOR'S ACRONYM(S)

AFRL/RYRN

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approval for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

As the military use of unmanned aerial vehicles increases, a growing need for novel strategies to control these systems exists. One such method for controlling many unmanned aerial vehicles simultaneously is the through the use of swarm algorithms. This research explores a swarm robotic algorithm developed by Kadrovach implemented on Pioneer Robots in a real-world environment. An adaptation of his visual sensor is implemented using stereo vision as the primary method of sensing the environment. The swarm members are prohibited from explicitly communicating other than passively through the environment. The resulting implementation produces a communication free swarming algorithm. The algorithm is tested for performance of the visual sensor, performance of the algorithm against stationary targets, and finally, performance against dynamic targets. The results show expected behavior of the swarm model as implemented on the Pioneer robots providing a foundation for future research in swarm algorithms.

15. SUBJECT TERMS

Artificial intelligence; Swarm Robotics; Computer Vision; Stereo Vision; Robotics; Unmanned Vehicles

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dr. Gilbert L. Peterson
U	U	U	UU	99	19b. TELEPHONE NUMBER *(include area code)* (937)255–3636x4281; gilbert.peterson@afit.edu