A Practical Search with Voronoi Distributed Autonomous Marine Swarms

by

Nicholas Craig Evans
B.S.E. 2014, Electrical Engineering, University of Alaska Anchorage

Submitted to the Department of Mechanical Engineering and the Joint Program in Applied Ocean Science & Engineering in partial fulfillment of the requirements for the degree of

Master of Science in Mechanical Engineering

at the

Massachusetts Institute of Technology

and the

Woods Hole Oceanographic Institution

September 2022

Author . . . . . .
Department of Mechanical Engineering, MIT
Applied Ocean Science & Engineering, WHOI
August 5th, 2022

Certified by . . . . . .
Dr. Michael Benjamin
Principal Research Scientist Mechanical Engineering, MIT
Thesis Supervisor

Accepted by . . . . . .
Dr. Nicolas Hadjiconstantinou
Professor of Mechanical Engineering, MIT
Chair, Department Committee on Graduate Students

Accepted by . . . . . .
Dr. David Ralston
Associate Scientist with Tenure, Applied Ocean Physics & Engineering, WHOI
Chair, Joint Committee for Applied Ocean Science & Engineering

A Practical Search with Voronoi Distributed Autonomous Marine Swarms

by

Nicholas Craig Evans

Submitted to the Department of Mechanical Engineering and the Joint Program in Applied Ocean Science and Engineering on August 5th, 2022 in partial fulfillment of the requirements for the degree of Master of Science in Mechanical Engineering

Abstract

The search for underwater threats in littoral regions is a problem that has been researched for nearly a century. However, recent developments in autonomy and robotics have made this issue more complex. The advent of capable autonomous underwater vehicles presents a 21^{st} century flare to this traditional problem. These vehicles can be smaller, quieter, and expendable. Therefore, new methods and tactics used to detect and track these vehicles are needed. The use of a swarm of marine robots can increase the likelihood of uncovering these threats. This thesis provides various Voronoi partition-based methods to autonomously control a swarm of identically capable autonomous surface vessels in a limited coverage and tracking problem. These methods increase the probability of interdiction of an adversary vehicle crossing a defined region. The results achieved from Monte Carlo simulations demonstrate how different protocols of swarm movement can improve detection probability as compared to a stationary swarm provided the detection capability does not change. The swarm control algorithms are employed on Clearpath Heron USVs to validate the autonomy algorithms.

Thesis Supervisor: Dr. Michael Benjamin
Title: Principal Research Scientist Mechanical Engineering

Acknowledgments

Any success I achieved throughout my time in the MIT-WHOI Joint Program can be attributed to the fantastic support and examples around me. I cannot finish my time here at MIT without extending my sincere gratitude to the Navy, the people in LAMSS and the AUV lab, Dr. Michael Benjamin, and of course my family.

I am so grateful the Navy provided me the the opportunity I had to grow and learn here at MIT. The Navy has never ceased to provide me with opportunities to get better, for which I am truly thankful. Thus, I would like to thank the Navy for its investment in me and my growth through this period. I feel indebted to the Navy for this opportunity, for which, I literally am.

I am in constant awe at the capability, motivation, and character of the great people I worked with here at MIT. No group of people embodies these traits more than the members of LAMSS and the AUV lab. I am exceedingly thankful for the countless hours of help that I received from Tyler, Supun, and Oscar. I could not have done it without you all. Thanks to Ray, Mark, Nick, and all the Mikes for the help and positive community. Henrik, thank you for inviting me into your lab during an especially unique time of human history. I have grown from knowing you all, and I aspire to be as selfless as the members of this team.

Here at MIT it can be commonplace to interact with highly accomplished and extraordinary faculty; however, it is rare that one of these people can truly make you feel like a respected colleague. Not only does Mike make you feel like you are part of the team, he treats you like a peer. Throughout my 16 years in the military, I have had the opportunity to work with and for many great leaders. I can honestly say that I have never seen a leader who treats peers, subordinates, or random curious strangers with such genuine kindness and respect as Mike Benjamin. I did not expect the style of my personal leadership goals to be shifted so greatly during my time here, but thanks to Mike they have. Thank you, Mike, for all the help and guidance, but thank you more for the unknowing example you set.

Through every significant achievement I have accomplished, there has been one person that I have continuously relied upon for inspiration, advice, and encouragement. I will never be able to thank you enough, Wynter, for being that constant presence of good that I have needed. I do not have the words to accurately express my gratitude for you; therefore, I will simply say I love you and thanks. Lilly, Wyatt, and Jasper, you have enriched my life beyond measure. Thank you for teaching me the overwhelming happiness of unconditional love, and of course, for the countless Minecraft facts that now inhabit my brain.

Contents

List of Figures

List of Tables

1 Introduction

> *"There is, one knows not what sweet mystery about this sea, whose gently awful stirrings seems to speak of some hidden soul beneath"* -Herman Melville in *Moby Dick*

Since the onset of submarine warfare in WW1, control of the seas has extended below the surface. The advances in sub surface threats and the effectiveness of submarine warfare throughout the 20th century has impacted many parts of society. From the films of the early 1900s that depicted the threat of German U-Boats to blockbuster movies centered around the submarines of the Cold War, the dangers of submarine warfare have been continuously imprinted on the public psyche. Further, the reality that a sub-surface threat could be lurking just off one's own shore has permeated generations. However, in many ways, the advances of the 21st century are still not understood or appreciated by the public. Technological advances in undersea robotics and artificial intelligence have brought about a new age in subsurface warfare and defense. These advances both necessitate an evolution in defense strategy and give rise to the technology that makes such an evolution possible.

Throughout history every advancement in weaponry or military technology has necessitated that any competing party also advance. In order to remain competitive, the counterparty must either adopt the new technology themselves or better the original advancement (Turchin *et al.*, 2021). Therefore, as the threat of autonomous underwater vehicles in coastal defense become a reality, the need for an advanced counter technol-

24. Turchin, P. *et al.*, 2021, Rise of the war machines: Charting the evolution of military technologies from the Neolithic to the Industrial Revolution.

ogy is required. The use of autonomous defense vehicles working together to detect and track sub surface threats is needed in order to counter the risk of AUVs.

This thesis evaluates and improves the effectiveness of a swarm of autonomous surface vessels in a detection limited area coverage and tracking scenario, which is fundamental to the harbor defense problem. Specifically, this thesis proposes and evaluates various Voronoi based deployment strategies to improve the swarm's probability of detecting a threat crossing an established region. These Voronoi based methods are evaluated through a series of Monte Carlo simulations to evaluate their effectiveness at detecting a threat crossing the swarm occupied region. Moreover, the ability of each method to produce an actionable estimated target speed and heading is examined. Lastly, the methods developed for this thesis are employed on Clearpath M300 USVs in a hardware in the loop simulation to demonstrate the deployability of this thesis's algorithms.

1.1 Motivation

Control and defense of the seas is an age-old problem. As soon as the first marine vessel created a military advantage, the ability to detect and defend against a marine threat became an essential element of harbor safety. The technology and capabilities that are available in the 21st century have made this an even more difficult challenge. Offensive marine vessels have continued to evolve. Now the threat of unmanned autonomous vehicles (UxVs) capable of intelligence gathering or munitions deployment must be accounted for in harbor defense. As the capabilities of AUVs become more advanced, the need to effectively detect and track AUVs in littoral regions is paramount.

The detection and tracking of underwater threats has been thoroughly researched since the first vessel was sunk by a submarine in WWI. Historically, submarines have

been countered by both surface vessels and submarines (Newbolt, 1919). While submarines are still countered from the surface and below, the methods have evolved significantly throughout the years. During WWI, naval warfare was largely surprised by the onset of German U-Boats. Tactics for anti-submarine warfare (ASW) consisted of methods of evasion or attacking the submarine when it surfaced (Newbolt, 1919). However, as naval warfare and submarines advanced, the methodology of ASW did as well. The use of piezoelectric transducers in active sonar became a go to tool for submarine detection as well as radar. Following WWII, ASW evolved to include air, surface, and submerged assets. These assets brought in the capability of radar, passive acoustics, active acoustics, as well as other electronic warfare measures (ESM). However, the task of locating and tracking submarines is still an extreme challenge.

The 21st century has brought about the most capable underwater threats. While the quantity and quality of these threats have increased, the US Navy's ASW platforms have declined post Cold War (Ketter, 2004). The overall cost of ASW tools and platforms has caused a draw down in US Navy capabilities (Ketter, 2004). Therefore, a cost-effective tool capable of combating the 21st century threats of today is needed to execute the ten steps (Table 1.1) of anti-submarine warfare as defined by Captain William J. Toti, U.S. Navy (Retired). This thesis provides a methodology that can be used to enable priority 5 (Toti, 2014).

1.2 Literature Review

The principles of underwater surveillance are vastly researched and continues to be a priority research area. In fact, Terracciano et al. explains that underwater surveillance continues to be a top priority on a global scale with interested parties including the

15. Newbolt, H. J., 1919, *Submarine and anti-submarine*
14. Ketter, T., 2004, Anti-Submarine Warfare in the 21st Century.
23. Toti, W. J., 2014, The Hunt for Full-Spectrum ASW.

1	Create conditions where an adversary chooses not to employ submarines
2	Defeat submarines in port
3	Defeat submarines' shore-based command and control (C2) capability
4	Defeat submarines near port, in denied areas
5	Defeat submarines in choke points
6	Defeat submarines in open ocean
7	Draw enemy submarines into ASW "kill boxes", to a time and place of our choosing
8	Mask our forces from submarine detection or classification
9	Defeat the submarine in close battle
10	Defeat the incoming torpedo

Table 1.1: The Ten Threads of Full-Spectrum ASW (Toti, 2014).

US Navy and NATO (Terracciano *et al.*, 2020). Throughout history many methods have been used to search the underwater environment. These approaches have ranged from manned single submarine search missions to teams of autonomous vehicles (Ferri *et al.*, 2017). In this section the current schemes of underwater surveillance and the use of robot teams in search problems are reviewed.

Terracciano et al. and Ferri survey the current state of underwater surveillance using robots and categorizes the common methods employed (Terracciano *et al.*, 2020)(Ferri *et al.*, 2017). Both authors discuss the importance of robots and unmanned vehicles in underwater surveillance. AUVs are commonly used in underwater surveillance and ASW type missions due to their ability to traverse the water column, which permits effective use of the acoustic environment. However, using AUVs for surveillance missions comes with many limitations. The underwater vehicle is severely limited in communications and naturally must exist in the harsh subsea environment (Ferri *et al.*, 2017). These are two areas mitigated by using USVs. In fact, Terracciano et al. notes the promising use of expanding the search mission by using teams of robots on the surface (Terracciano *et al.*, 2020).

21. Terracciano, D. S. *et al.*, 2020, Marine Robots for Underwater Surveillance.
11. Ferri, G. *et al.*, 2017, Cooperative Robotic Networks for Underwater Surveillance: an Overview.

The use of USVs in underwater surveillance is an active area of research. Moreover, the use of USVs in littoral surveillance missions is an increasingly popular research field (Healey *et al.*, 2007)(Simetti *et al.*, 2010)(Vencatasamy *et al.*, 2018). Healy evaluates the capability of various unmanned craft and promotes USVs as a highly capable surveillance vessel in areas where stealth is not essential (Healey *et al.*, 2007). Harbor defense and littoral operations are a key area where ASVs can be an effective surveillance tool. Algorithms for establishing secure areas of a harbor using vehicles operating on the 2D plane will be essential in the future of harbor defense (Healey *et al.*, 2007)(Vencatasamy *et al.*, 2018). This thesis focuses on the development of these algorithms and their applicability to USV teams.

The way many harbor defense scenarios are researched flow from a classical set of search problems. Robin and Lacroix define these problems as coverage or tracking problems for which the standard breakdown can be seen in fig. 1.1 (Robin & Lacroix, 2015). However, Robin and Lacroix explain there is a subset of problems that focus both on coverage and tracking. The simultaneous coverage and tracking problem is more complex and not as well researched. Pimenta et. al explored the problem of simultaneous coverage and tracking. In the paper titled *Simultaneous Coverage and Tracking (SCAT) of Moving Targets with Robot Networks*, Pimenta et al. demonstrates the effectiveness of covering a region using centroidal Voronoi tessellations (CVT) (Pimenta *et al.*, 2009). Once the area is covered using CVTs, the robots are set to track any intruder entering its Voronoi region. However, Pimenta et al. did not explore the effectiveness of a swarm's coverage when the region cannot be fully occupied with sensor coverage.

13. Healey, A. J. *et al.*, 2007, Collaborative Unmanned Systems for Maritime and Port Security Operations.
19. Simetti, E. *et al.*, 2010, Towards the Use of a Team of USVs for Civilian Harbour Protection: USV Interception of Detected Menaces.
25. Vencatasamy, K. *et al.*, 2018, *Secure a Zone from Intruders with a Group Robots*
18. Robin, C. *et al.*, 2015, Multi-robot target detection and tracking: taxonomy and survey.
17. Pimenta, L. C. A. *et al.*, 2009, *Simultaneous Coverage and Tracking (SCAT) of Moving Targets with Robot Networks*

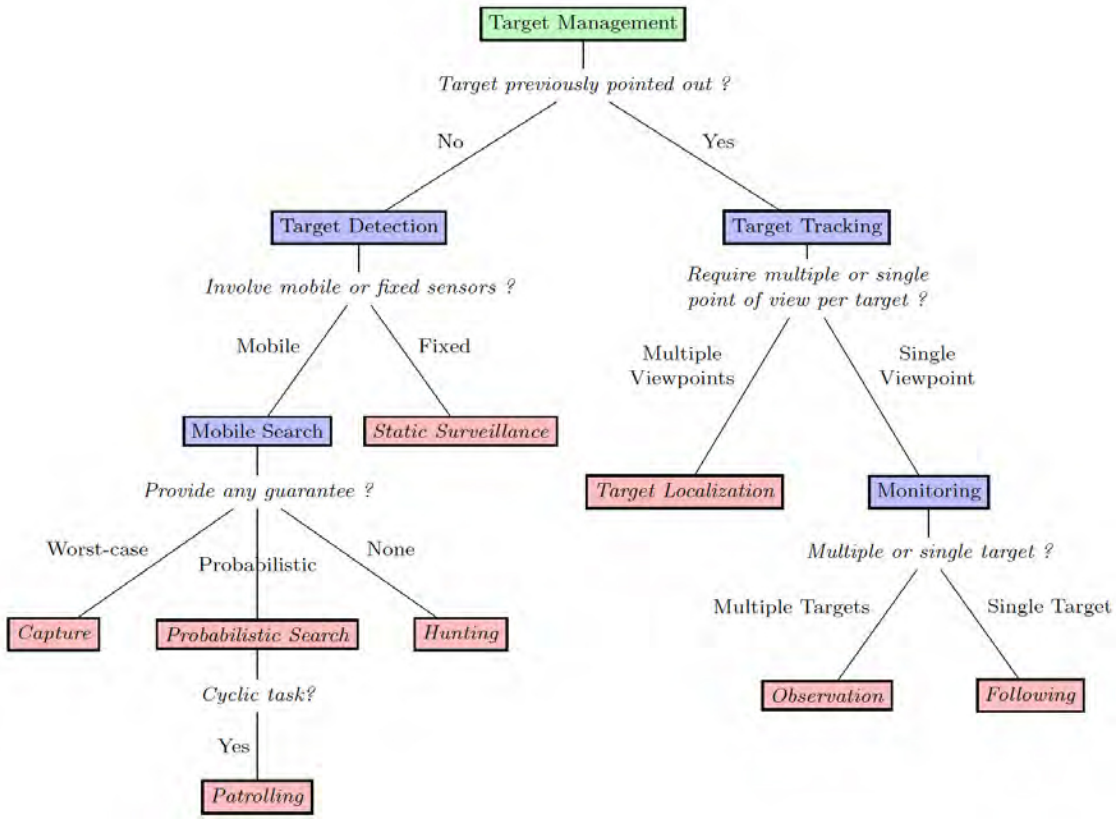


Figure 1.1: Target Management breakdown (Robin & Lacroix, 2015)

.

(Ben Slimane & Tagina, 2021)(D' Acunto, 2020)(Cortes *et al.*, 2004) have all discussed the use of Voronoidal dispersion to cover a region. Further, (Guruprasad & Ghose, 2011)(Teruel *et al.*, 2019)(XIONG *et al.*, 2019)(Pimenta *et al.*, 2009) have shown the effectiveness of CVTs for robotic dispersion in a cover problem. In these various implementations of Voronoi partitioning, it assumed that there are adequate sensors within the distribution to cover

2. Ben Slimane, N. *et al.*, 2021, Proposition of a Distributed Voronoi Partitioning Approach Enhanced with a Dispersion Phase for a Multirobot System.
9. D' Acunto, M., 2020, Optimized Dislocation of Mobile Sensor Networks on Large Marine Environments Using Voronoi Partitions.
8. Cortes, J. *et al.*, 2004, Coverage control for mobile sensing networks.
12. Guruprasad, K. R. *et al.*, 2011, Automated Multi-Agent Search Using Centroidal Voronoi Configuration.
22. Teruel, E. *et al.*, 2019, A distributed robot swarm control for dynamic region coverage.
27. XIONG, C. *et al.*, 2019, Path planning of multiple autonomous marine vehicles for adaptive sampling using Voronoi-based ant colony optimization.

the desired area. Further, once the members are distributed, the sensors are considered omnipotent in their assigned Voronoi partition. In this thesis, the cover and tracking problem is examined when the search area cannot be completely covered without gaps in detection area. Thus, holes exist in the network that cannot be adequately covered at every discrete time step. Further, this thesis investigates and improves a swarm's ability to detect and track a threat in a region when the ability of any one vehicle to detect the threat is probabilistic based on range to the target.

1.3 Contributions of this Thesis

1. The effectiveness of distributing vehicles in a Centroidal Voronoi Tessellation over a constant density region in "detecting" an adversary vehicle in limited coverage and tracking scenario is quantified, examined, and improved through the use of Monte Carlo simulation testing.
2. A probabilistic sensor is modeled to reflect real world limitations of underwater detection to examine the effectiveness of the proposed search methods and is deployed alongside a Kalman filter application to develop tracking solutions with limited data.
3. The various search methods are deployed on robotic vehicles to demonstrate the feasibility of the proposed methods, validate the data from simulation, and demonstrate the ability to deploy the search methods on autonomous marine vehicles.

1.4 Thesis Overview

Chapter 2 of this thesis provides the background information on the development of Centroidal Voronoi Tessellations, MOOS-IvP and CVTs, and coverage and search methodology. Chapter 3 defines the problem that will be examined as well as the applications that were developed for this thesis. Chapter 4 provides the results and analysis

for the elements proposed in Chapter 3. Lastly, Chapter 5 provides conclusions and avenues for future work.

2 Background

This chapter explains the function and methodology of CVTs, as well as the the MOOS-IvP implementation of the Lloyd algorithm. Then the principles of coverage and tracking are discussed. Lastly, the basics of active sonar and the basis of probabilistic detection are summarized.

2.1 Voronoi Application

2.1.1 Voronoi Diagrams

A voronoi diagram is geometrical solution to partitioning a space based off each node's nearest neighbor. In a Voronoi decomposition, a region is divided into cells occupied by each node with the cell boundaries defined by half the Euclidean distance to its nearest neighbors(Berg *et al.*, 1997). Figure 2.1 demonstrates the construct of a Voronoi diagram (Aurenhammer, 1991). In this diagram it can be seen that the individual wall of each partition is defined by the perpendicular line that is drawn at half the Euclidean distance to its closest neighbor. Thus, the region with n nodes is divided into n regions based on the node locations. Voronoi diagrams are widely researched in the computational

5. Berg, M. *et al.*, 1997, *Computational Geometry*
1. Aurenhammer, F., 1991, Voronoi Diagrams—a Survey of a Fundamental Geometric Data Structure.

geometry domain and are applicable over large set of mathematical, computer science, and natural science domains (Aurenhammer, 1991)(Du *et al.*, 1999).

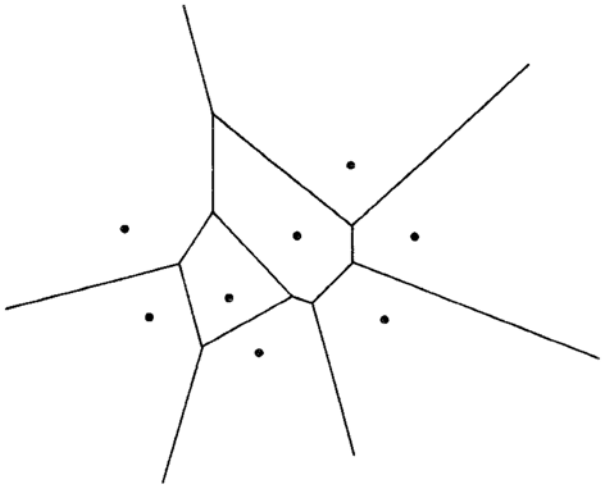


Figure 2.1: Voronoi Diagram (Aurenhammer, 1991).

Computing the Voronoi diagram of a region Q with nodes P can be seen in the algorithm 1, derived from the explanation in *Computational Geometry* by Berg et al. (Berg *et al.*, 1997).

Data: A set $P = Pi, ..., Pn$ of point sites in the plane Q

Result: The Voronoi diagram $V(P)$

for *Each Point* k *in* P **do**

Draw perpendicular line at half the Euclidean distance from Pk to Pn

Retain all line segments that are closest to point Pk within Q

Define Pk's Voronoi cell vertices as the intersection points of all the closest line segments.

end

Algorithm 1: Voronoi Diagram Algorithm (Berg *et al.*, 1997)

10. Du, Q. *et al.*, 1999, Centroidal Voronoi Tessellations: Applications and Algorithms.
5. Berg, M. *et al.*, 1997, *Computational Geometry*

2.1.2 Centroidal Voronoi Tessellation and the Lloyd Algorithm

Centroidal Voronoi Tessellations is a form of Voronoi diagram where the generators are located in the center of the cell. A Voronoi diagram for ten generators in a square is shown in Fig. 2.2. Here the circles represent the center of mass of the cell and the point is the generator. The generators need to be placed at the center of mass of the cell it occupies to be a CVT. The Lloyd algorithm provides an iterative process to establish a CVT for a given set of generators. Du et al. describes the Lloyd algorithm as a three step process seen in table 2.1 (Du *et al.*, 1999). Fig. 2.3 shows the initial and final locations for the generators in the ten point example following the completion of the Lloyd Algorithm (Du *et al.*, 1999).

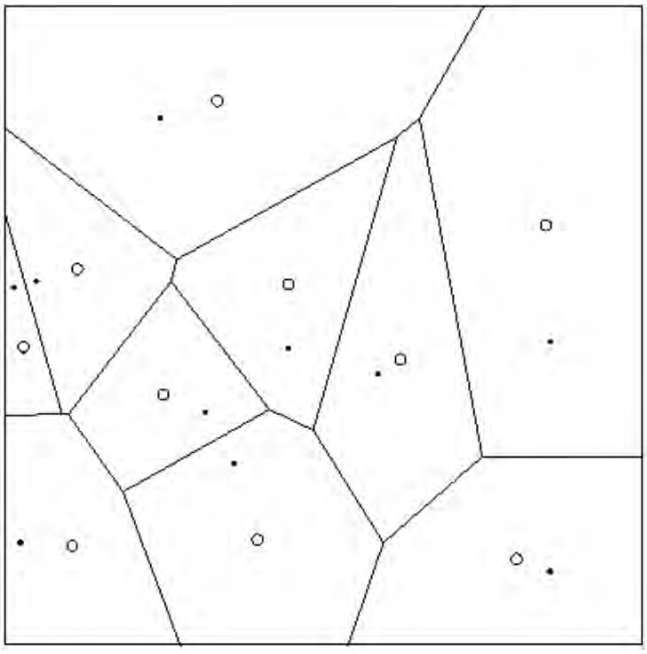


Figure 2.2: Voronoi Diagram with dots as initial generators and center of mass denoted by open circles (Du *et al.*, 1999)

1	Determine the Voronoi Diagram.
2	Calculate center of mass for each Voronoi Cell, then use calculated center in step 1.
3	Repeat the previous steps until completion parameter met.

Table 2.1: Lloyds Algorithm (Du *et al.*, 1999).

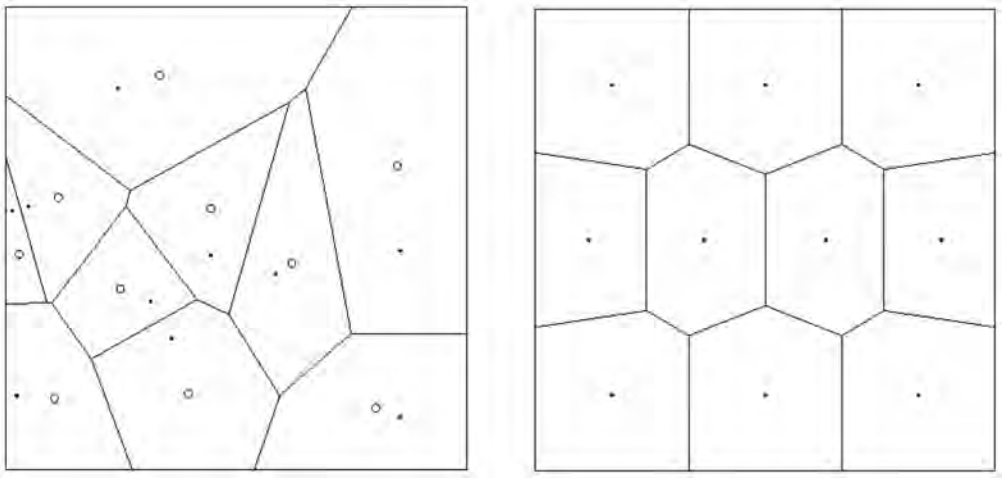


Figure 2.3: Initial and final Voronoi diagram using the Lloyd algorithm (Du *et al.*, 1999)

2.2 MOOS-IvP

MOOS-IvP is a software suite for robot autonomy. MOOS-IvP is a combination of MOOS (Mission Oriented Operating Suite) a middle-ware that institutes a publish and subscribe architecture and IvP (Interval Programming), which is a multi objective optimizer and solver. IvP implements pHelmIvP (Helm) that produces an objective function that is a solution to a multi objective problem. The helm includes different autonomous behaviors that provide input to the multi-objective solver.

MOOS is a publish and subscribe ecosystem that operates a MOOSDB (MOOS Database), which is the central hub that hosts all published variables. The architecture can be seen in fig. 2.4 (Newman, 2003). The MOOSDB is the central application that processes all

16. Newman, P., 2003, MOOS - Mission Orientated Operating Suite.

mail and can be accessed by any application. This allows applications to be designed for a specific purpose, and the MOOSDB will serve the application based off of its subscriptions. Each application can publish and subscribe to any variables contained in the MOOSDB. Therefore, special utility applications can run in the background and publish an output variable that another application is dependent on. This provides an ecosystem of MOOS applications and a MOOSDB that can work together to execute the desired mission.

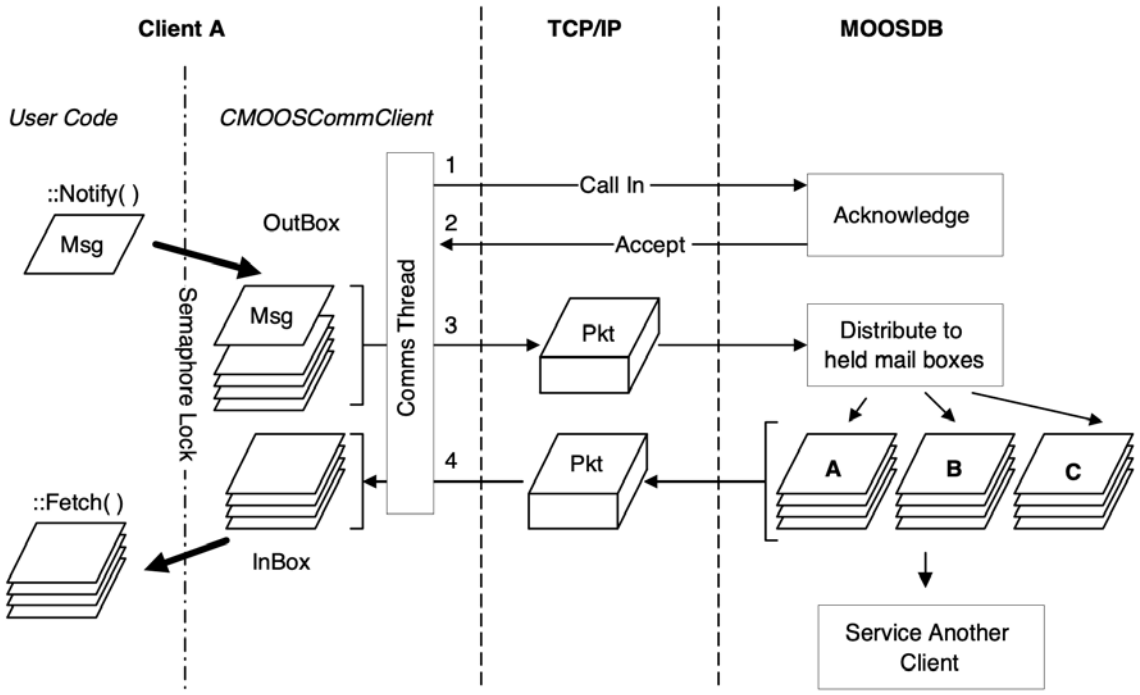


Figure 2.4: MOOS Architecture (Newman, 2003)

Building upon the foundations of MOOS, IvP implemented a special case for the Helm as pHelmIvP. The IvP helm then serves as a solver for multi objective optimization problems. This allows multiple behaviors to produce specific objective functions based on each of their individual decision criteria for both course and speed. Once the IvP solver receives the independent objective functions from each behavior and associated priority weights, it then optimally solves for the desired speed and heading. An example

of a course and speed objective function that has been optimized with the pHelmIvP can be seen in fig. 2.5. This objective function is produced from the waypoint behavior with the speed peak set to 4 m/s and heading peak set to 180 degrees. The IvP Helm then publishes the heading and speed requirements for the robot controller. This process is done recursively at a defined time interval; thereby, continually updating parameters needed for individual behaviors to produce objective functions. The IvP Helm then solves the optimization problem to publish speed and heading decisions every iteration. The information flow path for the IvP Helm can be seen in fig 2.6 (Benjamin *et al.*, 2009).

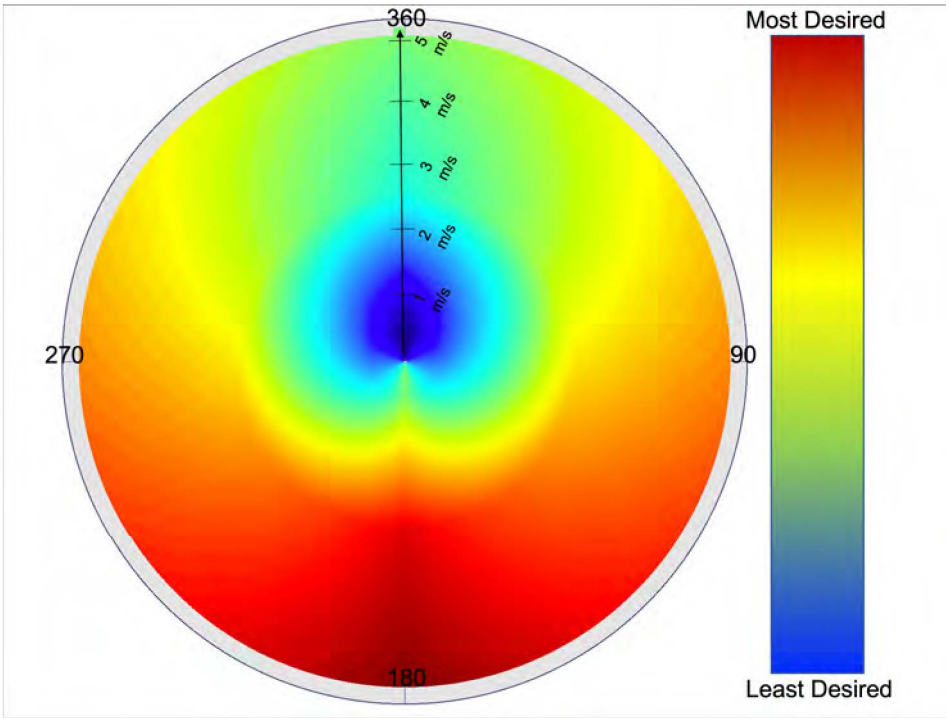


Figure 2.5: Example Objective Function

3. Benjamin, M. R. *et al.*, 2009, An Overview of MOOS-IvP and a Brief Users Guide to the IvP Helm Autonomy Software.

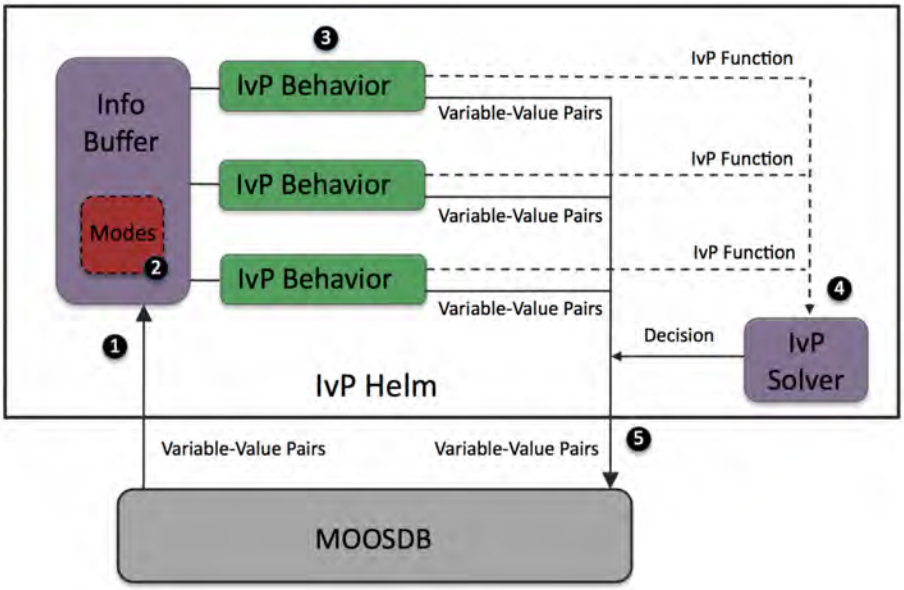


Figure 2.6: IvP Helm Architecture (Benjamin *et al.*, 2009). 1. The MOOSDB sends all required mail to IvP Helm's Info Buffer. 2. The active Modes are determined from Info Buffer variable-value pairs as defined in the behavior file. 3 Appropriate IvP Behaviors are activated and produce Individual IvP functions. 4. Each IvP function is delivered to the IvP Solver for Optimization. 5. MOOS variable-value pairs are published to the MOOSDB as required by the IvP Behaviors and the IvP Solver. This process is repeated at the designated IvP Helm frequency.

MOOS-IvP is delivered with an open-source set of autonomous behaviors and applications, which can be employed in self designed missions. Further, the MOOS-IvP package also provides a convenient way to extend the autonomous capabilities of the software by creating new helm behaviors or MOOS applications. This thesis makes use of this incredibly powerful tool to generate autonomous helm behaviors and MOOS applications. Moreover, the MOOS-IvP package provides a method to run large scale simulations and retain the data, which can be efficiently employed on hardware for follow on testing. These tools allow this thesis to provide results that are transferable to real world robotic systems.

2.2.1 MOOS-IvP CVT Implementation

As discussed earlier, MOOS-IvP can be easily extended to provide autonomy for user specific priorities. One such extension is the moos-ivp-swarm tree. The swarm tree was written by Dr. Michael Benjamin and provides useful tools for operating robotic systems in swarms. The swarm tree includes the tools to disperse a set of vehicles in a polygon region in a CVT dispersion. This is done using one MOOS application pProxonoi and one behavior BHV_Voronoi. These applications work in tandem to execute a version of the Lloyd algorithm with robotic platforms as the generators. The function of each application is discussed below. The combination of the pProxonoi app and the Voronoi behavior serve as the foundation for the analysis performed in this thesis.

pProxonoi

The pProxonoi app manages and provides the individual vehicle's tessellation. This is done with a combination of geometry tools available in the MOOS-IvP base code. MOOS-IvP has a robust set of methods to handle polygons as objects. Therefore, pProxonoi is initialized with the entire state space as a "proxonoi" polygon. Then each vehicle broadcasts its location to all vehicles within communication range. As each vehicle receives the location information of others in the swarm, it "chops" its proxonoi polygon at half the Euclidean distance to its nearest neighbors. Thus, each vehicle generates its own individual Voronoi cell within the operation region. The pProxonoi app then posts the voronoi cell polygon to its associated MOOSDB for the behavior BHV_Voronoi to respond to. pProxonoi executes this process iteratively on an interval designated by the app user. Each pProxonoi iteration is conducted using the vehicles updated location and locations of its neighbors. An example of the initial Voronoi diagram generated from each vehicle's self-generated proxonoi polygon can be seen in fig. 2.7.

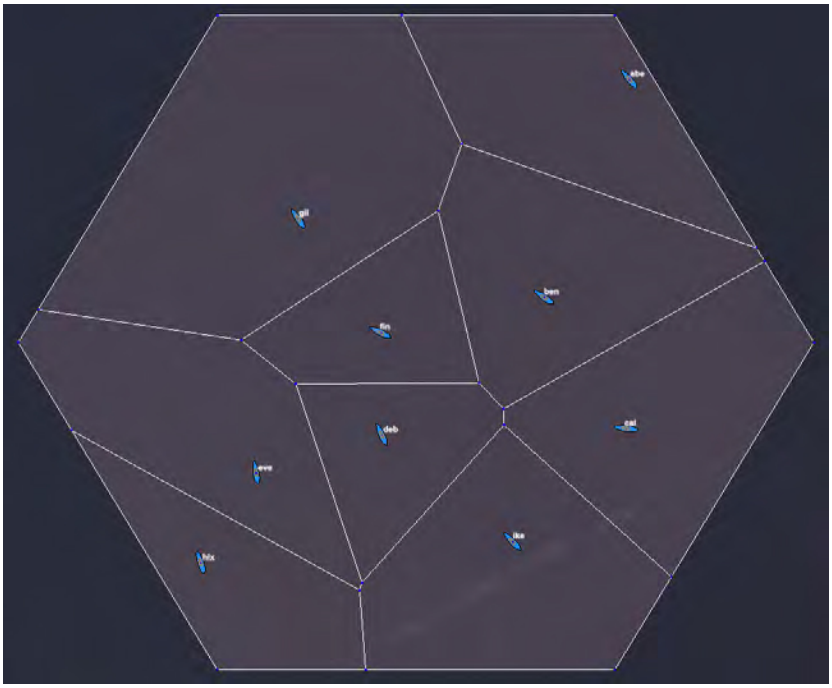
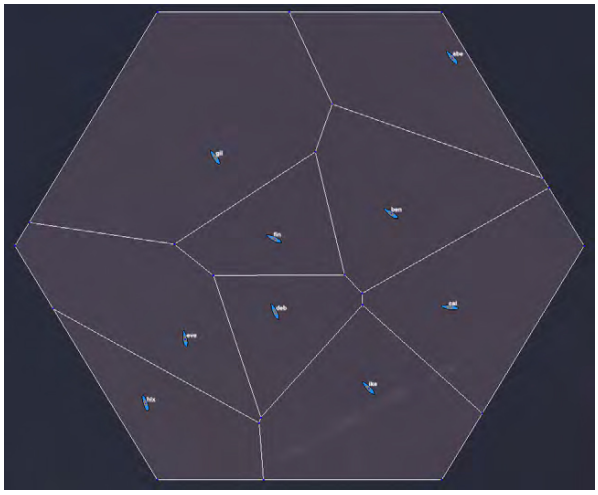


Figure 2.7: Voronoi diagram using individual vehicle generated proxonoi polygons

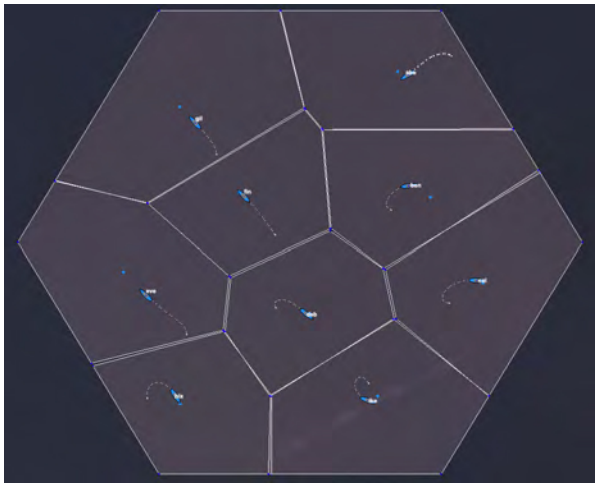
Voronoi Behavior

The Voronoi behavior subscribes to the proxonoi polygon definition which is the output of the pProxonoi application. Once the Voronoi behavior ingests a new polygon region it finds the centroid of that region and produces an objective function directing the vehicle to the centroid. As the vehicle moves towards the centroid, the pProxonoi app continues to generate new proxonoi polygons based off the vehicle's current position. Therefore, the voronoi behavior reprocesses the new proxonoi region and navigates towards the new centroid. This process is repeated until the vehicle enters a "capture radius", which is the distance sufficiently close to the centroid to be considered stable. This radius is a configuration variable determined by the mission writer. The back and forth between the pProxonoi app and the Voronoi behavior replicate the Lloyd algorithm, with the exception that a new Voronoi tessellation and centroid are calculated for each vehicle prior to it actually achieving the centroid of the previous region. This adjustment allows

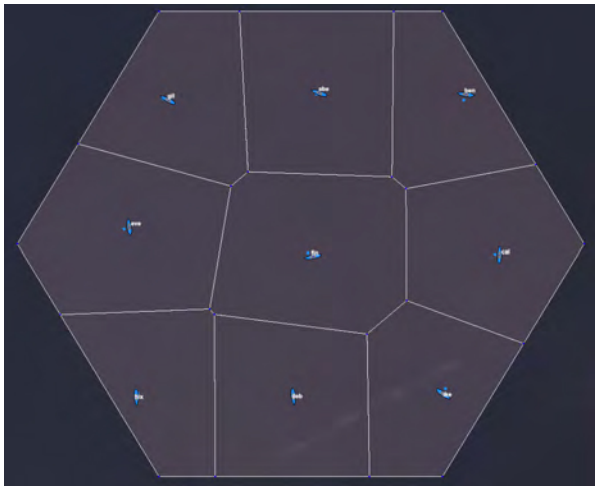
the vehicle to maintain continual movement. An example of the initial, transitional, and final step for the Voronoi behavior can be seen in fig. 2.8



(a) Initial Voronoi Tessellation



(b) Moving towards centroid



(c) Final CVT

Figure 2.8: The Voronoi Behavior

2.3 Coverage and Tracking

The area coverage and tracking problem presented in this thesis differs from many of the classical search problems. For instance, most commonly explored problems require prior knowledge of the target in order to develop a probabilistic search model (Stone *et al.*, 2016). The prior knowledge provides the searching entity a sound starting point and subsequent search options. Further, many game theory problems assume the target is present in the search space. For this thesis, the searching swarm has no prior knowledge of the target and therefore the probability of locating the target is uniform across the search space. Additionally, the target is not always present in the search space and only crosses the region at an unknown time. This prevents many search algorithms from being effectively employed. No particular point in the search space has any higher probability of detection; therefore, the swarm is dispersed to optimally cover the region with a CVT. Further, each vehicle is limited in its ability to detect the target vehicle within its tessellation by restricting its detection range.

The goal of this thesis is to show that a swarm is able to detect a target and produce an estimated target solution that can be used in traditional search algorithms. Thus, the target solution could ultimately serve as the prior knowledge and probabilistic search criteria for a follow on sophisticated search team. While many traditional forms of search theory are not deployable in this scenario, Stone et al. present one similar solvable problem in *Optimal Search for Moving Targets*. A background of the specifics of this type of search problem and the details of it are found in the section below.

20. Stone, L. *et al.*, 2016, *Optimal Search for Moving Targets*

2.3.1 One Sided Search of a Moving Target

This thesis addresses a one sided search of a moving target (Benkoski *et al.*, 1991). This form of search problem has been investigated since the 1940's; however, the solution for a variety of moving target problems did not get addressed until much later and can be quite contrived (Benkoski *et al.*, 1991). In this particular problem, the target being searched for does not respond to being tracked, nor does it operate evasively in the presence of the searcher. Further, restricting the searcher to operate like a real world vessel greatly inhibits the use of traditional search algorithms. Many of the previous optimal solutions do not restrict the searcher and instead permit a search in any sector of the search field at each time step (Benkoski *et al.*, 1991). The use of traditional assumptions allow the detection probability to be modeled with solvable mathematics. The constraints of this thesis can not be quantified as such, and therefore must be deduced by Monte Carlo simulations. The remainder of this section is dedicated to a one sided search of a moving target, whereby it can be modeled and solved.

Stone et al. in *Optimal Search for Moving Targets* present a one sided search problem where the initial location of the target is unknown. Specifically, a channel is modeled where the target traverses a straight line through the channel at an unknown time. Stone et al. solve for the target detection probability given three detection vehicles with sufficiently constrained operating parameters. Fig. 2.9 shows the setup of the search problem. The ability of the search vessel to detect the target is calculated using a Poisson Scan Model with a detection radius of 5 units. The detection vehicles are programmed to complete a figure eight pattern while searching. This pattern is then compared to an optimally generated path. The detection probability is determined through direct integration of the problem presented in eqn. 2.1 instead of Monte Carlo

4. Benkoski, S. J. *et al.*, 1991, A survey of the search theory literature.

testing (Stone *et al.*, 2016). This analytical solution can be obtained due to the exact constraints of the problem.

$$
\begin{aligned}
&X(t,\omega) = \text{Target position at time } t \text{ and uncertainty } \omega \\
&y(t) = \text{Single vessel position at time } t \\
&r(X,y) = \text{Poisson Scan Model Detection given } X \text{ and } y \\
&P = E\left[exp\left(-\int_0^T r(X(t,\omega),y(t))\,dt\right)\right]
\end{aligned}
\tag{2.1}
$$

$$
\begin{aligned}
&\beta = \text{Scan Rate} \\
&\Phi = \text{Standard normal distribution function} \\
&F = \text{Sensor Parameter} \\
&X = X(t,\omega) \\
&y = y(t) \\
&||X-y||^2 - b = \text{Signal loss} \\
&\sigma = \text{Variability} \\
&r(X,y) = \beta\Phi\left(\frac{F-||X-y||^2-b}{\sigma}\right)
\end{aligned}
\tag{2.2}
$$

20. Stone, L. *et al.*, 2016, *Optimal Search for Moving Targets*

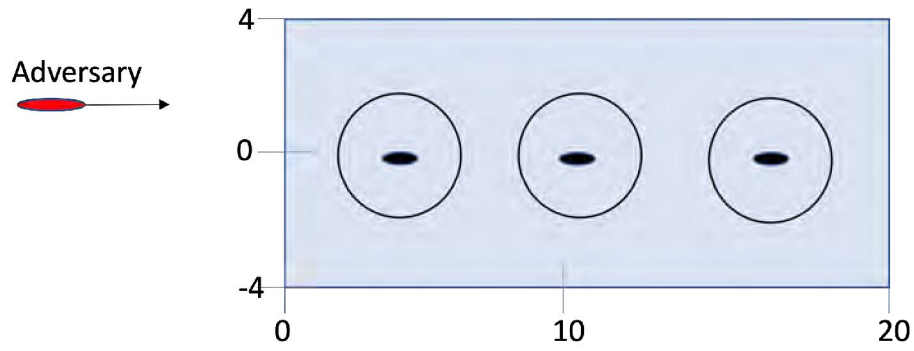


Figure 2.9: Channel Search Problem

The fig. 2.10 shows the paths used by the three vehicles to determine detection probability. The dashed lines in this figure show the pre-programmed paths and the solid line shows the optimally solved solution in order to maximize detection. The paths show each vehicle essentially partitioned the space evenly and performed a search in its associated region. Further, each vehicle moved about the center of its associated region and maintained movement. The detection probability results are shown in table 2.2(Stone *et al.*, 2016).

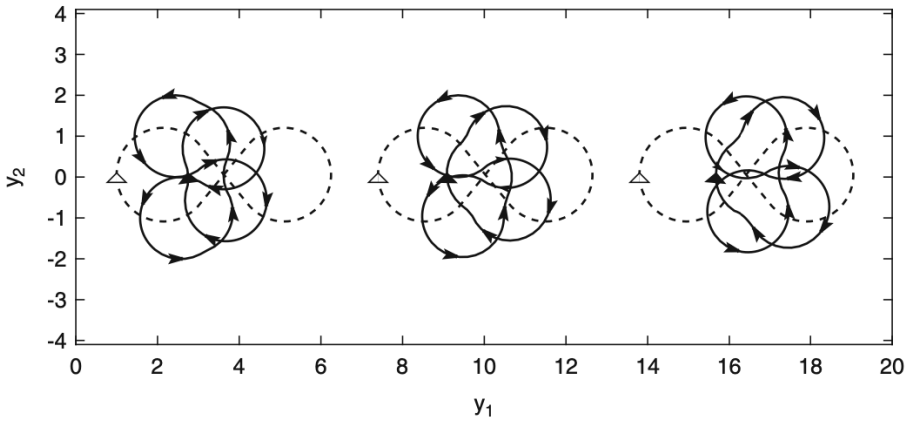


Figure 2.10: Search Vehicle Patterns (Stone *et al.*, 2016)

`Figure Eight`	.90335
`Optimized Solution`	.94086

Table 2.2: Channel Problem Detection Probabilities.

This problem captures much of the goals of this thesis; however, the problem assessed for this thesis does not assume a singular travel channel. The results of this thesis are derived from a target moving in a singular direction but the symmetry of the setup allows for this case to provide results indicative of east-west and north-south travel. This added layer of complexity make the Monte Carlo simulation essential in the quantification of detection probability.

2.3.2 CVTs in Cover and Tracking Scenarios

Below is a summary of different uses of CVTs in cover and tracking scenarios. First, an application of CVTs and the Lloyd's algorithm is shown to be used in a group of robots to maneuver through a large region while maintaining the CVT of a smaller convex region. Then the use of a weighted CVTs in a region search problem is examined. Lastly, a CVT method to cover a region and track a target is summarized.

Teruel et al. demonstrate a method to improve CVT convergence of a robot swarm as it transitions through a large region by a means of smaller convex regions (Teruel *et al.*, 2019). This allows a swarm to effectively search an entire region through a series of smaller steps. Each smaller step is a new polygon in which the swarm achieves a new CVT of this region. An example of this methodology can be seen in fig. 2.11 (Teruel *et al.*, 2019). The results provided in this study highlight an efficient method to control swarms in different CVT regions with limited communications. The method of coverage proposed in this experiment works well for searching a region for a stationary target

22. Teruel, E. *et al.*, 2019, A distributed robot swarm control for dynamic region coverage.

where the search region requires adaptability; however, this methodology focuses more on the efficiency of achieving a CVT not as an effective search tool.

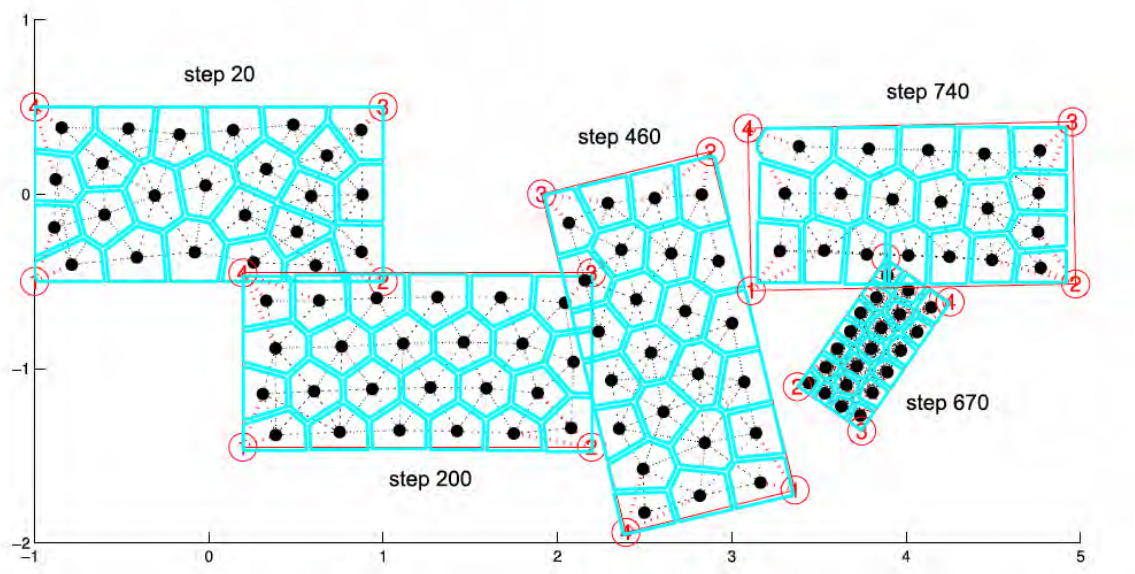


Figure 2.11: Depiction of the transitioning Voronoi polygons where black points are generators and dashed lines represent communication channels. Each frame demonstrates the stability of the CVT at the final time step before it transitions to the next polygon. A total of five polygons are traversed over the region. The shift in times steps, size, and orientation show the convergence efficiency.

Automated Multi-Agent Search Using Centroidal Voronoi Configuration proposes a method to deploy a group of robots and search a region. The strategy involves a series of two step actions to reduce the uncertainty of a region search function. The first step is to achieve a CVT of the region with the robots acting as generators, and then the individual agents perform a search in each region. This is then repeated until the region uncertainty is reduced. Each vehicle starts with the same uncertainty function definition, which serves as the basis for each follow on search iteration. The cover and search methodology was shown to effectively reduce the uncertainty through Matlab simulations (Guruprasad & Ghose, 2011).

12. Guruprasad, K. R. *et al.*, 2011, Automated Multi-Agent Search Using Centroidal Voronoi Configuration.

The simultaneous cover and tracking problem is presented and evaluated by Pimenta et al.. This problem is related to the scenario examined in this thesis. A swarm of vehicles is distributed in a region using CVTs. The swarm then detects and tracks intruders in the region. In this system, vehicles are assigned as cover or tracking vehicles. Each vehicle maintains a time varying density function that weights the task assignment for each vehicle. Therefore, each vehicle will balance its task assignment based on the need to track the vessel or to ensure coverage of the region. A vehicle is assigned as a tracker when the target appears within "tracking range" of a cover vehicle. It is assumed that a vehicle can identify the target vehicle's speed and position if it is within its Voronoi cell. Further, is assumed that each vehicle can determine the target's location and speed if it is within a designated distance from the Voronoi cell. Thus, region coverage is maintained at all times and the ability to track intruders is refined (Pimenta *et al.*, 2009).

Pimenta et al. consider the simultaneous cover and tracking problem as two separate tasks. In this scenario each vehicle is a fully capable tracking vessel and has the ability to formulate independent tracking solutions of the intruder. This assumption leads to the swarm acting as a cover mechanism and individuals as capable trackers. In this thesis, the swarm vehicles are only capable of developing tracking solutions by individual detections within the swarm. Therefore, in order to develop a target tracking solution the swarm needs two or more detections spaced out in time.

2.4 Active Sonar and Sonar Detection Probability

Active sonar detection in ASW has been in use since the early 1900s. However, there are countless use cases for active sonar: fish counters, fish finders, and bathymetric mapping. It is especially useful in scenarios where the object being detected provides

17. Pimenta, L. C. A. *et al.*, 2009, *Simultaneous Coverage and Tracking (SCAT) of Moving Targets with Robot Networks*

consistent high strength returns. In this section an overview of the principles of active sonar are discussed with emphasis on the probability of detection. These principles present the foundation of the probabilistic detection model used for this thesis. The probabilistic nature of active sonar provides justification for implementing a probabilistic sensor in simulation, and this assumption reflects real world swarm detection capability.

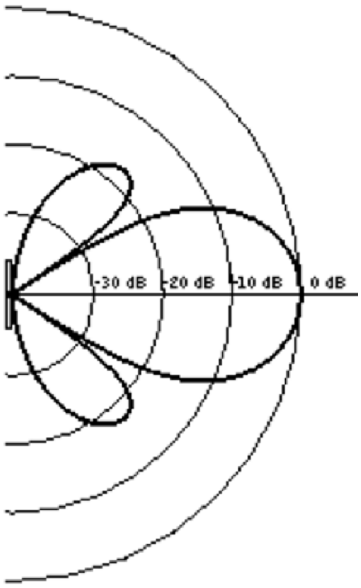


Figure 2.12: Single Beam Sonar Beam Pattern (Yongjie, 2019)

Active sonar is simply a two way transducer that is used to produce a ping of a known source frequency and then listens for the return of the reflected transmission. Fig. 2.12 shows the beam pattern for a single beam transducer (Yongjie, 2019). Most transducers, or echo sounders, use piezoelectric ceramic materials. The ceramic material functions similarly to traditional piezoelectric crystal. Therefore, deflections in the physical structure produces a potential difference and vice versa. Thus, an oscillating voltage will cause the ceramic to physically oscillate, which is used to produce a pressure wave in the water. The echo sounder produces a pressure wave of a known frequency and

28. Yongjie, S., 2019, Underwater Acoustic Transducers.

when the pressure wave reflection returns it is converted to a measurable current (Butler & Sherman, 2016). The actual return strength of the pressure wave is given by the active sonar equation presented in eqn. 2.3 (Wagner *et al.*, 1999).

$$
\begin{aligned}
SL &= \text{Source Level} \\
TS &= \text{Target Strength} \\
TL &= \text{Transmission Loss} \\
SE &= \text{Signal Excess} \\
NS &= \text{Noise Level} \\
SE &= SL - 2TL + TS - NS
\end{aligned}
\tag{2.3}
$$

In order to determine the probability of a particular active sonar device to produce a signal excess with the strength to break the threshold of detection is dependent on each of the terms in the sonar equation. The source strength is a design characteristic of the particular device in use. The transmission loss is directly related to the distance between the source and the target, while the target strength is largely dependent on the material, the presented surface, and location within the beam. A generic example of the detection probability in a static setting can be seen in fig. 2.13, where the width of the single beam sonar is a design feature (Bureau of Naval Personnel, 1953). The data in these sources suggest that detecting a target can be modeled as a probabilistic function of the distance from the source and the target and the perpendicular distance from the center of the beam to the target. In this thesis, the probabilistic function is assumed to be Gaussian. Further discussion is provided in Chapter 3.

7. Butler, J. L. *et al.*, 2016, *Transducers and Arrays for Underwater Sound. 2nd ed.*
26. Wagner, D. *et al.*, 1999, *Naval Operations Analysis*
6. Bureau of Naval Personnel, N., 1953, *Naval Sonar*

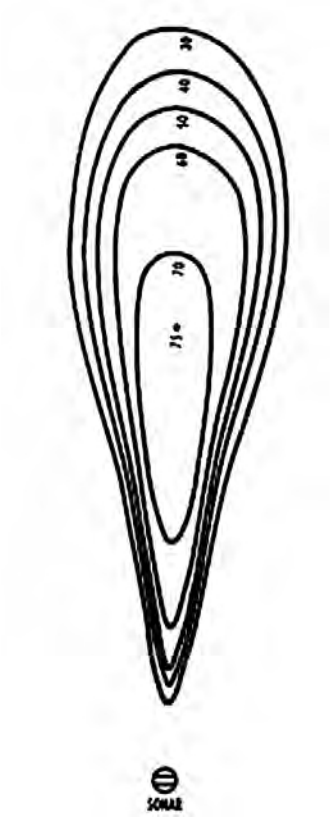


Figure 2.13: Probability of Detection in Sonar Beam (Bureau of Naval Personnel, 1953)

3 Application and Scenario Development in MOOS-IvP

This chapter provides an explanation of the behaviors and applications created for this thesis. Then, the test case scenario that is analyzed for this thesis is defined.

3.1 MOOS-IvP Applications

3.1.1 uFldSearchDetect

uFldSearchDetect is an application that is launched on the shoreside MOOSDB in order to monitor the number of discrete "detections" of an adversary vehicle. The shoreside computer is the mission control platform that produces the visual display pMarineViewer and routes all communications. The uFldSearchDetect application tracks all vehicle locations including the adversary. It makes use of the CPAMonitor class to determine the closest point of approach between the adversary vehicle and any member of the swarm in order to determine if the adversary was "detected". A detection is defined as the adversary coming within a configured discrete "detection" range of a swarm vehicle. uFldSearchDetect is configured to trigger a detection report if the adversary comes within the detection range of a swarm vehicle. uFldSearchDetect is further capable of tracking various range thresholds for detection. Fig. 3.1 demonstrates the detection

threshold for uFldSearchDetect set to 10 m and 20 m. The inner ring in the figure corresponds to the 10m detection range and the outer ring to the 20 m.

Figure 3.1: Detection Rings.

The uFldSearchDetect application provides a method to analyze how close the adversary vehicle gets to each swarm vehicle during a crossing. It tracks the number of "detections" at each configured range increment for follow on analysis. Fig. 3.2 demonstrates the top down probability of detection for a single vessel using the uFldSearchDetect application with a set max detection radius.

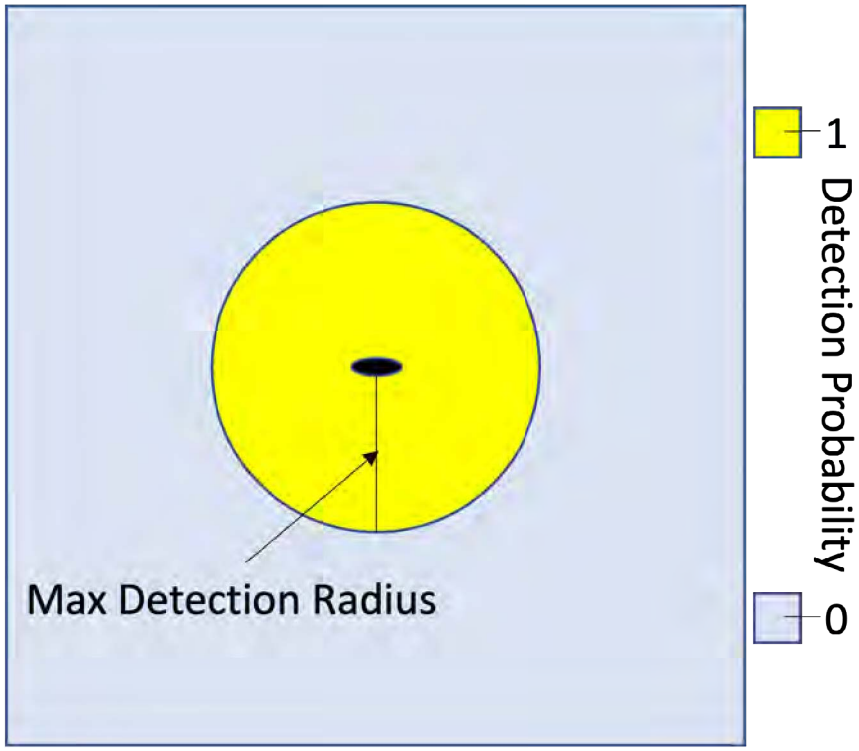


Figure 3.2: Top Down Discrete Detection

3.1.2 pSonarSimDetect

The pSonarSimDetect app is a probabilistic sensor that mimics the behavior of an active sonar. The application is configured with a desired frequency, beam width, and the target vehicle's name. The target vehicle's name is only necessary if the simulated adversary vessel type is not designated as UUV in the adversary's Node Report. Additionally, the app may be configured with the target vehicle's depth if the sim vehicle's depth is not properly included in the target's node report.

pSonarSimDetect subscribes to all Node Reports posted to a vehicle's MOOSDB. The application then processes the node reports to determine if the node report belongs to a UUV or the target name designated at configuration. Further, the application only processes node reports that are published at the assigned sonar frequency. After the node report is validated, the planar distance is then calculated to the target. After which,

the application applies the limitations of the beam width to determine the maximum detection radius at the target's depth, which is determined by eqn. 3.3. The sonar beam is assumed to be a cone shape with beam width defined by depth, which can be seen in fig. 3.3. This is a simplification of the actual beam pattern which was shown in Chapter 2. Once the maximum detection radius is calculated, it is compared to the target vehicle's planar distance. If the distance to the target vehicle is less than the maximum detection radius, the detection probability is determined.

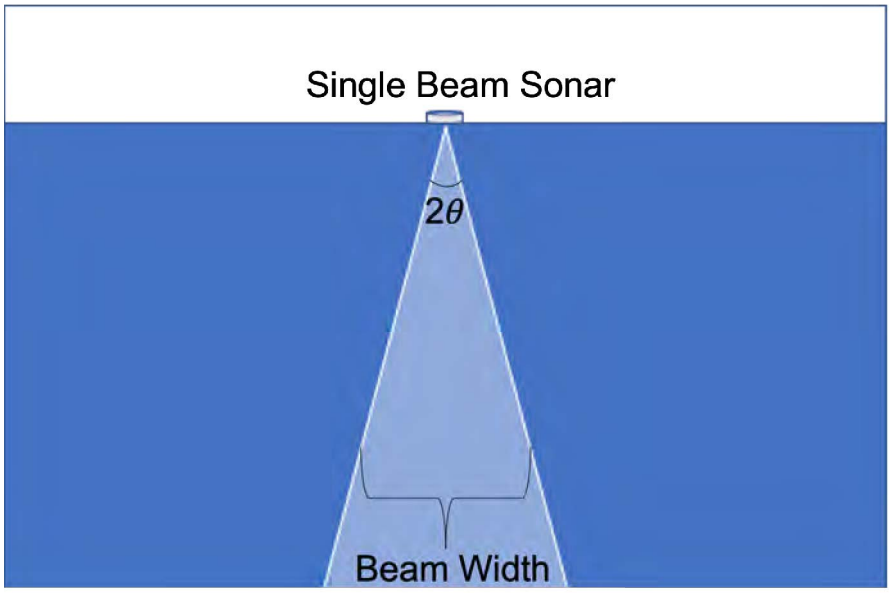


Figure 3.3: Sonar Beam

$$\theta = BeamAngle/2 \tag{3.1}$$

$$BeamWidth = 2 \times TargetDepth \times tan(\theta) \tag{3.2}$$

$$MaxDetectionRadius = BeamWidth/2 \tag{3.3}$$

The maximum detection radius is used as the zero-probability detection distance limit for the pSonarSimDetect app. The app takes a Gaussian normal distribution with the

mean set at the detection vehicle's current position. The detection probability associated with a target directly underneath the detection vessel is set to 1. The distribution for the detection probability, which is based off target vehicle distance to the center of the detection vehicle, is shown in fig. 3.4. In this example the maximum detection radius is set to 10m. In order to create a Gaussian distribution with a drop off to near zero probability at the max detection radius, the standard deviation is $\sigma = MaxDetectionradius/3$. Therefore, the distribution over the region accounts for approximately 99%. Any distance outside the max detection radius is programmed to have a zero probability of detection.

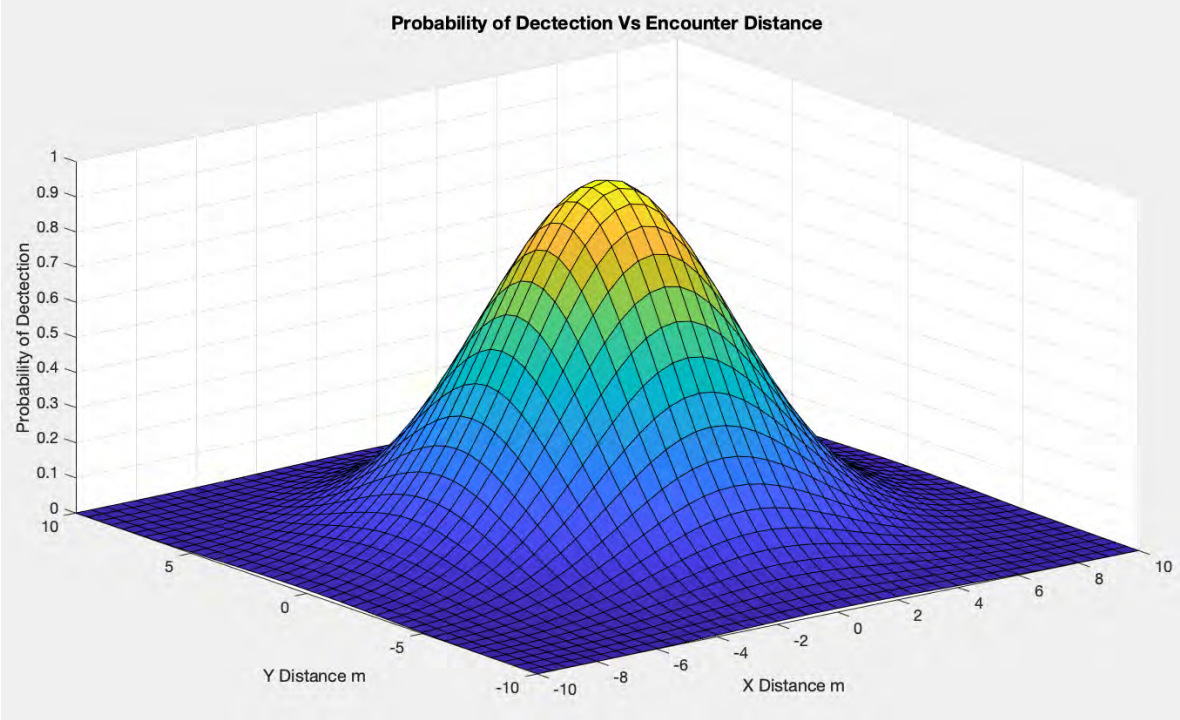


Figure 3.4: Active Sonar Probabilistic Detection Curve

Once the Guassian function is established, the detection probability value is determined for the distance to the target vehicle. A random number between zero and one is then generated. If the random number is less than the detection probability value, the target is considered detected. If the target is detected, a detection report is

generated and shared with the swarm. Since a single beam echo sounder cannot accurately determine where the target is within the beam, the detection report contains the position information of the detection vehicle and the time of detection in the form "$X_{DetectionVehicle}$;$Y_{DetectionVehicle}$;time of detection". The detection report can contain significant positional error, considering the detection vehicle's position is used to represent the target vehicle's location. The process described above is applied to each target vehicle node report received. Therefore, multiple detection reports can be generated by the pSonarSimDetect app; however, the detection reports are limited by the assigned sonar frequency.

The ability for a sonar to detect a target vehicle as a Gaussian distribution was established based of the discussions of active sonar detection in (Wagner *et al.*, 1999) and (Bureau of Naval Personnel, 1953). (Wagner *et al.*, 1999) describes the ability of an active sonar to detect a vessel as probabilistic based on range, while (Bureau of Naval Personnel, 1953) describes detection within the beam as probabilistic based off location in the beam. Further, the test case outlined by (Stone *et al.*, 2016) based its detection results on the Poisson Scan Model that emulates similar sensors and is normal distribution based. Therefore, in order to model an active sonar, pSonarSimDetect assumes that the ability of a vessel to detect a target vessel is probabilistic and Gaussian from the center of the beam to the max detection radius.

Fig. 3.5 is a top down view depicting the probabilistic detection capability of a single vehicle in its designated cell using the pSonarSimDetect application.

26. Wagner, D. *et al.*, 1999, *Naval Operations Analysis*
6. Bureau of Naval Personnel, N., 1953, *Naval Sonar*
20. Stone, L. *et al.*, 2016, *Optimal Search for Moving Targets*

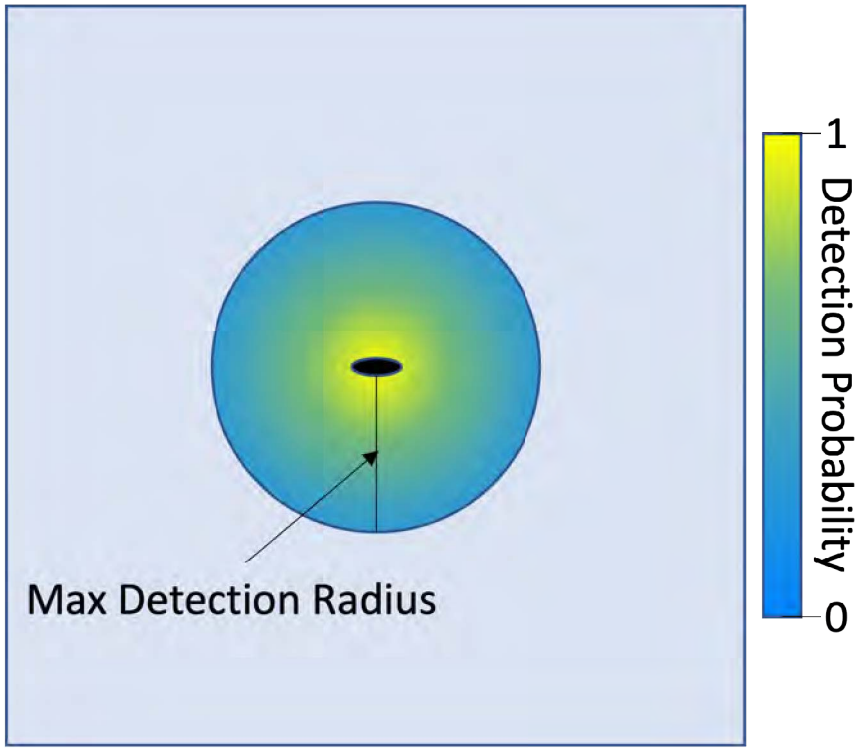


Figure 3.5: Top Down Probabilistic Detection

3.1.3 pKalmanSolutionGen

The pKalmanSolutionGen app is a MOOS application that subscribes to the detection report, which is posted by the pSonarSimDetect app and produces an estimate of the target's heading and speed with associated variance. The application uses a Kalman filter with the detection report acting as the update field. The application tracks the x position, y position, veloctiy in the x direction ($\dot{x}$), and velocity in the y direction ($\dot{y}$). Each detection report includes an x position, y position, and time. The mathematical implementation of the Kalman filter for the x position and $\dot{x}$ velocity is provided in the equation sequence below, which is founded on the work in (Zekavat & Buehrer, 2019). The y position and velocity are calculated in the exact same way as the x. After each update step, the position (x,y) and velocity($\dot{x}$,$\dot{y}$) are used to determine the target's head-

29. Zekavat, R. *et al.*, 2019, *An Introduction to Kalman Filtering Implementation for Localization and Tracking Applications*

ing and speed with a last location of (x,y). These estimated values can then be used to produce a time dependent probabilistic position estimate for follow on search behaviors.

The $\vec{x}$ and error P_0 are initialized with the form seen in eqn.3.4. The values chosen to initialize the position and speed come from the first detection report. The velocity is a complete guess as there is no initial measurement of velocity, therefore the variance associated with the $\dot{x}$ is sufficiently large to account for this error. The initial speed estimate is a configuration variable that is designated based off the assumed speed of the target vehicle.

$$\begin{aligned} \vec{x}_0 &= \begin{bmatrix} x_0 \\ \dot{x}_0 \end{bmatrix} \\ P_0 &= \begin{bmatrix} \sigma^2_{x_{init}} & 0 \\ 0 & \sigma^2_{\dot{x}_{init}} \end{bmatrix} \end{aligned} \tag{3.4}$$

The eqns. in 3.5 show the kinematic model H, the prediction noise variance matrix w_k, and the measurement noise matrix v_k. These values are used to influence the estimate x_{est} under a Gaussian noise.

$$\begin{aligned} H &= \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \\ w_k &= \begin{bmatrix} \sigma^2_{x_{pred}} & 0 \\ 0 & \sigma^2_{\dot{x}_{pred}} \end{bmatrix} \\ v_k &= \begin{bmatrix} \sigma^2_{x_{meas}} & 0 \\ 0 & \sigma^2_{\dot{x}_{meas}} \end{bmatrix} \end{aligned} \tag{3.5}$$

The flowchart found in fig. 3.6 shows the application of the Kalman Filter (Zekavat & Buehrer, 2019). The simplified equations in 3.4, 3.6, 3.7, and 3.8 show the governing equations for each step of the Kalman filter.

$$\begin{aligned} \vec{x}_{est} &= H\vec{x}_0 \\ P_{est} &= HP_0H^T + w_k \end{aligned} \tag{3.6}$$

$$K = P_{est}(P_{est} + v_k)^{-1} \tag{3.7}$$

$$\begin{aligned} P_0 &= (I - K)P_{est} \\ \vec{x}_0 &= K(\vec{x}_{meas} - \vec{x}_{est}) \end{aligned} \tag{3.8}$$

29. Zekavat, R. *et al.*, 2019, *An Introduction to Kalman Filtering Implementation for Localization and Tracking Applications*

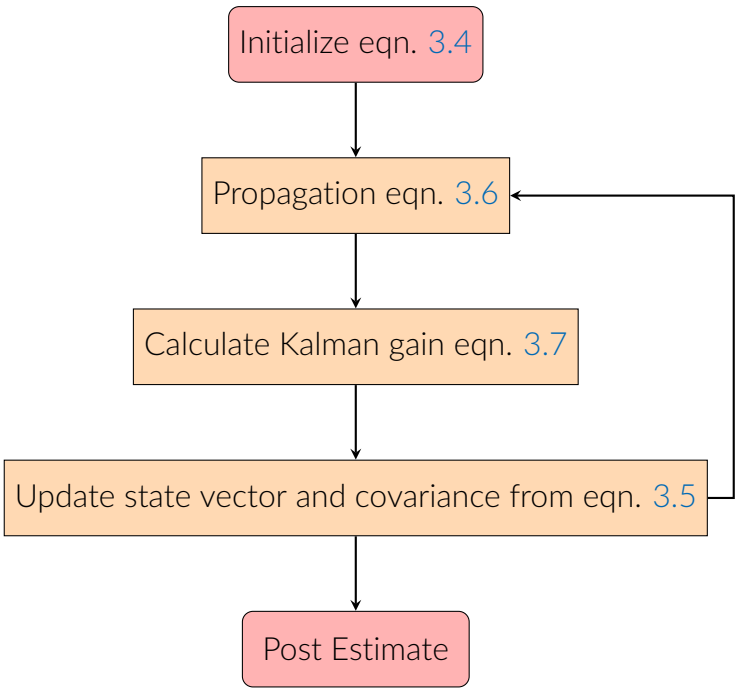


Figure 3.6: Kalman Filter Implementation steps

3.1.4 Region Search Control Behavior

The RegionSearchControl behavior is a behavior that governs how the vehicles operate once the CVT is achieved. There are three configurations for the RegionSeachControl behavior: The CVT cell rotation, the stochastic heading only approach, and the stochastic free. Each of these methods are run independently of the Voronoi behavior. They rely on the output CVT cell for each vehicle from the pProxonoi app. However, once the behavior is active, it no longer adjusts the associated vehicle's cell. Therefore, this behavior relies on the vehicles to first be dispersed in a CVT configuration from the Voronoi behavior and the pProxonoi app. Once stability in the CVT is achieved the RegionSearchControl behavior is activated. The RegionSearchControl behavior mode must not be active alongside the Voronoi behavior mode. For each mode of this behavior the speed can be set via the speed configuration variable. Additionally, each mode allows the vehicle to set the peak utility speed of the objective function randomly. An

example of the region search control behavior objective function can be seen in fig. 3.7, where the peak desired heading and speed is 250 degrees and 1.5 m/s respectively. A description for each configuration mode is detailed below.

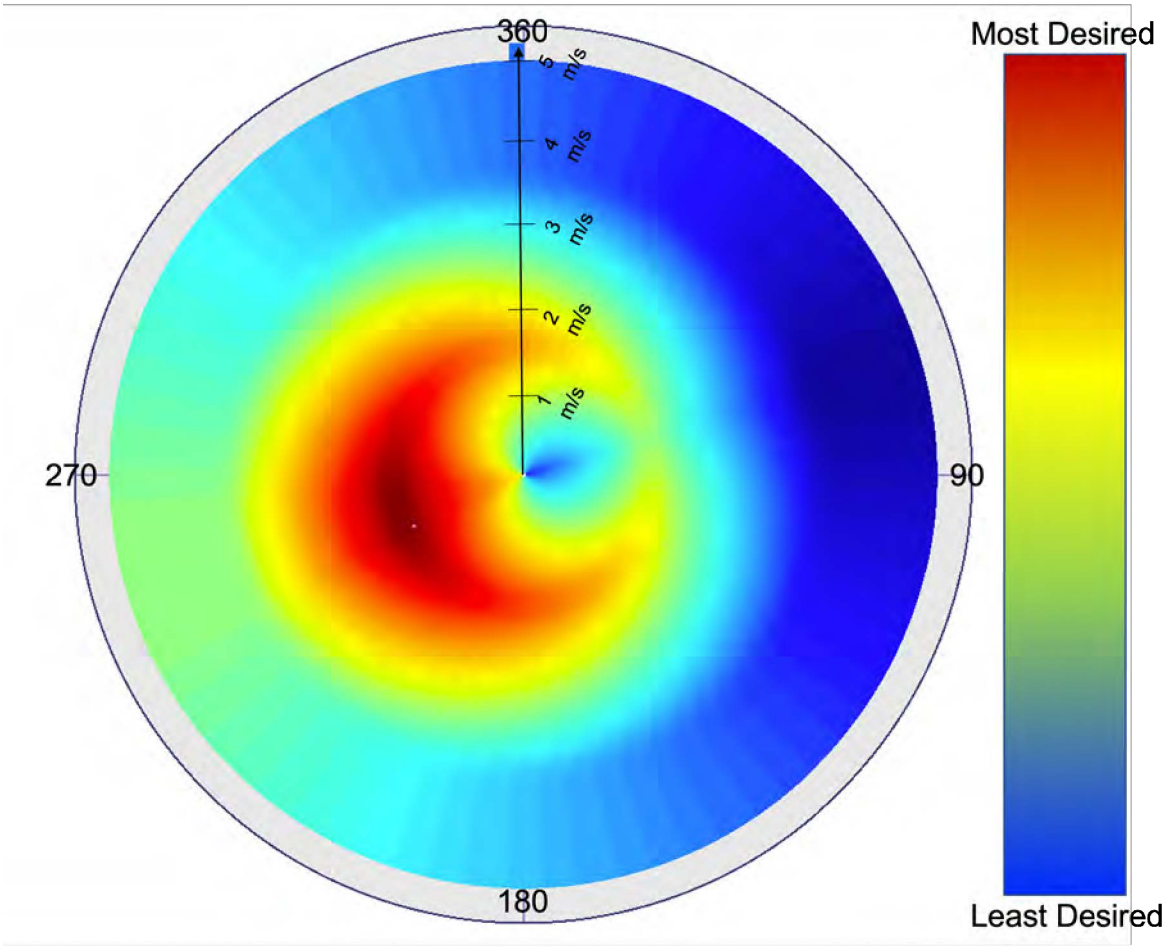


Figure 3.7: Region Search Objective Function

Rotation in Cell

If the RegionSearchControl behavior mode is set to "rotate", the CVT cell rotation method will be employed. When configured in the rotate mode, the behavior policy for generating successive objective functions will cause the vehicle to circle the centroid of its associated CVT cell. The algorithmic steps for the rotate method can be seen in Alg.

2. The radius of the circle, which vehicle will maintain, is a percentage of the maximum radius of the polygon region.

while $CVT \neq stable$

do

    Execute Voronoi Behavior;

end

if *All swarm vehicles* $stable$

then

    Set $Rotate$ to active;

end

while $Rotate = active$

do

    Set $Heading$ to Heading to Region Center;

    if *Distance to center < Spin Radius;*

    then

        $Heading = Heading + 180$;

    else

        D = Distance to Center - Spin Radius;

        F is a scaling factor;

        $Heading = Heading + 90 - (F \times D)$;

    end

end

Algorithm 2: Rotate in Cell Method

Fig. 3.8 shows a swarm of 7 vehicles in a polygon region with the rotate mode of the RegionSearchControl behavior active. The speed for the peak of the objective function is set with the configuration variable speed.

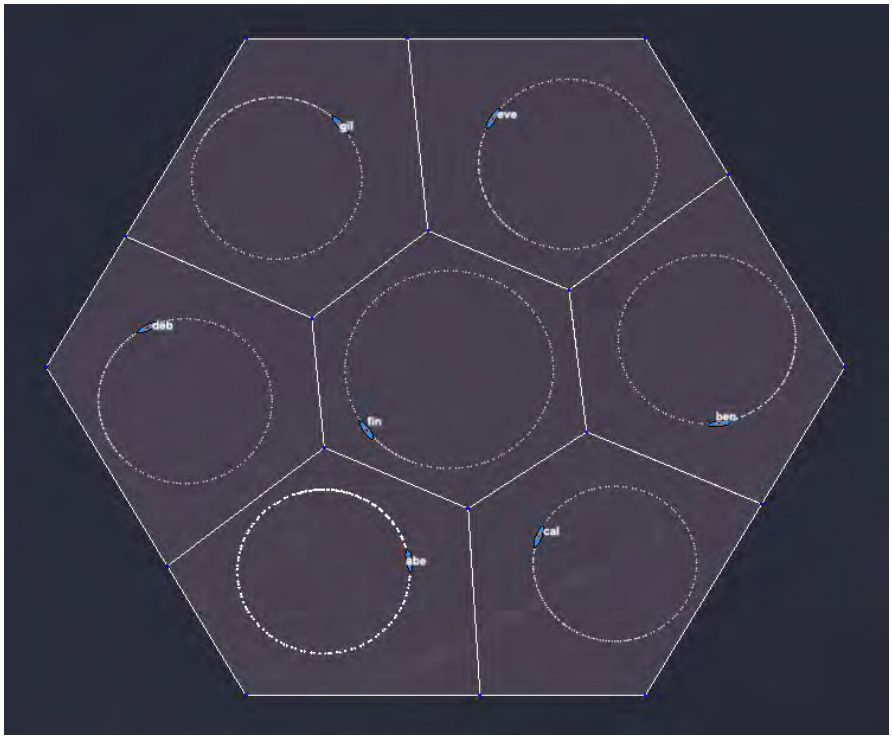


Figure 3.8: Cell Rotation Example

Stochastic Heading in Cell

If the RegionSearchControl behavior mode is set to "stochastic_heading", the stochastic heading method will be executed. Alg.3 summarizes the heading decision for the stochastic heading method. In order to properly execute the behavior, the vehicle looks ahead an established amount of time steps in the future to see if at the current speed the vehicle will proceed outside of the CVT cell. If it is determined that the vehicle will travel outside of the cell it turns and returns to the center. The vehicle will alternate taking a random heading to the edge of its CVT cell and traversing back towards the center.

while $CVT \neq stable$

do

    Execute Voronoi Behavior;

end

if *All swarm vehicles* $stable$

then

    Set $Stochasticheading$ to active;

end

while $Stochasticheading = active$

do

    Set $Heading$ to Previous Heading;

    if *Distance to center < Capture Distance &* $Status$ *=* $Returning$*;*

    then

        $Heading = Random$;

    end

    if *Projected to leave Cell in* x *time steps;*

    then

        $Heading$ = Heading to Region Center;

        Set $Status$ to $Returning$;

    end

end

Algorithm 3: Stochastic Heading in Cell Method

Fig. 3.9 shows a swarm of 7 vehicles in a polygon region with the stochastic heading method of the RegionSearchControl behavior active. The speed for the peak of the objective function is set with the configuration variable speed.

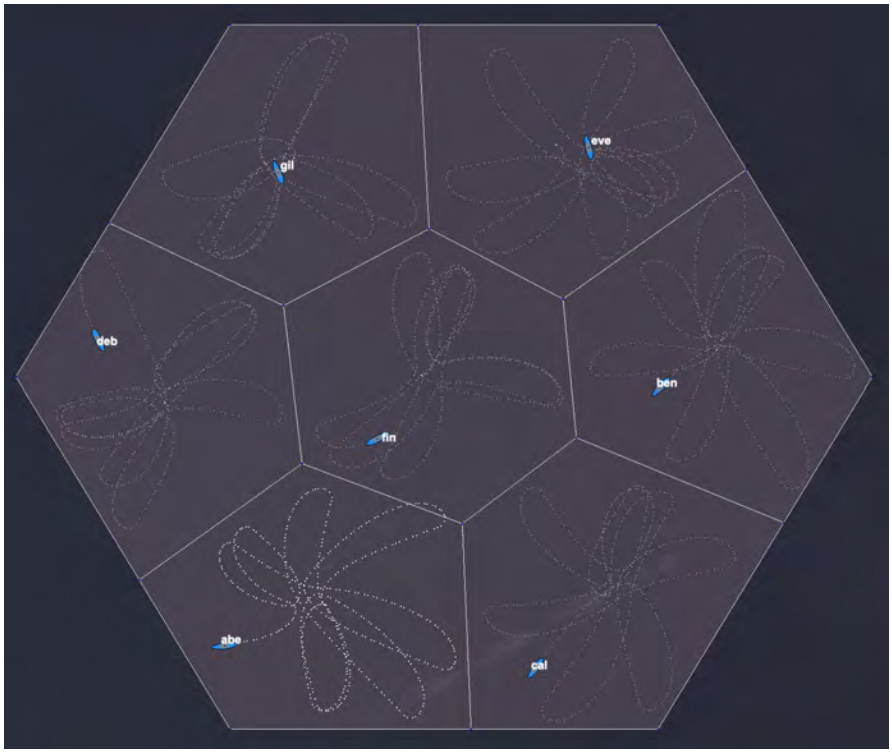


Figure 3.9: Cell Stochastic Heading Method Example

Stochastic Free Mode in Cell

If the RegionSearchControl behavior mode is set to "stochastic_free", the stochastic free method will be executed. The decision making process can be seen in the Alg. 4. The vehicle's position is projected out in time using its current speed and heading. If it is determined that it will be leaving the CVT in this time increment, the vehicle will set a new random heading. The vehicle will also determine if it will leave the cell in an established drop dead time. If it is determined it will leave within the drop dead time or if the vehicle actually exits the cell it will return towards the center until the above criteria is satisfied. This is required in the case that the calculated random headings continue to drive the vehicle towards leaving the cell. At times this results in the vehicle traversing a boundary of the cell, but overall, the vehicle maintains stochastic headings within its assigned region.

while $CVT \neq stable$

do

    Execute Voronoi Behavior;

end

if *All swarm vehicles* $stable$

then

    Set $Stochasticfree$ to active;

end

while $Stochasticfree = active$

do

    Set $Heading$ to Previous Heading;

    if *Projected to leave Cell in* $\leq$x+y$timesteps$

    then

        $Heading = Random;$

    end

    if *Projected to leave Cell in* x *time steps or Out of Cell*

    then

        $Heading$ = Heading to Region Center;

    end

end

Algorithm 4: Stochastic Free in Cell Method

Fig. 3.10 shows a swarm of 7 vehicles in a polygon region with the stochastic free method of the RegionSearchControl behavior active. It can be seen that the path each vehicle drives is random and generally covers the assigned region. The speed for the objective function is set with the configuration variable speed.

Figure 3.10: Cell Stochastic Free Method Example

3.1.5 Vector Field CVT

The Vector Field behavior is a behavior that overlays a constant speed rotational vector field with a polygon region. The objective function produced from the vector field behavior will cause the vehicle to rotate around the center of the operation region. The objective function produced by the vector field behavior can be seen in fig. 3.11, where the peak desired heading and speed is set to 5 degrees and 1.5 m/s respectively. An example of a constant speed vector field can be seen in fig. 3.12.

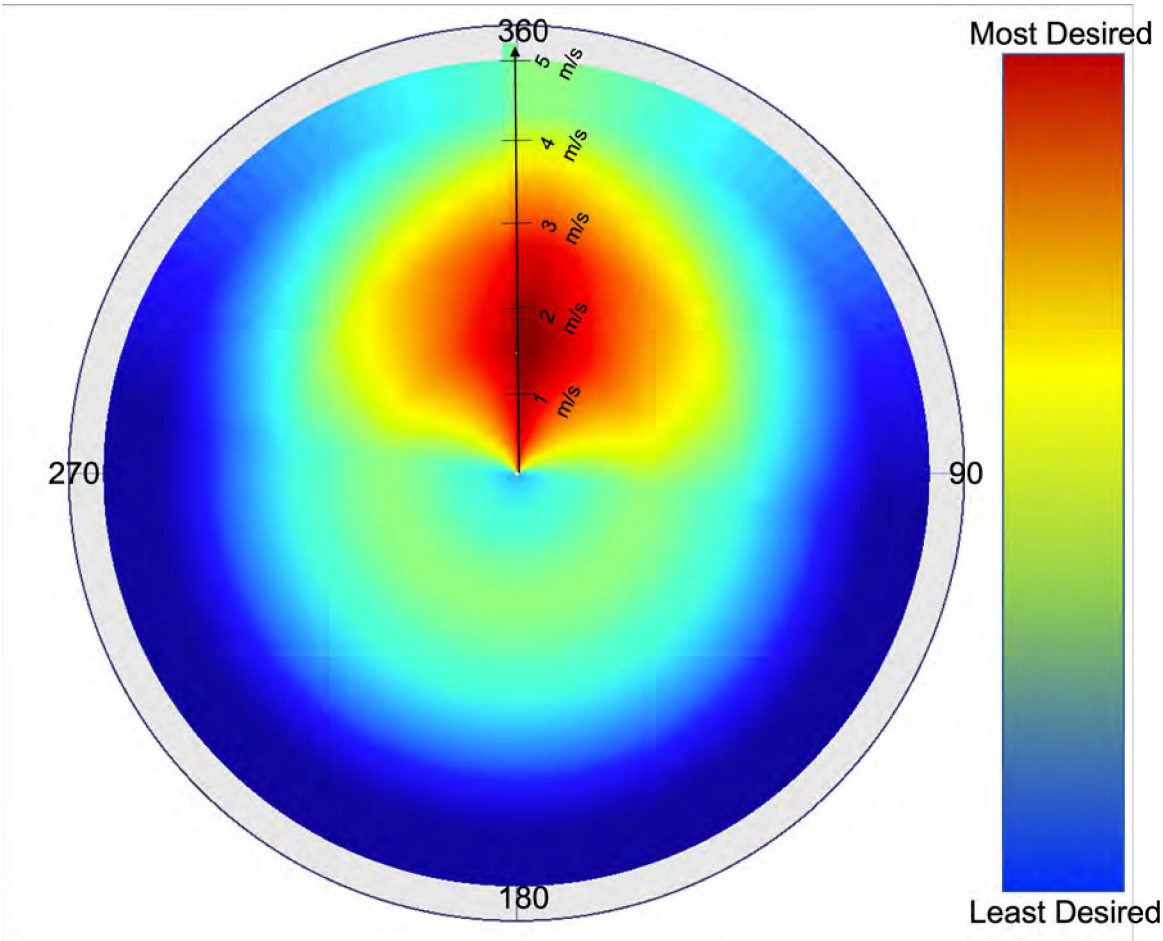


Figure 3.11: Vector Field Objective Function

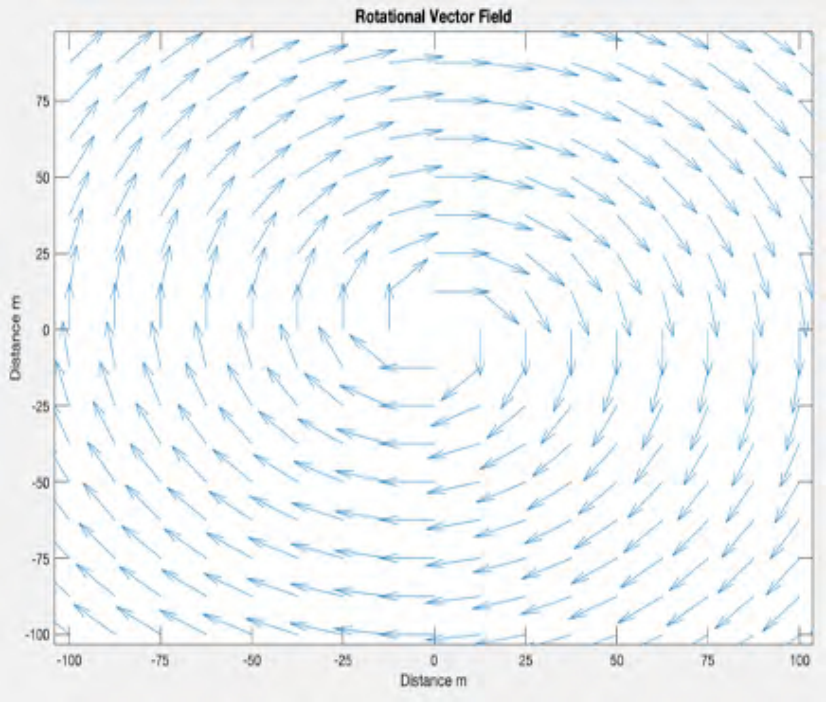


Figure 3.12: Constant Speed Vector Field

The vector field behavior is meant to be run alongside the Voronoi behavior. Therefore, while the vehicles rotate in the vector field, they will also attempt to stay in a CVT configuration. The objective functions for the Voronoi behavior as well as the Vector Field behavior are sent to the pHelmIvP for an optimal solution. This allows the vehicles to continue to rotate around the region center while maintaining the CVT. However, there can be instances when the vehicle strays far enough from the center of its Voronoi cell that it must travel against the Vector Field to maintain proper dispersion. Fig. 3.13 shows the swarm with both the Vector Field behavior and the Voronoi behavior running simultaneously. The vehicle trails are shown with the trailing white dots.

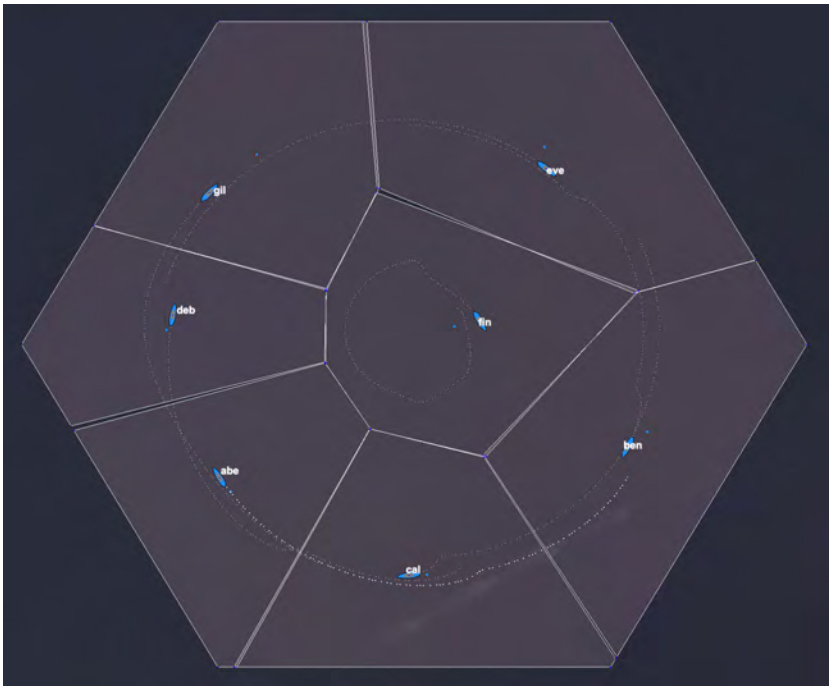


Figure 3.13: Vector Field and Voronoi Behavior Operating

3.2 Experimental Test Setup

The purpose of this test setup is to demonstrate the capability of a swarm to cover an open region, detect a threat passing through from any angle, and develop a tracking solution for the threat target. In order to accomplish this goal a simulated 200m by 200m square region was chosen as the cover region. A swarm of nine surface vehicles were initialized inside this region, and an adversary vehicle is set to cross the square region from one side to the other. Each swarm vehicle is limited in its detection capability, which leads to imperfect coverage of the region.

The Simulations were conducted with the following parameters in order to compare the effectiveness of the swarm to "detect" the adversary using the search methods described above.

1. A swarm of 9 vehicles with identical applications (detection capability) were employed with the capabilities listed in table 3.1.

1	Unlimited Communications
2	Differential Thrust
3	Maximum Speed = 5 m/s
4	Probabilistic Sensor pSonarSimDetect
5	Solution Generation application pKalmanSolutionGen

Table 3.1: Simulated Vehicle Capabilities.

2. The adversary is simulated with the capabilities listed in table 3.2.

1	Depth of 40m
2	Reports Location at 6 Hz
3	Maximum Speed = 5 m/s

Table 3.2: Simulated Adversary Vehicle Capabilities.

The scenario is initialized with the setup shown in fig. 3.14, where the starting position of the adversary is a random location inside the western ellipse. Once the vehicles are initialized, the voronoi behavior is initiated to distribute the vehicles as a CVT of the region as shown in fig. 3.15.

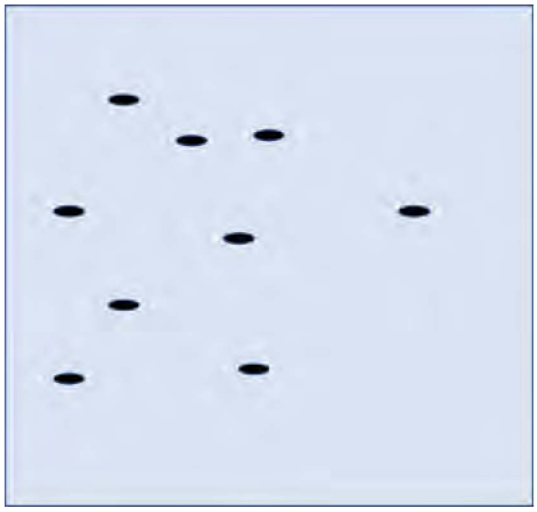
Figure 3.14: The initial test setup.

Figure 3.15: The final CVT configuration.

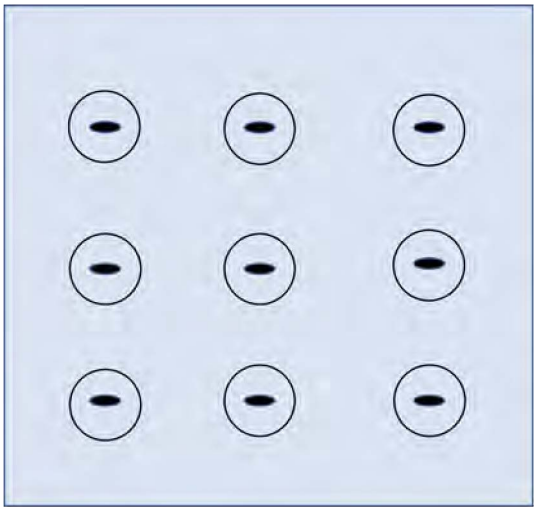
After the vehicles are distributed in a CVT, the specific behavior under test is activated and the "Adversary" vehicle crosses the region, which an example can be seen in fig. 3.16. In order to evaluate the effectiveness of the swarm at detecting the adversary vessel, the uFldSearchDetect and pSonarSimDetect applications are employed. Once the adversary completely crosses the region and enters the ellipse to the east, it resets to

a new start point within the ellipse to the west of the square region seen in figs. 3.14 and 3.15. The adversary vehicle then crosses the region to the east with a new random end point in the eastern ellipse. While this test setup has the adversary crossing in only one direction, the symmetry of the problem produces results that would be representative of crossing from either side of the region. The general test flow can be seen in fig. 3.17.

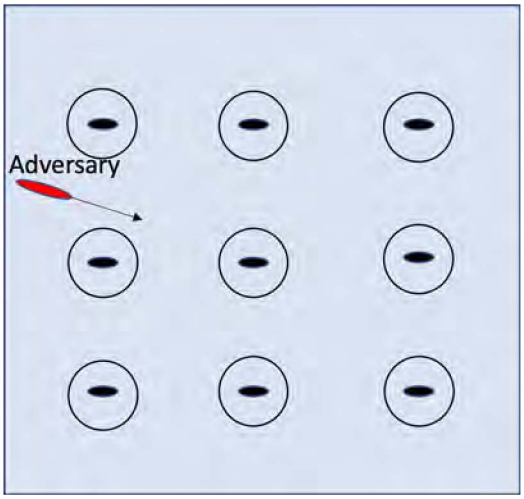
Figure 3.16: The in progress mission.



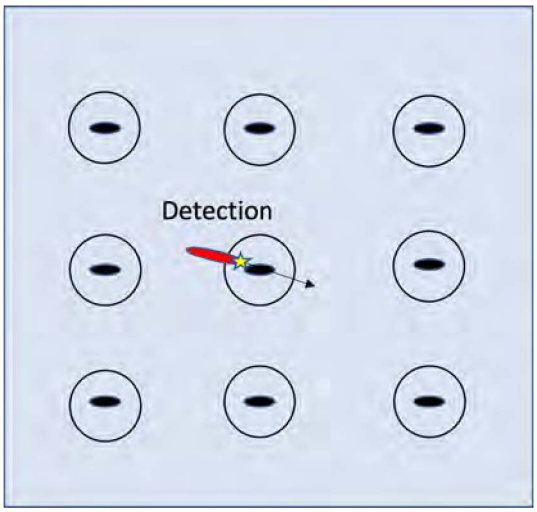
(a) Swarm Vehicles Starting Postion



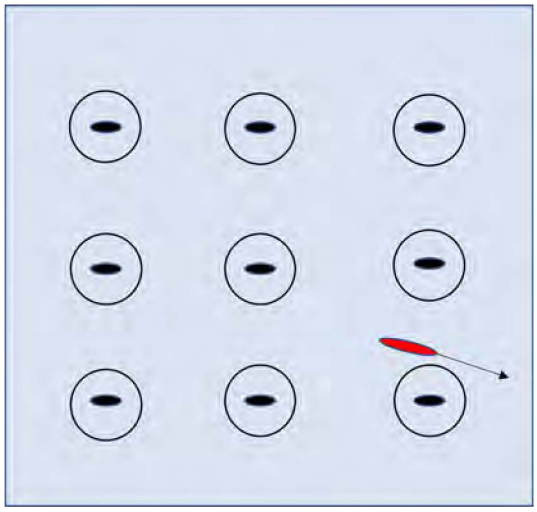
(b) CVT with Detection Radius



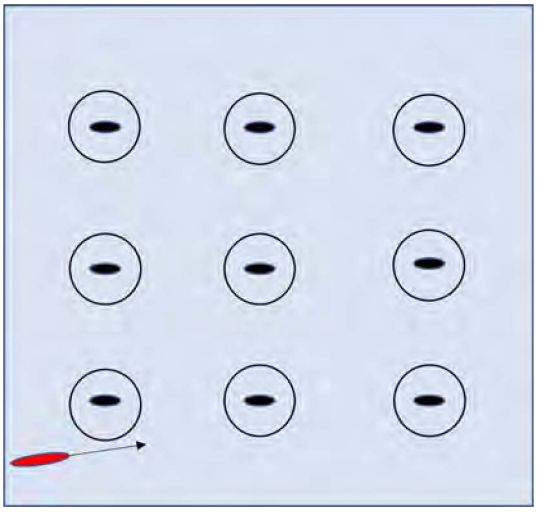
(c) Random Adversary Start



(d) Swarm Vehicle Detects Adversary



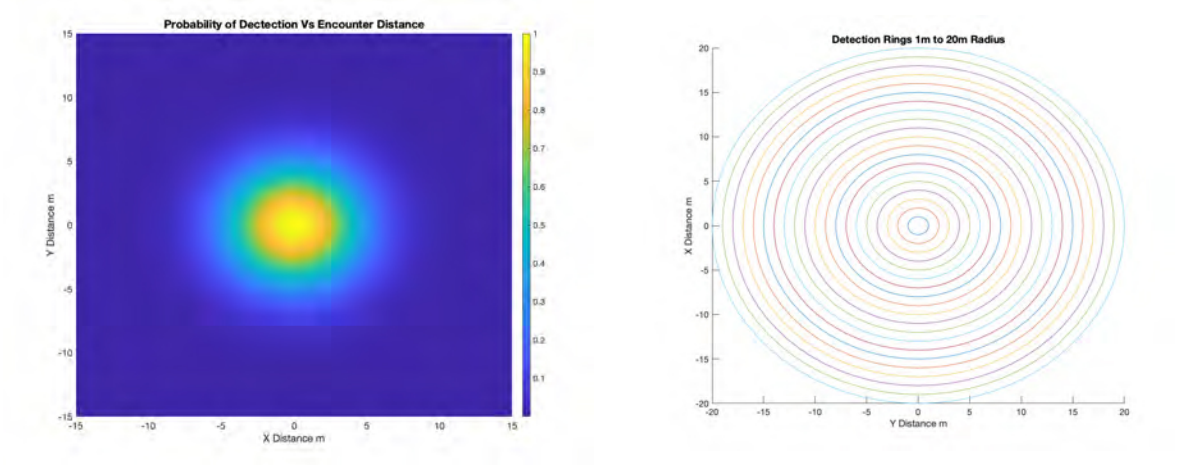
(e) Adversary Crosses Region



(f) New Adversary Initiates Region Crossing

Figure 3.17: A step by step example of the test setup

As the adversary crosses the region two independent methods for a swarm vehicle detecting it are employed. Throughout the simulation pSonarSimDetect is used to simulate probabilistic detection, and uFldSearchDetect is employed to provide discrete detection results. The uFldSearchDetect application is configured to track discrete detections at a 1 m interval from 1m to 20m. The pSonarSimDetect was used to report detections based of the adversary vehicles depth using a single 30 degree beam. Both methods are deployed side by side in order to compare the detection data using two different methods. Each method produces detection reports with a unique variable-value pair. Therefore, the data from each detection method can be analyzed individually and compared.



(a) Probabilistic detection for pSonarSimDetect

(b) Discrete Detection Rings 1m to 20m

Figure 3.18: Simulation Detection Methods

3.3 Monte Carlo Test Procedure

The Monte Carlo simulations were deployed using MOOS-IvP's organic batch simulation script xlaunch.sh. This script executes the launch script for the mission and brings it down when an exit criterion is met. This is done by using the MOOS-IvP application uQueryDB, which serves as a method to query the associated MOOSDB for specified

variable values. Once the exit criterion is met, uQueryDB will post the termination request inside the xlaunch script. For the simulated test scenario of this thesis, the number of times that the adversary vehicle crosses the op region is used as the query variable. Once the vehicle crosses the region a designated number of times the mission is brought down. This method permits various scenarios to be launched by consecutive calls to xlaunch. This was the method employed for this thesis.

After the adversary crossed the op region a designated number of times, the scenario was restarted with a change to one of the tested parameters. This allowed the simulations to be run in a headless configuration with no display GUIs at a much higher time warp. Thereby, the tests were run in an automated method to acquire the results displayed in the results section. The required source code and the automated launch mission scripts can be obtained and run in accordance with the instructions provided in the Appendix.

3.4 Hardware Test

The purpose of the hardware test is to demonstrate the deployability of the search algorithms on actual autonomous vehicles. To this end, the hardware testing for this thesis was conducted using the Clearpath M300 ASV. Fig. 3.19 is a photograph of the M300 ASV. The vessel has a "front seat" computer that is responsible for all networking and control features, while a "back seat" computer contains the MOOS-IvP framework and produces all control commands. Each vessel is connected via a local wifi network with a shoreside computer that acts as a message handler between vehicles as well as local scenario control for the operator.

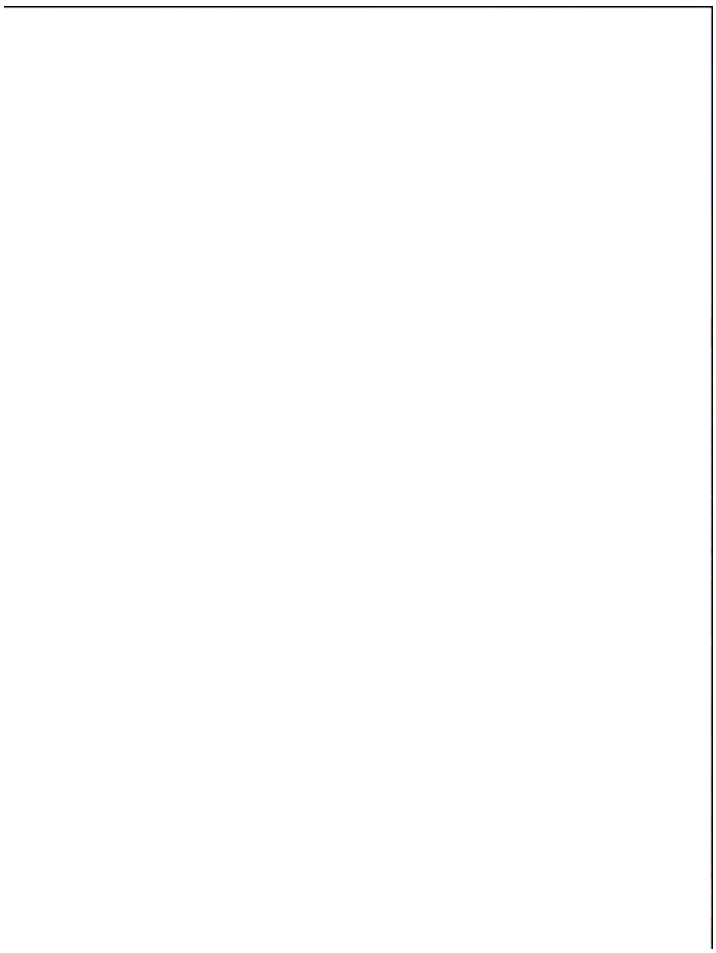


Figure 3.19: Heron USV.

The scenario tested in the Monte Carlo simulations was replicated at a slightly smaller scale with a fleet of 7 Heron USVs shown in fig. 3.21. The cover region was designed as a 150 meter by 150 meter box. This was done to accommodate the vessel traffic on the Charles River near MIT. The location of the cover region can be seen in the Google Earth image found in fig. 3.22. It is important to note that a 7 vehicle group can generate a differing CVT diagram for each deployment. Further, the variation in the CVT diagram can lead to asymmetrical configurations. An example of which can be seen in fig. 3.20. This is significant because symmetry was the assumption used to justify why the adversary vehicle only crosses the cover region in one direction for testing. However, the purpose of the hardware test is to demonstrate the deployability

of the algorithms, not to determine the probability of detection. In order to determine the probability of detection for this specific scenario, the simulation would need to be altered. The adversary vehicle would need to cross the region from any direction to correct for the lack of symmetry in the vehicle dispersion.

Figure 3.20: 7 Vehicle CVT Partition

For the hardware test, each vehicle was deployed with the pertinent applications listed in table 3.3, and the shoreside computer hosted the applications listed in 3.4. The shoreside applications listed in table 3.4 are included on the shoreside to easily consolidate the solution data; however, these applications could be deployed on each vehicle. The UUV adversary was simulated on a local laptop with the fleet of Herons carrying out the search mission. This setup demonstrates the deployability of the search algorithms.

Figure 3.21: The 7 Heron USVs used for Testing.

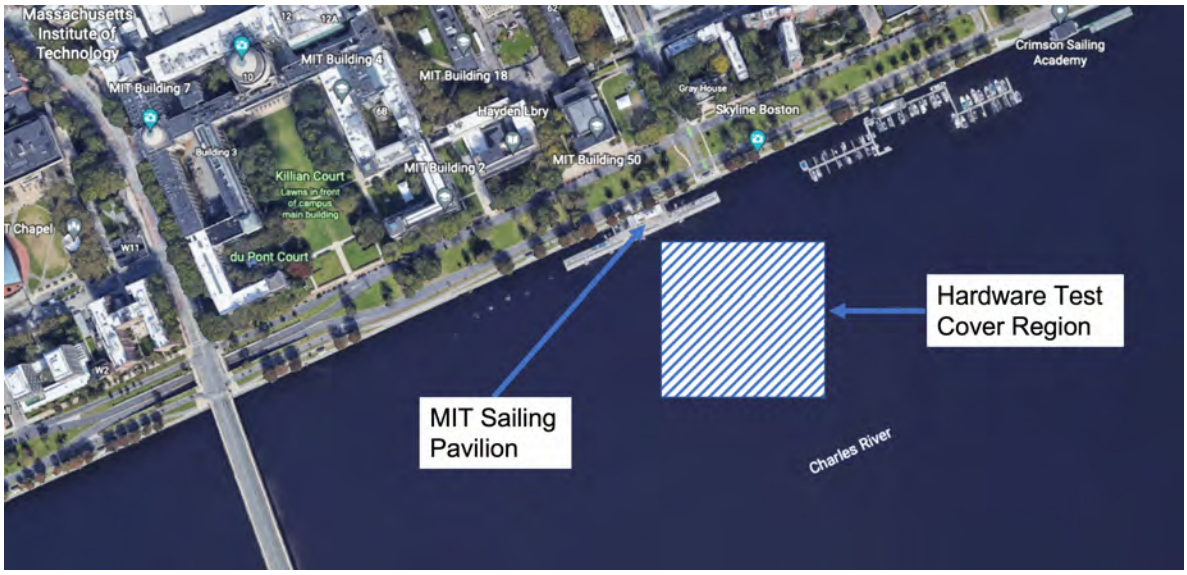


Figure 3.22: Hardware Test Cover Region from Google Earth Image.

1	pSonarSimDetect
2	Region Search Behavior
3	Vector Field Behavior
4	Voronoi Behavior

Table 3.3: Heron Hosted Applications.

1	uFldSearchDetect
2	pKalmanSolutionGen

Table 3.4: Shoreside Computer Hosted Applications.

4 Results

This chapter presents the results from both the Monte Carlo simulation testing and the hardware deployment testing outlined in Chapter 3. The Various search methods of the previous chapter are compared to each other, and they are ultimately evaluated against the stationary swarm in a CVT partitioned region.

The simulation results include the discrete detection results and the probabilistic sensor results. For each search method the swarm speed to adversary speed ratio was changed in order to demonstrate the search algorithm effectiveness and limitations. Each search method was tested using speed ratios $Speed_{swarm}/Speed_{Adversary}$ of 3.0, 1.5, 1.0, .75, .6, and .5. All simulation results presented are based on the Monte Carlo simulation setup described in Chapter 3, where each search method was evaluated at each speed ratio for 500 adversary crossings. The simulation results are presented for the convergence analysis, the discrete detections using uFldSearchDetect, the probabilistic sensor detections, and the Kalman filter solution generation. Then the simulation results are ultimately compared.

The hardware results present the search methods deployed on 7 heron USVs. Each method was deployed alongside a simulated adversary. The Adversary was set to cross the cover region 5 times for each scenario in order to demonstrate the detection applications functioning in a hardware scenario. The speed ratio for the hardware test was set at 1 for every search method.

4.1 Monte Carlo Simulation Convergence

The assumption that 500 adversary crossings is a sufficient number of crossings to draw comparison conclusions from is explained in this section. A series of simulations were conducted ranging from 10 crossings to one thousand crossings. The results of probabilistic detection for each simulation batch are shown in Fig.4.1. Probabilistic detection was chosen as the convergence test parameter due to it containing the most random variability in all test cases. Fig.4.1 shows the weighted mean detection probability and standard deviation for the set of batch runs. Upon examining Fig.4.1, we can see that after 100 test runs are completed, the variability of detection decreases substantially, to less than one standard deviation from the mean of the batch. Therefore, to provide a buffer between this lower value and to reduce the time of each Monte Carlo simulation, 500 crossings were chosen.

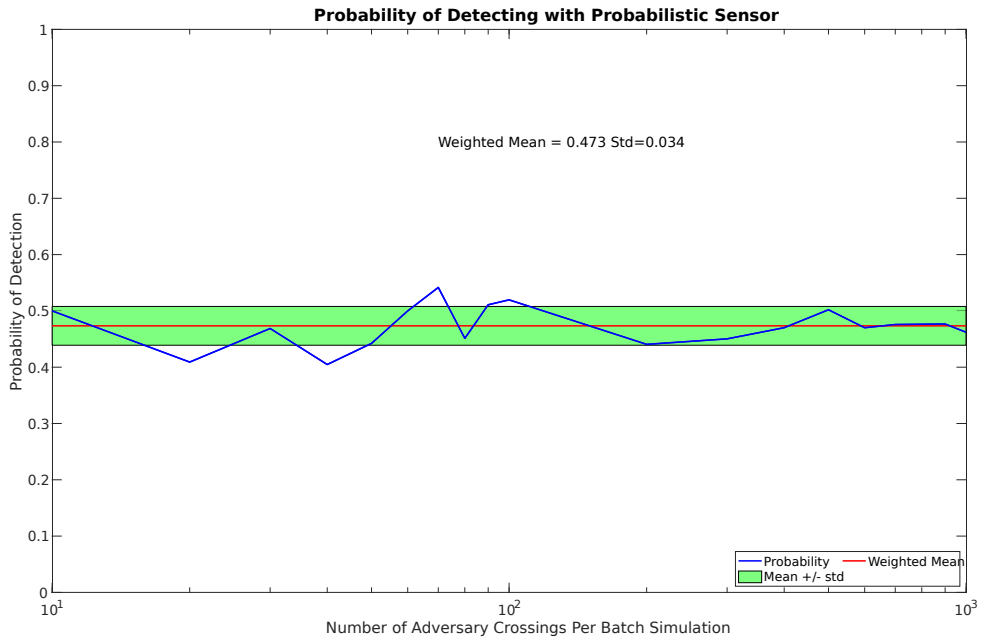


Figure 4.1: Convergence Data Plot

The mean and standard deviation for discrete detections using a stationary swarm are presented in Fig. 4.2. The stationary swarm was tested using 500 crossings for increasing target speeds to demonstrate the variance of the test scenario. As the speed ratio between the swarm and target vessel does not change, the probability of detection for each test batch simulation should be the same. The variance seen in fig. 4.2 is indicative of the repeatability of the results. The average standard deviation over the detection distances is 0.015.

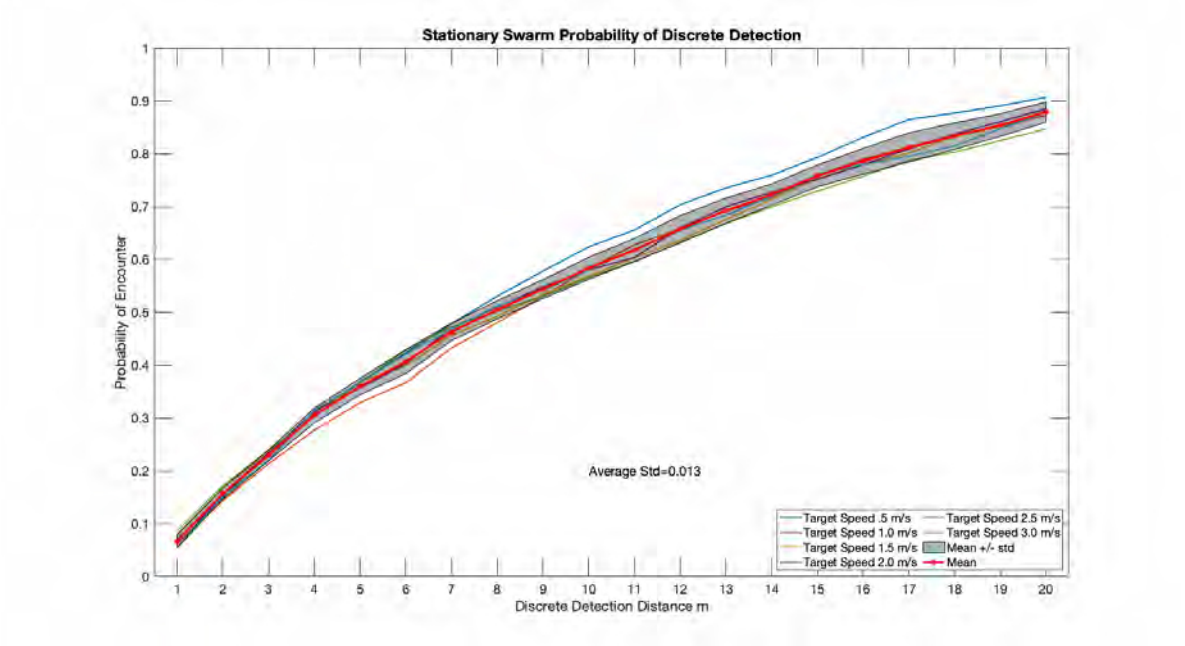


Figure 4.2: Deviation of Stationary Swarm Detection

4.2 Discrete Distance Dectection Simulation Results

4.2.1 Stationary Swarm Simulation Results

This section provides the results that were obtained for adversary "detection" using a stationary swarm in a CVT partitioned space. The stationary CVT partitioned space can be seen in fig 4.3. This is the first step discussed in the test procedure of the previous

chapter. The results of the stationary swarm in this section are used as the standard by which the other search methods are measured against in the simulation comparison section of this chapter.

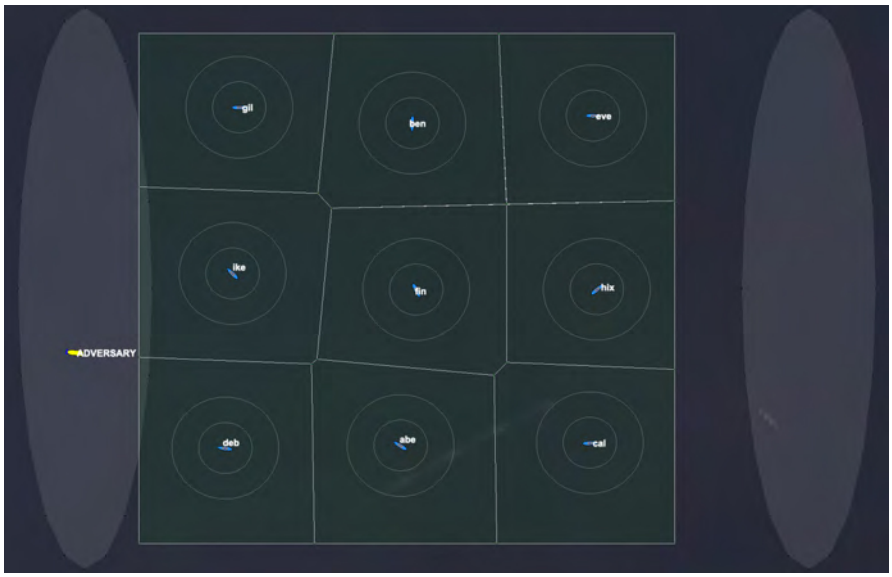


Figure 4.3: The final CVT configuration depicting the 10m and 20m detection radius.

Fig.4.4 shows the numerical results for the number of discrete detections that the swarm achieved for varying adversary speeds. It is clear that the number of detections increases as the detection radius increases. Further, fig. 4.4 shows that changing the speed of the adversary vehicle does not noticeably effect the number of detections.

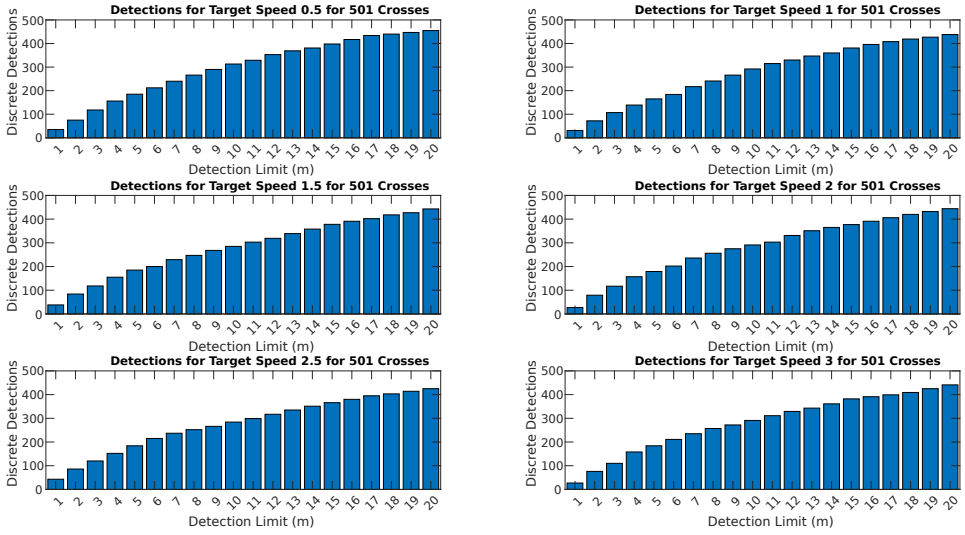


Figure 4.4: Raw counts of interactions for discrete distances for Stationary Swarm

Fig.4.5 plots the probability of detection for increasing the adversary speed. The individual probability values are found in table 4.1 Considering the swarm is always stationary the speed ratio remains the same and is equal to zero. Therefore, the ability of the swarm to detect the adversary should not vary with speed. Thus, the variation in detection probability shown is directly related to the convergence variance of the Monte Carlo simulations using 500 test runs. Further, the adversary positional data was reported at the same frequency across all simulations. Therefore, the fact that the 0.5 m/s adversary simulation had a slightly higher detection probability than the rest could be due to the smaller distance between adversary reports. This would lead to a higher fidelity on adversary position. However, The plot lines are generally well grouped. The error induced by the adversary position reporting is grouped with the convergence error of the Monte Carlo simulation to represent the accuracy limitations of the test setup. In order to account for this, the mean results are used as the standard of comparison in the comparison section. The plot of mean detection probability is given in fig. 4.6.

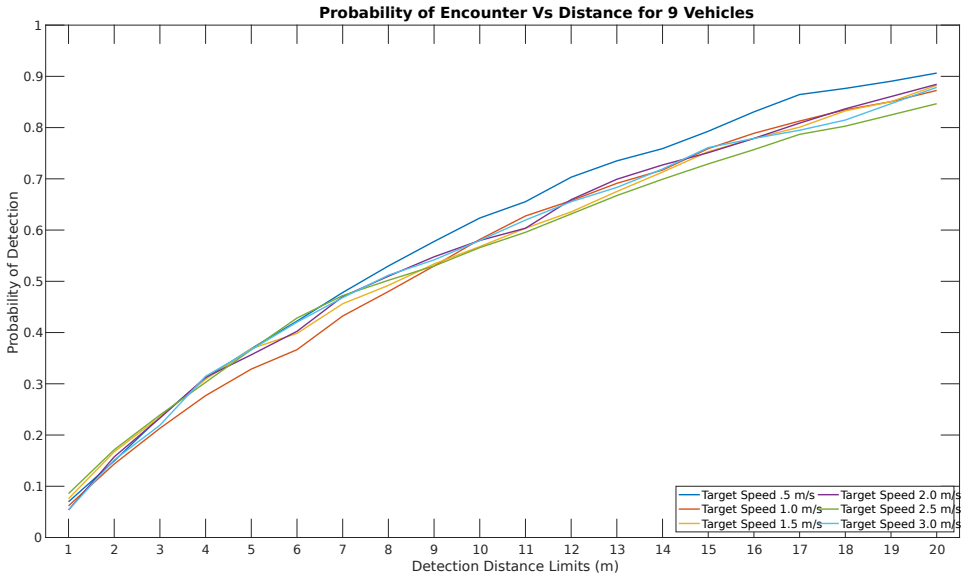


Figure 4.5: Probability of Interaction at Discrete Distances for Stationary Swarm

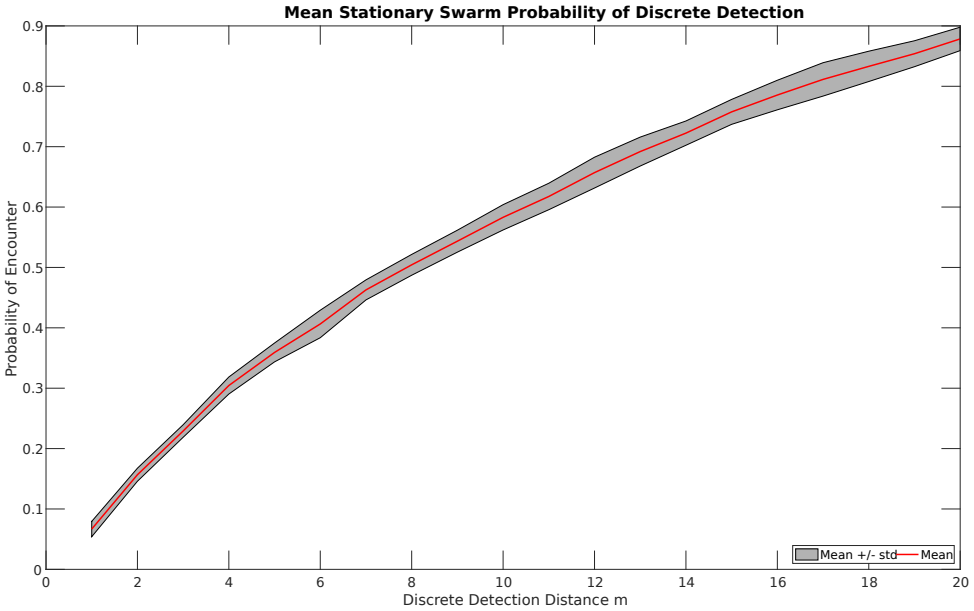


Figure 4.6: Average Probability for Stationary Swarm

Table 4.1: Probability of Interaction at Discrete Distances for Varying Adversary Speeds.

$Ad.v$	1.0	2.0	3.0	4.0	5.0	6.0	7.0	8.0	9.0.	10.0
m/s	m	m	m	m	m	m	m	m	m	m
0.500	0.070	0.149	0.235	0.311	0.369	0.422	0.478	0.530	0.578	0.624
1.000	0.062	0.143	0.213	0.277	0.329	0.367	0.432	0.480	0.530	0.582
1.500	0.076	0.167	0.235	0.309	0.369	0.398	0.456	0.492	0.534	0.568
2.000	0.054	0.157	0.233	0.313	0.357	0.402	0.470	0.510	0.548	0.580
2.500	0.086	0.171	0.239	0.303	0.367	0.428	0.472	0.502	0.530	0.566
3.000	0.054	0.151	0.219	0.315	0.367	0.420	0.468	0.512	0.542	0.580
$Ad.v$	11.0	12.0	13.0	14.0	15.0	16.0	17.0	18.0	19.0.	20.0
m/s	m	m	m	m	m	m	m	m	m	m
0.500	0.655	0.703	0.735	0.759	0.793	0.831	0.865	0.876	0.890	0.906
1.000	0.627	0.657	0.691	0.717	0.759	0.789	0.813	0.835	0.851	0.873
1.500	0.604	0.635	0.675	0.713	0.753	0.779	0.801	0.833	0.851	0.882
2.000	0.604	0.659	0.699	0.727	0.751	0.779	0.809	0.837	0.861	0.884
2.500	0.596	0.631	0.667	0.699	0.729	0.757	0.787	0.803	0.825	0.847
3.000	0.620	0.655	0.683	0.719	0.761	0.779	0.795	0.815	0.847	0.878

4.2.2 Vector Field Simulation Results

The simulation results for the Monte Carlo simulations with the vector field search behavior enacted are presented in this section. The raw count data is displayed in fig. 4.7. The probability of detection at the discrete intervals are shown in fig. 4.8 and table 4.2. It can be seen that the speed ratio plays a significant role in the detection capability of the swarm. As the speed ratio decreases (Target vessel gains speed advantage), the probability of detection also decreases. The lower speed ratios tend to converge within the accuracy of the simulation as discussed above.

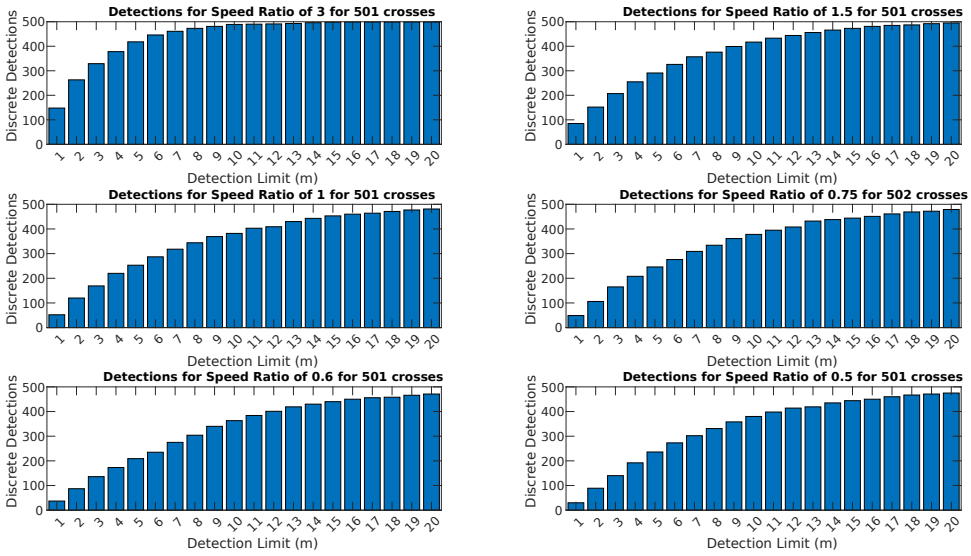


Figure 4.7: Raw counts of interactions for discrete distances for Vector Field Method

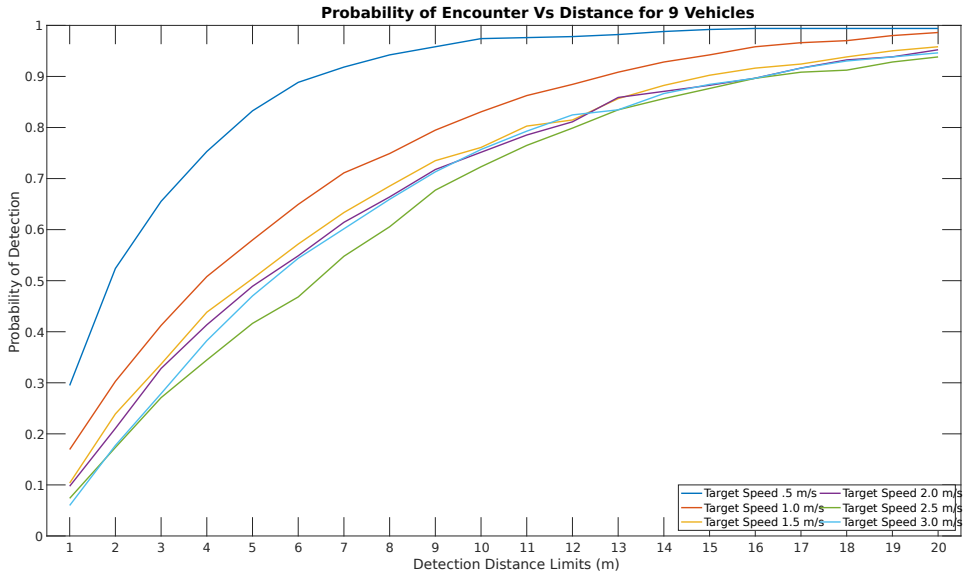


Figure 4.8: Probability of Interaction at Discrete Distances for Vector Field Method

Table 4.2: Probability of Interaction at Discrete Distances for Varying Speed Ratios for Vector Field Method.

SR	1.0 m	2.0 m	3.0 m	4.0 m	5.0 m	6.0 m	7.0 m	8.0 m	9.0. m	10.0 m
3.000	0.295	0.524	0.655	0.753	0.833	0.888	0.918	0.942	0.958	0.974
1.500	0.169	0.303	0.412	0.508	0.580	0.649	0.711	0.749	0.795	0.831
1.000	0.104	0.239	0.337	0.438	0.504	0.572	0.633	0.685	0.735	0.761
0.750	0.097	0.211	0.328	0.414	0.489	0.549	0.614	0.664	0.718	0.751
0.600	0.074	0.173	0.271	0.345	0.416	0.468	0.548	0.606	0.677	0.723
0.500	0.060	0.177	0.279	0.382	0.470	0.544	0.602	0.659	0.713	0.757

SR	11.0 m	12.0 m	13.0 m	14.0 m	15.0 m	16.0 m	17.0 m	18.0 m	19.0. m	20.0 m
3.000	0.976	0.978	0.982	0.988	0.992	0.994	0.994	0.994	0.994	0.994
1.500	0.863	0.884	0.908	0.928	0.942	0.958	0.966	0.970	0.980	0.986
1.000	0.803	0.815	0.857	0.882	0.902	0.916	0.924	0.938	0.950	0.958
0.750	0.785	0.811	0.859	0.871	0.883	0.897	0.917	0.932	0.938	0.952
0.600	0.765	0.799	0.835	0.857	0.876	0.896	0.908	0.912	0.928	0.938
0.500	0.793	0.825	0.835	0.867	0.884	0.896	0.916	0.930	0.938	0.946

4.2.3 CVT Rotation Simulation Results

The simulation results for the Monte Carlo simulations for the search behavior with CVT rotation enacted are presented in this section. The raw count data is displayed in fig. 4.9. The probability of detection at the discrete intervals are shown in fig. 4.10 and table 4.3. It can be seen that the highest speed ratio greatly increases the detection probability of the swarm; however, beyond the largest speed ratio the detection probability stays more constrained than that of the vector field.

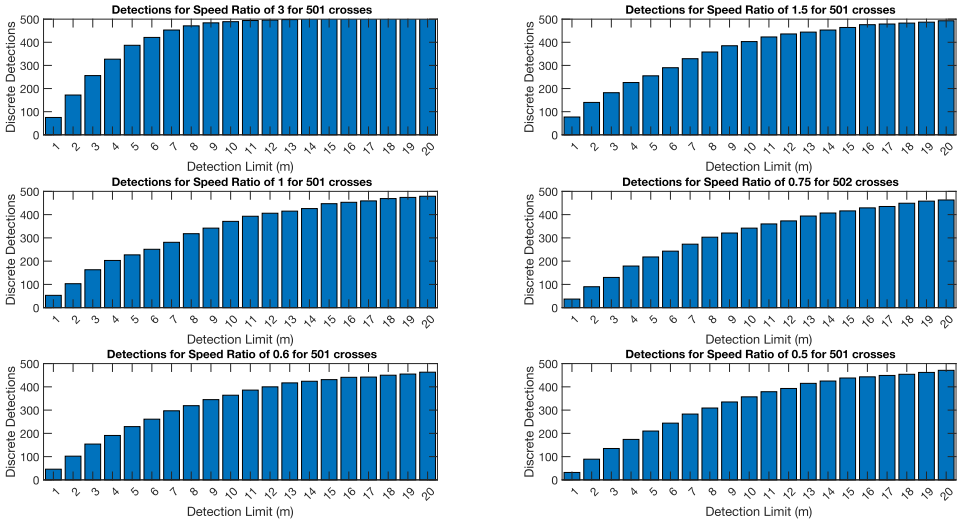


Figure 4.9: Raw counts of interactions for discrete distances for CVT Rotation

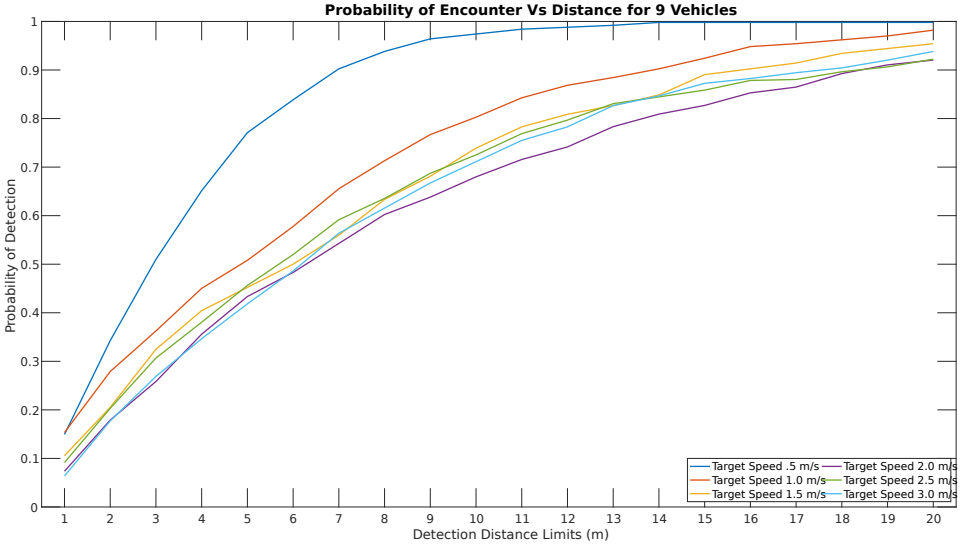


Figure 4.10: Probability of Interaction at Discrete Distances for CVT Rotation

Table 4.3: Probability of Interaction at Discrete Distances for Varying Adversary Speeds.

SR	1.0 m	2.0 m	3.0 m	4.0 m	5.0 m	6.0 m	7.0 m	8.0 m	9.0. m	10.0 m
3.000	0.149	0.343	0.510	0.651	0.771	0.839	0.902	0.938	0.964	0.974
1.500	0.153	0.279	0.363	0.450	0.508	0.578	0.655	0.713	0.767	0.803
1.000	0.106	0.205	0.325	0.404	0.452	0.500	0.560	0.633	0.681	0.739
0.750	0.074	0.179	0.258	0.356	0.433	0.483	0.543	0.602	0.638	0.680
0.600	0.092	0.203	0.307	0.380	0.456	0.520	0.592	0.635	0.687	0.725
0.500	0.064	0.177	0.269	0.347	0.418	0.486	0.564	0.616	0.667	0.711

SR	11.0 m	12.0 m	13.0 m	14.0 m	15.0 m	16.0 m	17.0 m	18.0 m	19.0. m	20.0 m
3.000	0.984	0.988	0.992	0.998	0.998	0.998	0.998	0.998	0.998	0.998
1.500	0.843	0.869	0.884	0.902	0.924	0.948	0.954	0.962	0.970	0.982
1.000	0.783	0.809	0.827	0.849	0.890	0.902	0.914	0.934	0.944	0.954
0.750	0.716	0.742	0.783	0.809	0.827	0.853	0.865	0.893	0.911	0.920
0.600	0.769	0.797	0.831	0.845	0.859	0.878	0.880	0.896	0.906	0.922
0.500	0.755	0.783	0.827	0.847	0.873	0.882	0.894	0.904	0.920	0.938

4.2.4 Stochastic Heading

The results for the Monte Carlo simulations for the search behavior with Stochastic Heading method are presented in this section. The raw count data is displayed in fig. 4.11. The probability of detection at the discrete intervals are shown in fig. 4.12 and table 4.4. The speed ratio has a similar effect as shown in the above methods. The lower speed ratios tend to merge to within the simulation accuracy.

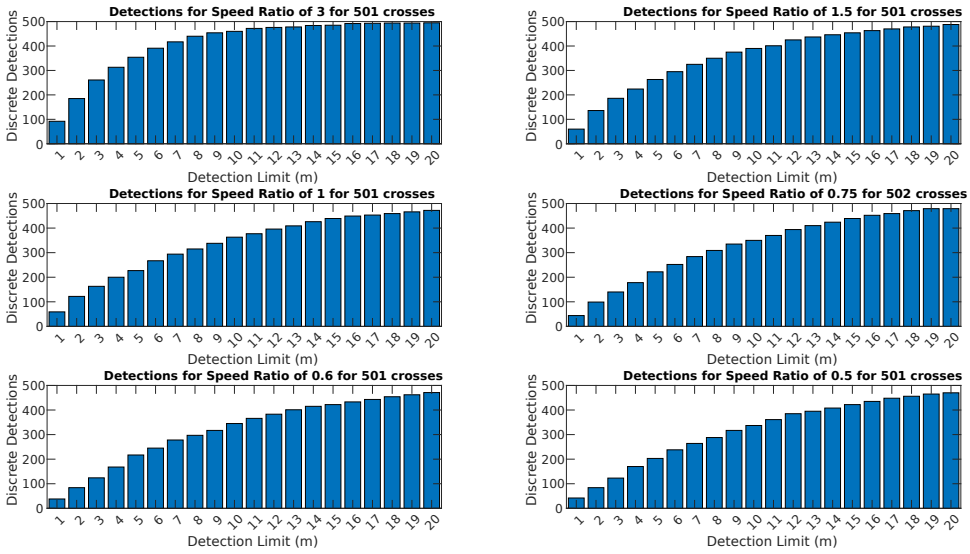


Figure 4.11: Raw counts of interactions for discrete distances for Stochastic Heading

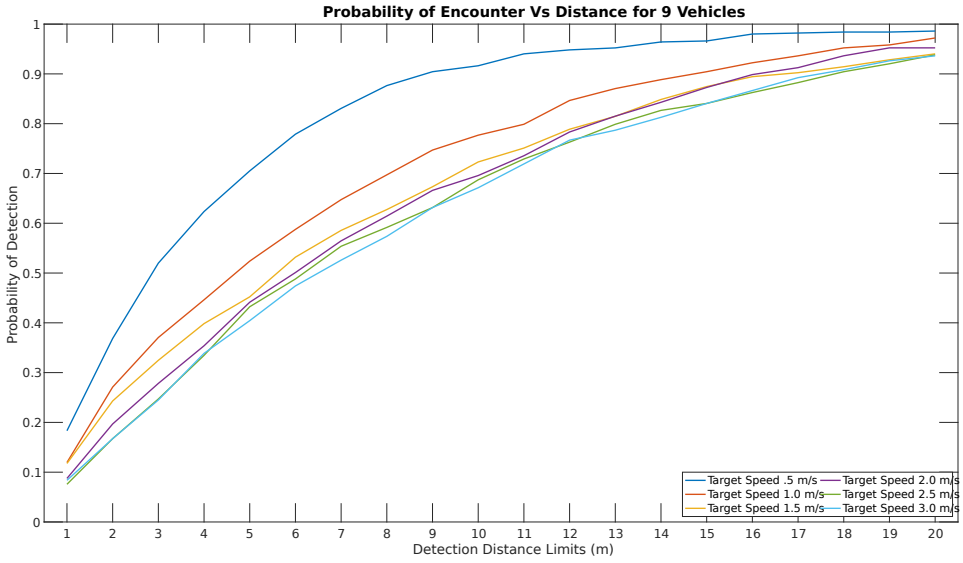


Figure 4.12: Probability of Interaction at Discrete Distances for Stochastic Heading

Table 4.4: Probability of Interaction at Discrete Distances for Varying Adversary Speeds.

SR	1.0 m	2.0 m	3.0 m	4.0 m	5.0 m	6.0 m	7.0 m	8.0 m	9.0. m	10.0 m
3.000	0.183	0.369	0.520	0.624	0.705	0.779	0.831	0.876	0.904	0.916
1.500	0.120	0.271	0.371	0.446	0.524	0.588	0.647	0.697	0.747	0.777
1.000	0.118	0.243	0.325	0.398	0.452	0.532	0.586	0.627	0.673	0.723
0.750	0.087	0.197	0.278	0.354	0.441	0.501	0.565	0.614	0.666	0.696
0.600	0.076	0.167	0.247	0.335	0.432	0.488	0.554	0.592	0.631	0.687
0.500	0.084	0.167	0.245	0.339	0.404	0.474	0.526	0.574	0.631	0.671

SR	11.0 m	12.0 m	13.0 m	14.0 m	15.0 m	16.0 m	17.0 m	18.0 m	19.0. m	20.0 m
3.000	0.940	0.948	0.952	0.964	0.966	0.980	0.982	0.984	0.984	0.986
1.500	0.799	0.847	0.871	0.888	0.904	0.922	0.936	0.952	0.958	0.972
1.000	0.751	0.789	0.815	0.849	0.875	0.894	0.902	0.914	0.928	0.940
0.750	0.736	0.783	0.815	0.843	0.873	0.899	0.913	0.936	0.952	0.952
0.600	0.729	0.763	0.799	0.827	0.841	0.863	0.882	0.904	0.920	0.938
0.500	0.719	0.767	0.787	0.813	0.841	0.867	0.892	0.908	0.926	0.936

4.2.5 Stochastic Free

The results for the Monte Carlo simulations for the search behavior with Stochastic Free method are presented in this section. The raw count data is displayed in fig. 4.13. The probability of detection at the discrete intervals are shown in fig. 4.14 and table 4.5. The detection probability differs very little from the Stochastic Heading method.

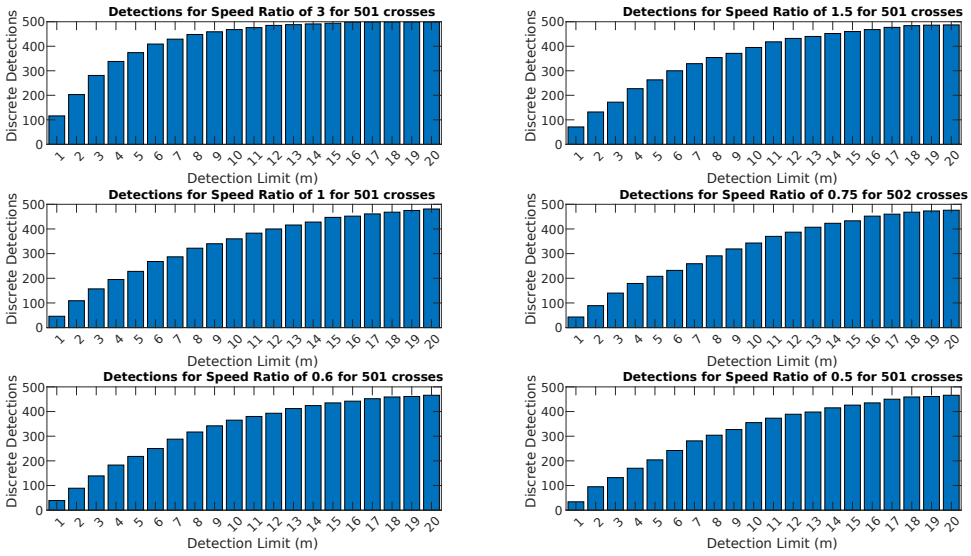


Figure 4.13: Raw counts of interactions for discrete distances for Stochastic Free

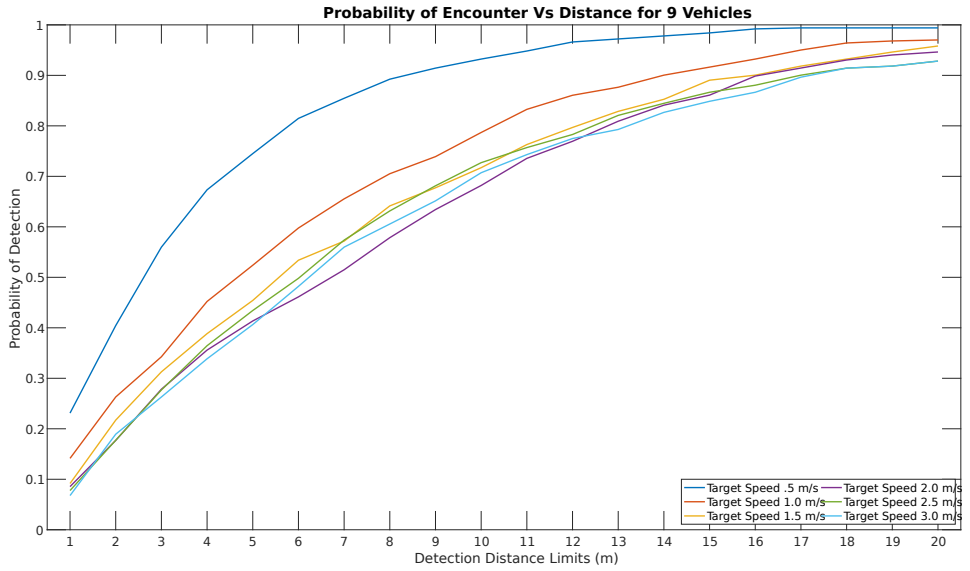


Figure 4.14: Probability of Interaction at Discrete Distances for Stochastic Free

Table 4.5: Probability of Interaction at Discrete Distances for Varying Adversary Speeds.

SR	1.0 m	2.0 m	3.0 m	4.0 m	5.0 m	6.0 m	7.0 m	8.0 m	9.0. m	10.0 m
3.000	0.231	0.404	0.560	0.673	0.745	0.815	0.855	0.892	0.914	0.932
1.500	0.141	0.263	0.343	0.452	0.524	0.598	0.655	0.705	0.739	0.787
1.000	0.092	0.217	0.313	0.388	0.454	0.534	0.572	0.641	0.677	0.717
0.750	0.085	0.177	0.278	0.356	0.414	0.461	0.515	0.579	0.634	0.682
0.600	0.078	0.177	0.277	0.365	0.434	0.498	0.574	0.631	0.681	0.727
0.500	0.068	0.189	0.263	0.339	0.406	0.482	0.560	0.606	0.651	0.707
SR	11.0 m	12.0 m	13.0 m	14.0 m	15.0 m	16.0 m	17.0 m	18.0 m	19.0. m	20.0 m
3.000	0.948	0.966	0.972	0.978	0.984	0.992	0.994	0.994	0.994	0.994
1.500	0.833	0.861	0.876	0.900	0.916	0.932	0.950	0.964	0.968	0.970
1.000	0.763	0.797	0.829	0.853	0.890	0.900	0.918	0.932	0.946	0.958
0.750	0.736	0.769	0.809	0.841	0.861	0.899	0.915	0.930	0.940	0.946
0.600	0.757	0.783	0.821	0.845	0.867	0.880	0.900	0.914	0.918	0.928
0.500	0.743	0.775	0.793	0.827	0.849	0.867	0.896	0.914	0.918	0.928

4.3 Probabilistic Sensor Simulation Results

In the section, results are presented from the pSonarSimDetect app and the pKalmanSolutionGen app. The results are presented from the Monte Carlo simulations with 500 adversary crossings for each speed ratio.

4.3.1 Probability of Detection and Solution Generation

Each search method was executed using the probabilistic detection application pSonarSimDetect. pSonarSimDetect produced detection reports used by the pKalmanSolutionGen app for solution development. The pKalmanSolutionGen app was employed on the shoreside MOOSDB. The results in this section show the probability of detec-

tion using the sonar sim app along with the probability that a solution is produced. In order to generate a solution at least two detection reports that are a minimum of 10m apart are required. The distance between detections is the only restriction pKalmanSolutionGen places on using a detection report. The detection reports can be generated by the same vehicle as long as the detection reports are greater than 10m apart. This allows a moving swarm vehicle to report a detection on a second encounter with the adversary vehicle. Therefore, the number of solutions generated is a mark of the swarm's ability to obtain more than one detection over space. Thus, instead of the swarm simply producing an expanding area of uncertainty plot, it can produce a solution to be used for follow on search and target tracking.

While target speed should not impact the detection probability for the stationary swarm using the discrete detection application uFldSearchDetect, it does impact detection probability for pSonarSimDetect. The sonar simulation application has a designated frequency; therefore, the number of detection opportunities increases as the time the adversary exists within the sonar detection radius increases. Thus, the sonar application will get more opportunities for detection at slower adversary speeds. This result can be seen in fig. 4.15. Further, this additional factor impacts the results for each search method. In fact, the frequency limitation on the sonar application can have a greater impact on the search methods employed in this thesis. The speed ratio tested relates the maximum speed of the swarm to the adversary speed, but does not account for the direction the speed is in. Therefore, if the swarm vehicle's velocity is in the opposite direction of the adversary's velocity, the exposure time the sonar has to the adversary vehicle is even less than the stationary swarm. So while the discrete detection plots using uFldSearchDetect show a much higher probability to encounter the adversary with a moving swarm, the additional encounters do not fully translate to increased sonar

detection. The sonar application for this thesis was set at 2Hz frequency, therefore increasing this frequency would minimize this effect.

The detection radius limit using the pSonarSimDetect app was approximately 10m assuming a 40m deep target. Therefore, the data presented in this section reflect detections that were Gaussian with a max detection radius of 10m. The detection probability results in this section do not always decrease monotonically. However, if the sonar results for these methods are compared to the discrete detections results, it can be seen that in the under 10 m detection threshold these same inconsistencies exist. The variability between the speed ratios at the less than 10m detection threshold were small and can be attributed to the overall accuracy limitations of the Monte Carlo Simulation.

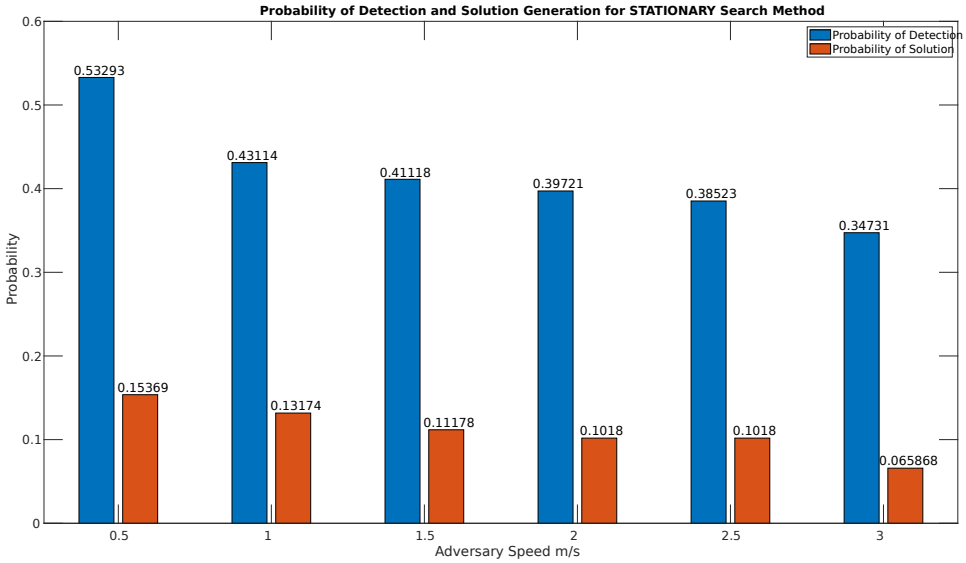


Figure 4.15: Probabilistic Sensor Detections for Stationary Swarm

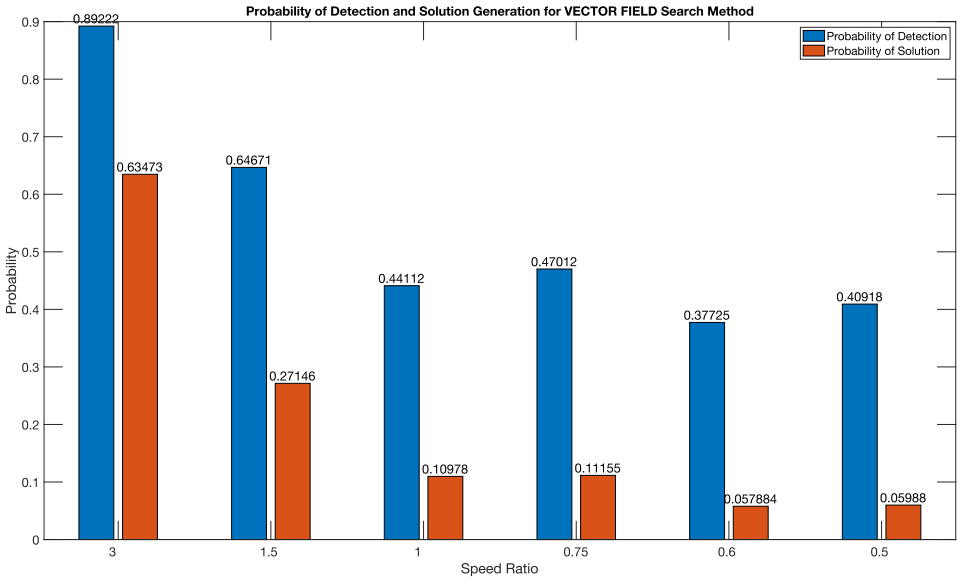


Figure 4.16: Probabilistic Sensor Detections for Swarm with Vector Field

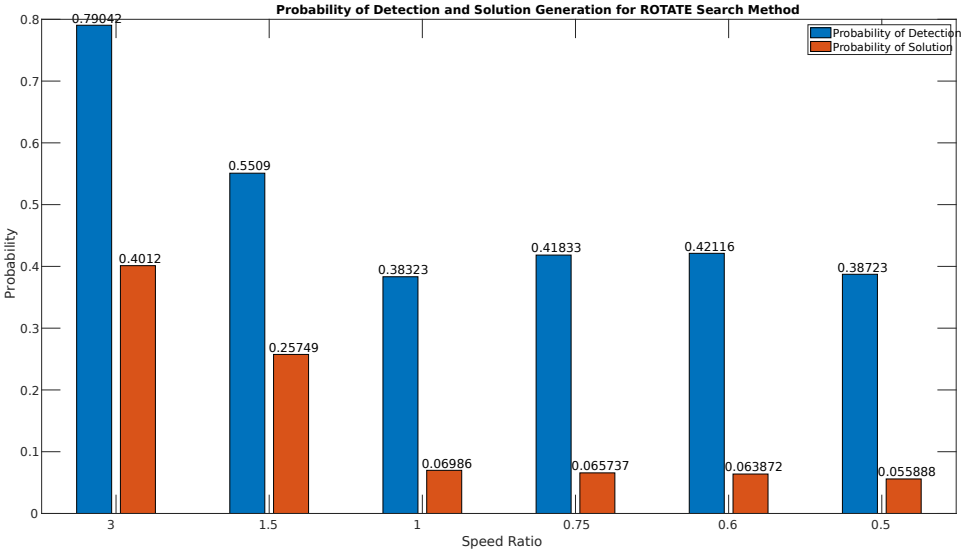


Figure 4.17: Probabilistic Sensor Detections for Swarm with CVT Rotation

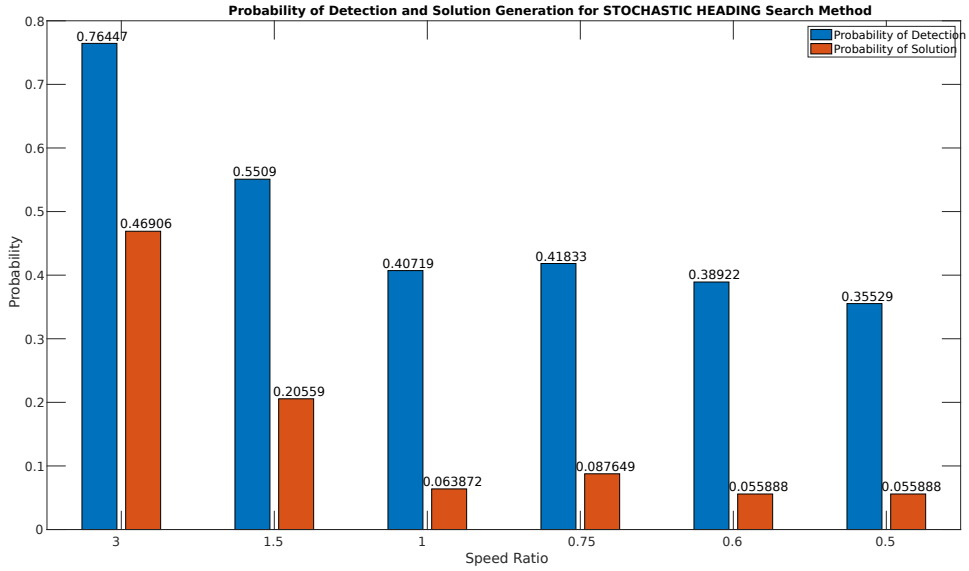


Figure 4.18: Probabilistic Sensor Detections for Swarm with Stochastic Heading

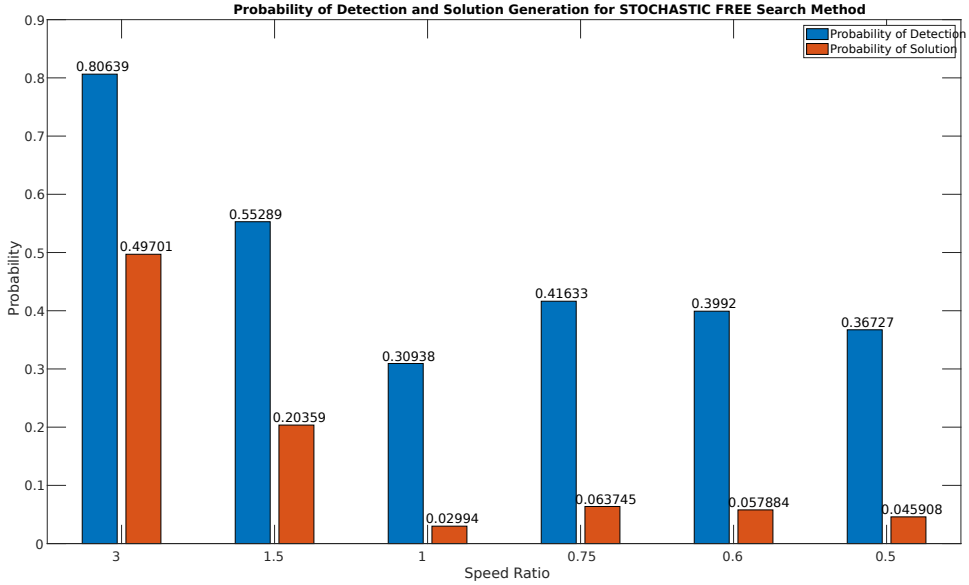


Figure 4.19: Probabilistic Sensor Detections for Swarm with Stochastic Free Method

4.3.2 Solution Accuracy

The solutions generated from the pKalmanSolutionGen app can vary in accuracy. The quality of the estimates produced is reliant on the quality of the data being supplied. This section provides analysis of the quality of the estimates produced for each search method.

Stationary Swarm

The stationary swarm produced the speed and heading estimate results shown in figs. 4.20 and 4.21. The speed estimate produced was the closest estimate to actual speed of any of the search methods below. Speed is estimated from the location and time of detection reports. Therefore, the swarm speed can influence the difference in detection reports generated by swarm vehicles. The stationary swarm did not have this error added to the speed and heading estimates. This lead to fairly accurate speed and heading estimates.

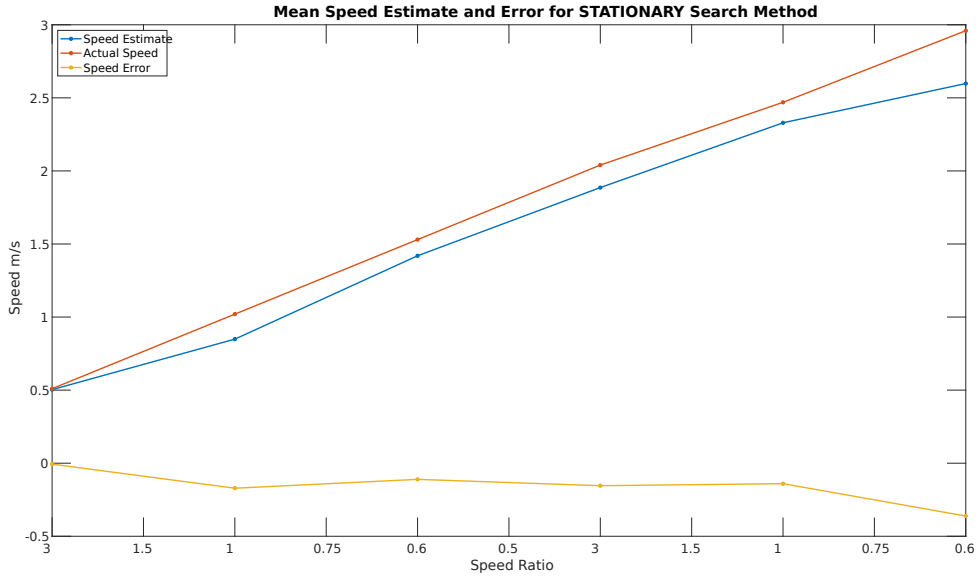


Figure 4.20: Speed Estimate

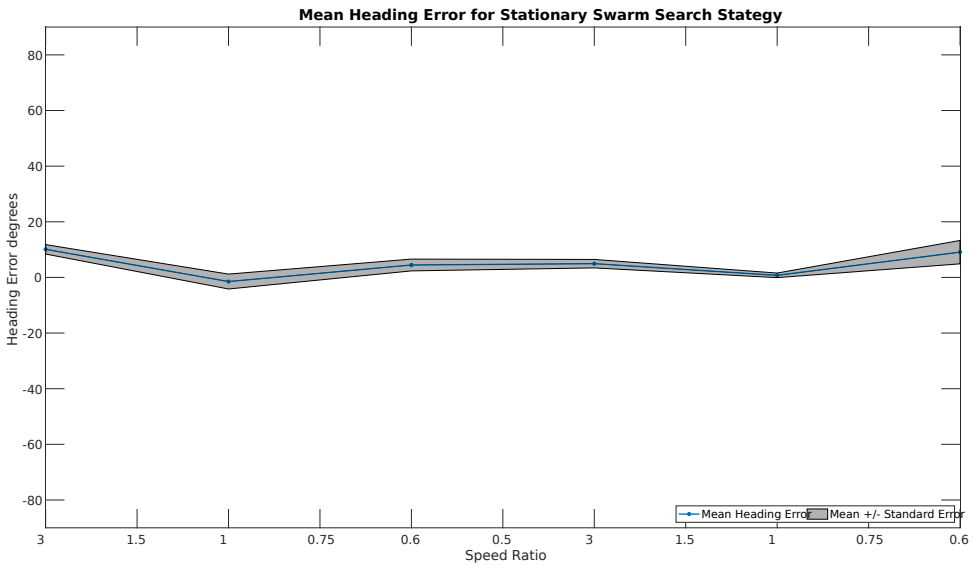


Figure 4.21: Heading Error

CVT Rotatation

The CVT Rotation search method produced the results shown in figs. 4.22 and 4.23. The Heading and speed error for estimates where the adversary speed was less than the swarm speed tended to be more accurate. As the speed ratio decreased the speed estimates became more poor and were continuously under estimated.

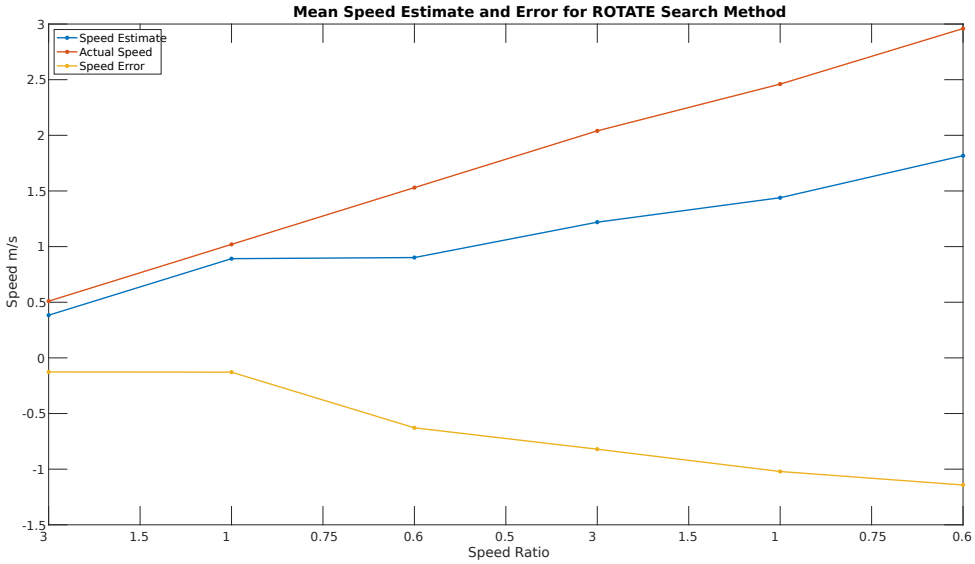


Figure 4.22: CVT Rotation Speed Estimate

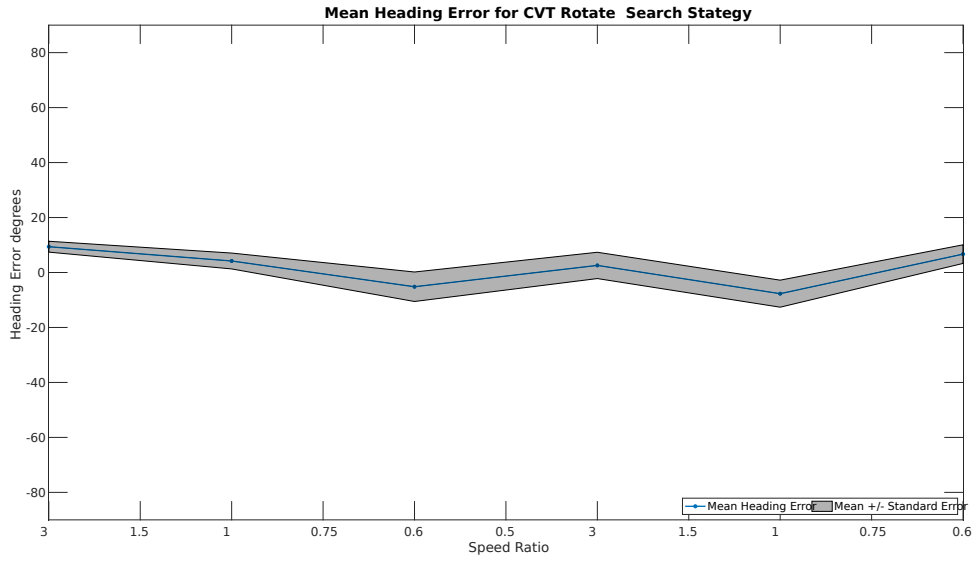


Figure 4.23: CVT Rotation Heading Error

Stochastic Heading

Figs. 4.25 and 4.24 shows the speed and heading estimate results for the Stochastic Heading method. The results obtained show that the speed estimate gets consistently

under estimated as the target speed increases. However, the heading estimate had a relatively consistent average error.

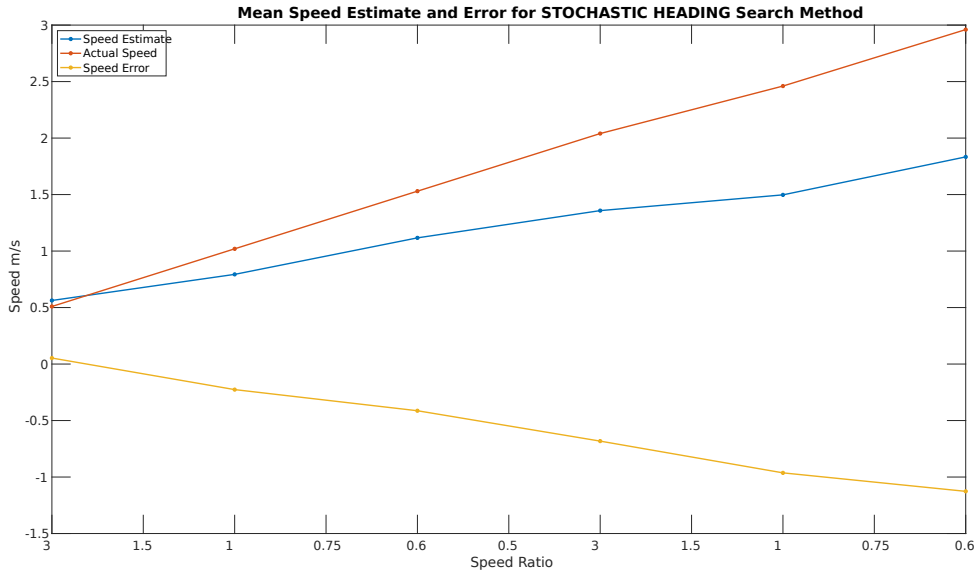


Figure 4.24: Stochastic Heading Speed Estimate

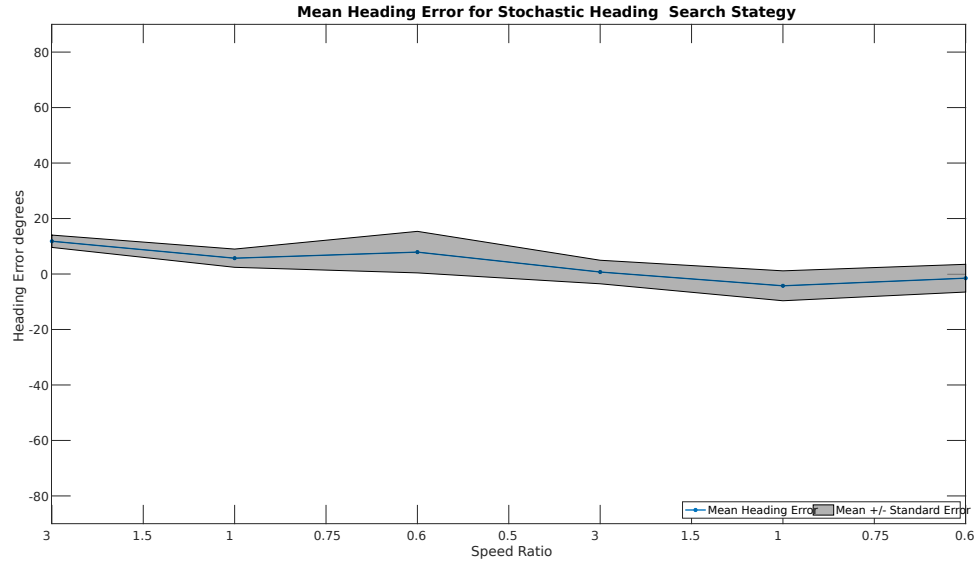


Figure 4.25: Stochastic Heading Heading Error

Stochastic Free

The speed estimates and heading error averaged over each speed ratio for the Stochastic Free search method is shown in figs. 4.27 and 4.26. The results obtained are similar to the stochastic heading. The speed estimate gets consistently under estimated as target speed increases, while the heading error remains low.

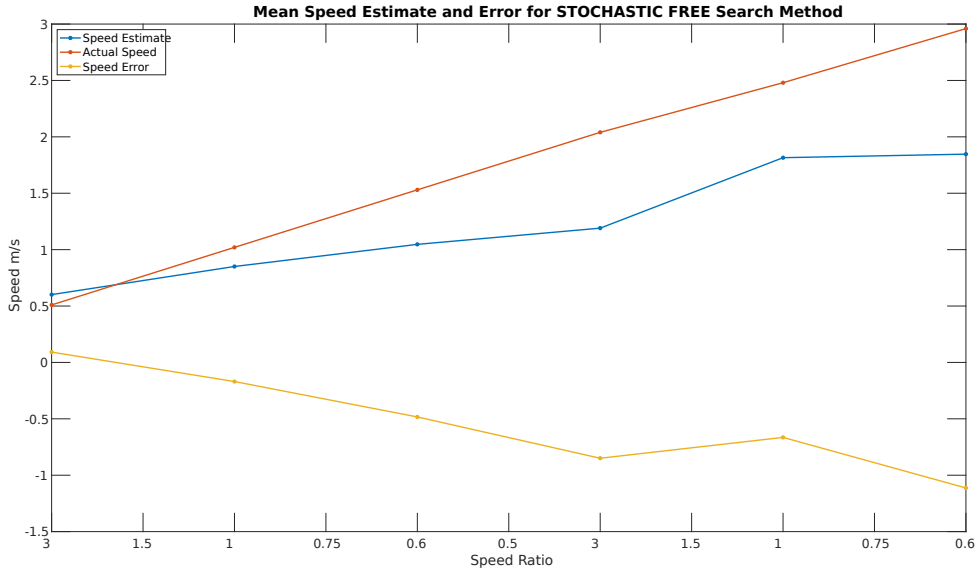


Figure 4.26: Stochastic Free Speed Estimate

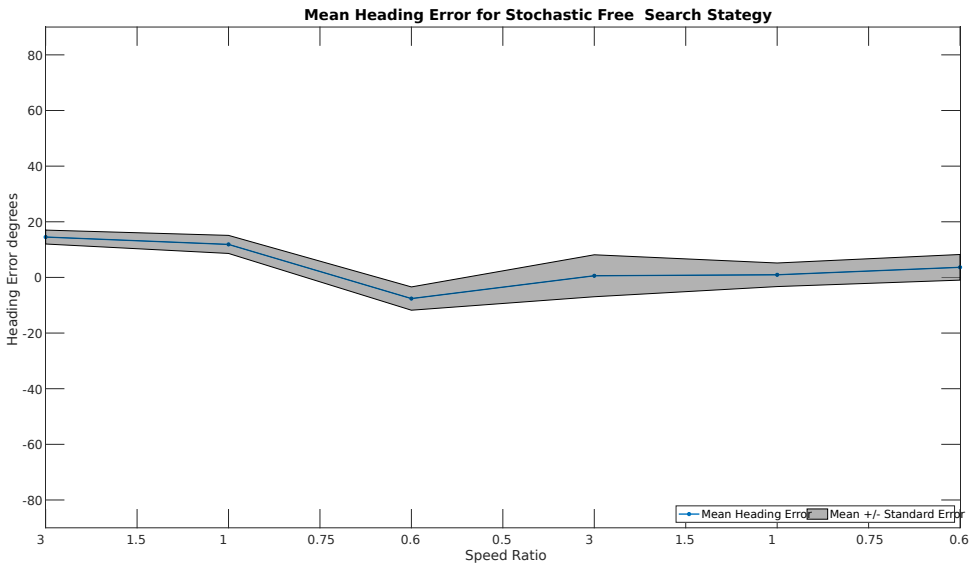


Figure 4.27: Stochastic Free Heading Error

Vector Field

The Vector Field speed and heading estimates are depicted in fig. 4.29 and 4.28. The results followed the trend set by the stochastic related search methods. However, the heading error average is lower than any other search method.

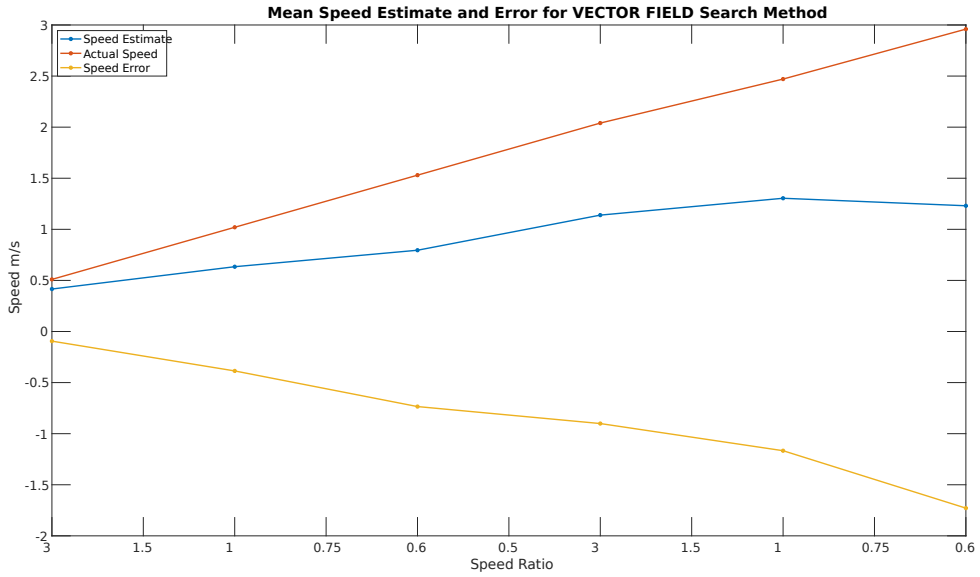


Figure 4.28: Vector Field Speed Estimate

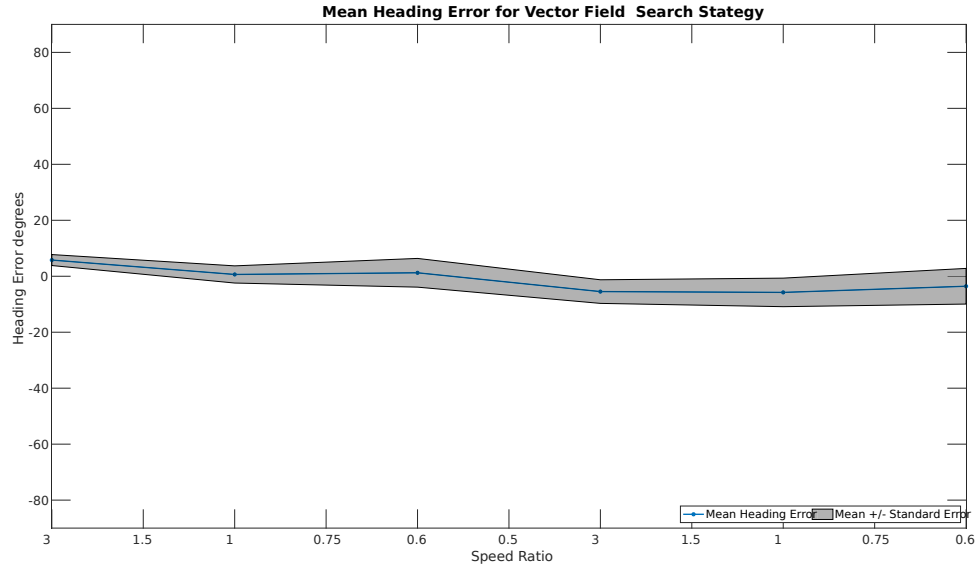


Figure 4.29: Vector Field Heading Error

4.3.3 Simulation Data Conclusions

In this section the results for each search method and speed ratio are averaged and compared. The individual speed ratio comparisons can be found in the appendix. From the results it can be seen that the the vector field search method produces the highest possible discrete detection probability. Further, the Vector Field method produces the greatest balance between discrete detection, probabilistic detection, and solution generation. Although the Vector Field method generated more detections and solutions, the solutions had the greatest average error over speed and heading. It is clear that any search method that involves motion increases the ability of the swarm to detect the passing adversary. However, the quality of the speed and heading estimates produced by the swarm deteriorate. Therefore, it is assumed that if there were adequate vehicles to provide complete coverage of the region without movement, the stationary swarm would produced the most accurate solution estimate. However, in the coverage limited problem, methods that involve motion produce the best results. The margin of error, when the various speed ratios are averaged, prevent a definitive second choice from being chosen between the Stochastic Free, Stochastic Heading, and CVT Rotation methods. If the the speed ratio is known, then the most effective search method can be discerned. These direct speed ratio comparisons are shown in the appendix.

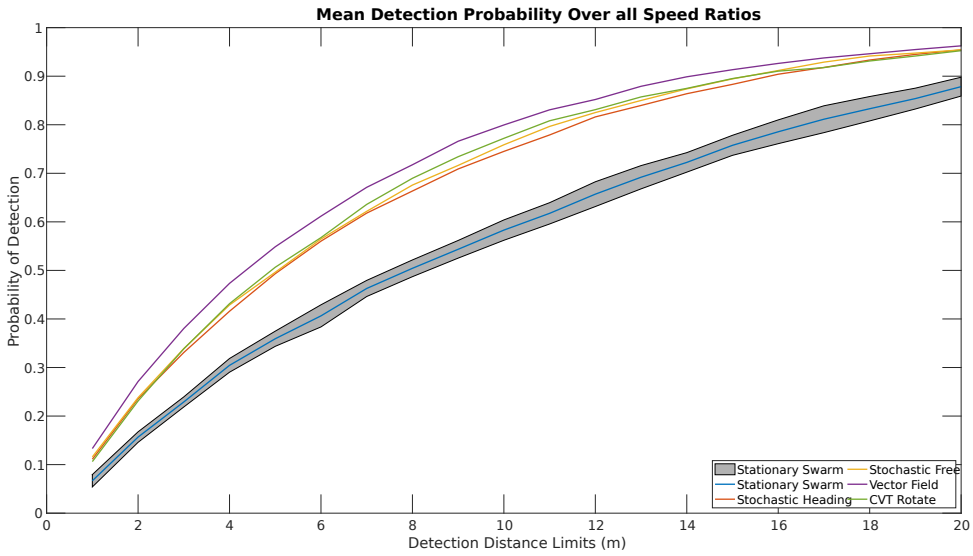


Figure 4.30: Mean Probability of Detection for all Methods

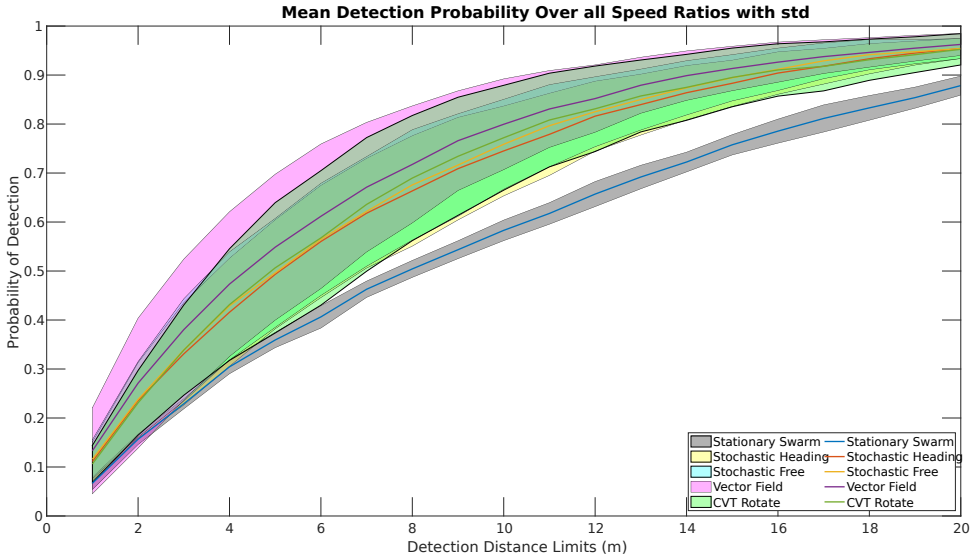


Figure 4.31: Mean Probability of Detection for all Methods with Standard Deviation

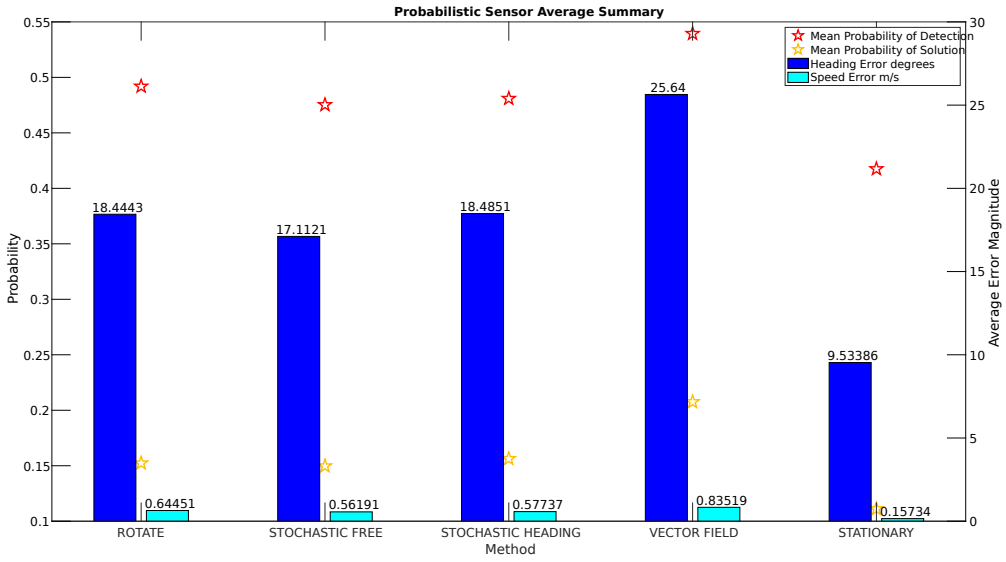


Figure 4.32: Probabilistic Sensor Results Summary

4.4 Hardware Test

The results for the hardware test scenario described in Chapter 3 are presented for each search method in this section. For each search method a simulated adversary vehicle crossed the hardware test region five times. The number of crossings was set at five in order to balance the demonstration of detection probability with the physical run time on the Charles River. The river traffic constrained the length of time each scenario was able to run. The swarm speed was set to 1 m/s with an the simulated adversary speed also set to 1 m/s. Therefore, the speed ratio of the hardware test was 1.0. The scenario was conducted in the 150m by 150m cover area defined in Chapter 3. Fig. 4.33 shows an example of a 7 vehicle CVT in the cover region. The rings indicate the 10 m and 20 m detection rings. It can be seen that the 20 m ring provides far more area coverage than the scenario tested in simulation.

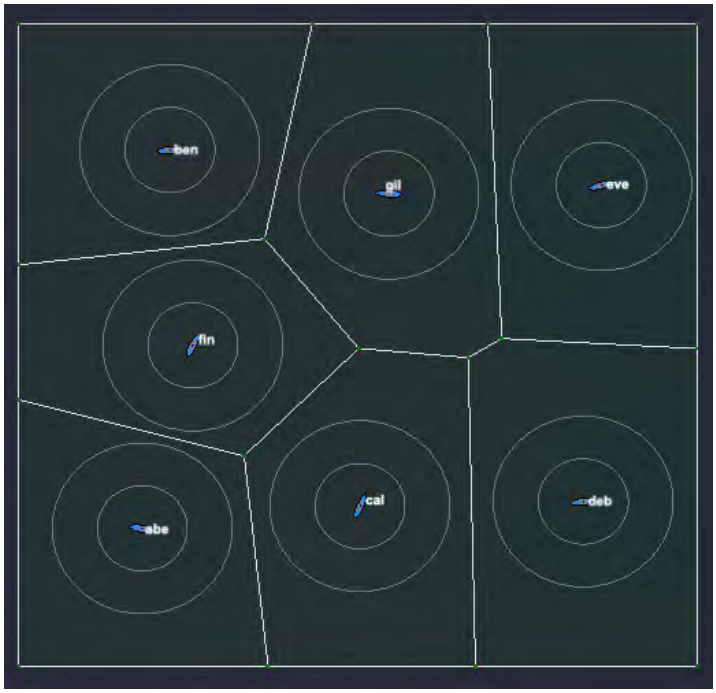


Figure 4.33: 7 Vehicle CVT in the 150m by 150m cover region with 10m and 20m detection rings.

The results for the number of detections of the simulated vehicle for each search method are presented here for completeness. However, the purpose of the hardware test was to demonstrate the deployability of the search algorithms on an autonomous platform. It was not the goal to replicate the number of batch simulations conducted in the Monte Carlo simulations. Therefore, the probability of detection for each search method should be inferred from the Monte Carlo simulation. The data presented here in the hardware section should be viewed as validation for the deployability of the system used in simulation.

4.5 Hardware Results

Fig. 4.34 shows the fleet of 7 Heron vehicles on the Charles River deployed for the test scenario.

Figure 4.34: Heron Vehicles on Charles River

4.5.1 CVT Rotation

The image in fig. 4.35 shows the actual GPS position history of the 7 vehicle swarm on the Charles River. The green vehicle tails are shown in the image to demonstrate the path each vehicle maintained during testing. It can be seen that the vehicles accurately achieved a CVT of the region and executed the CVT Rotation behavior. The small disruptions to a perfectly circular path is attributed to physical condition on the river as well as GPS navigation. The overall behavior was executed successfully.

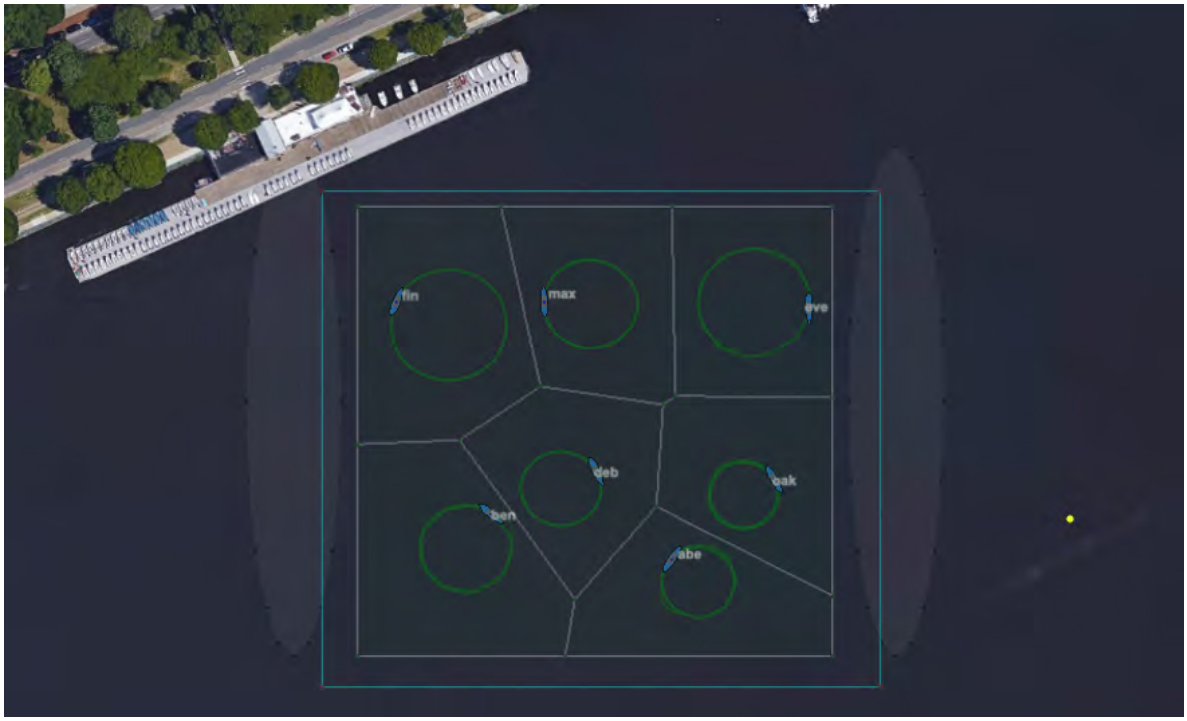


Figure 4.35: CVT Rotation search method operating on the Heron USVs as seen from pMarineViewer during the hardware test. The inner square region is the established 150m by 150m cover region. The Outer square region is the operational safety boundary established for the hardware test. The east and west ovals are the start and finish regions for the simulated adversary vehicle.

Figs. 4.36 and 4.37 show the detection counts and detection probability of the simulated adversary over five region crossings. A similar trend can be seen in the simulated data. Table 4.6 shows the results for the simulated sonar detection and solution generation.

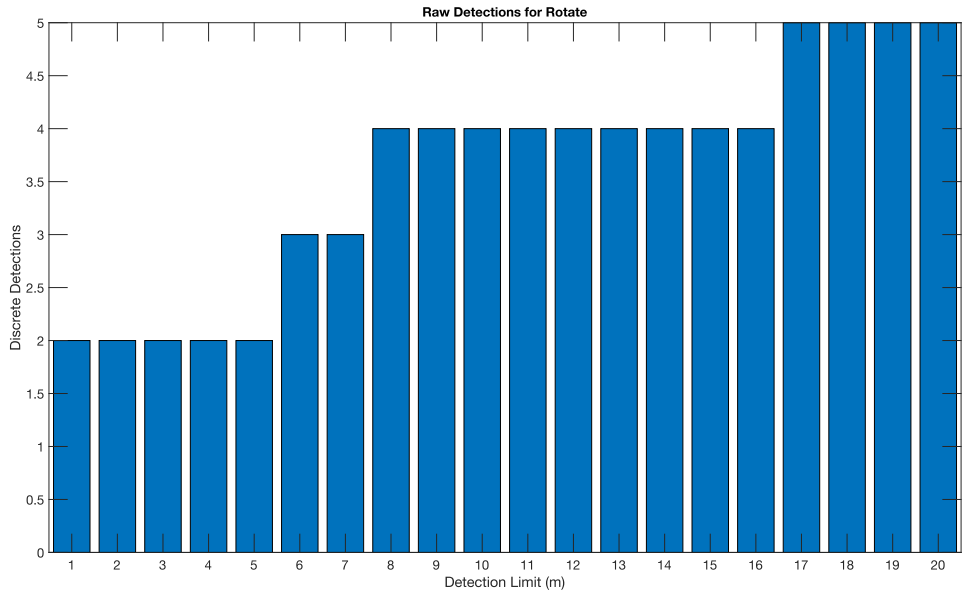


Figure 4.36: Bar Graph of Detections for CVT Rotation Method on Hardware

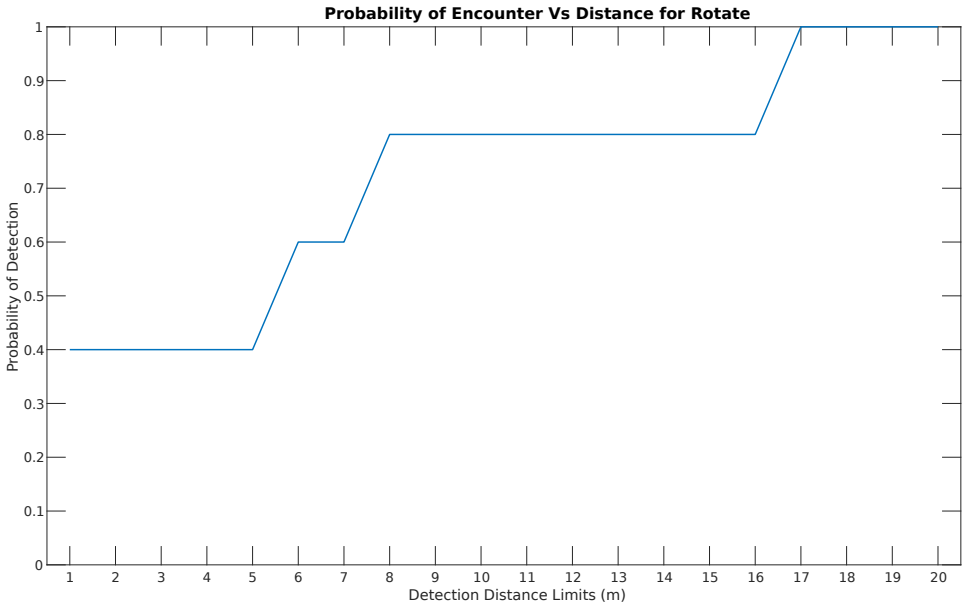


Figure 4.37: Detection Probability plot for CVT Rotation Method on Hardware

Sonar Detections	2
Number of Solutions	1
Heading Error	16.75 degrees
Speed Error	-.017 m/s

Table 4.6: Simulated Sonar Sensor Results for CVT Rotation Hardware Test.

4.5.2 Stochastic Heading

The image in fig. 4.38 shows the actual GPS position history of the 7 vehicle swarm on the Charles River while executing the Stochastic Heading search method. The green vehicle trails are shown in the image to demonstrate the path each vehicle maintained during testing. The region was first partitioned into a CVT diagram based off vehicle position. Then, each vehicle executed the stochastic Heading search method within its associated partition. The image shows the flower shaped pattern that the stochastic heading search method produces. However, it can be seen that a few of the vehicles did not maintain a perfectly continuous heading while attempting to maintain the ordered heading. This can occur when the vehicle's perceived heading is unstable, which can be caused by a weak GPS connection or poor IMU heading depending on the current mode for heading estimation. The overall behavior was executed successfully.

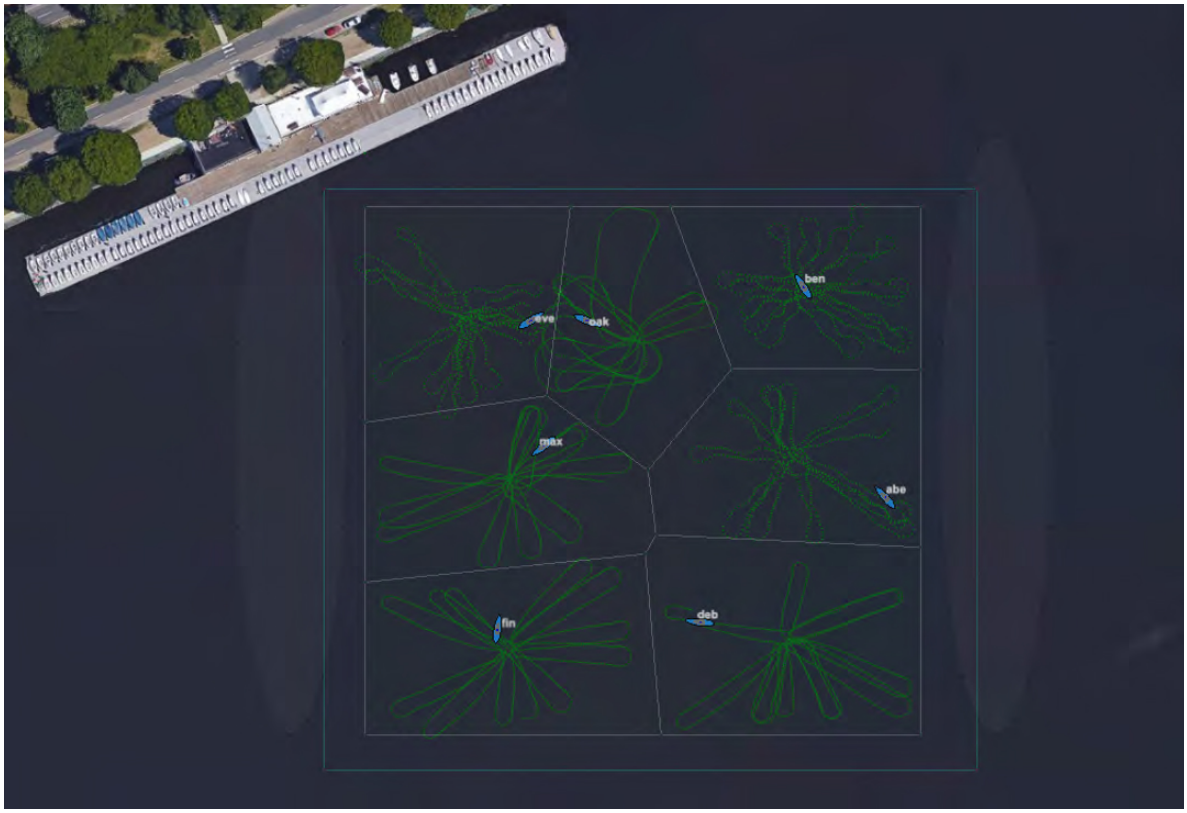


Figure 4.38: Stochastic Heading search method operating on the Heron USVs as seen from pMarineViewer during the hardware test. The inner square region is the established 150m by 150m cover region. The Outer square region is the operational safety boundary established for the hardware test. The east and west ovals are the start and finish regions for the simulated adversary vehicle.

Figs. 4.39 and 4.40 show the detection counts and detection probability of the simulated adversary over five region crossings. A similar trend can be seen in the simulated data. Table 4.7 shows the results for the simulated sonar detection and solution generation.

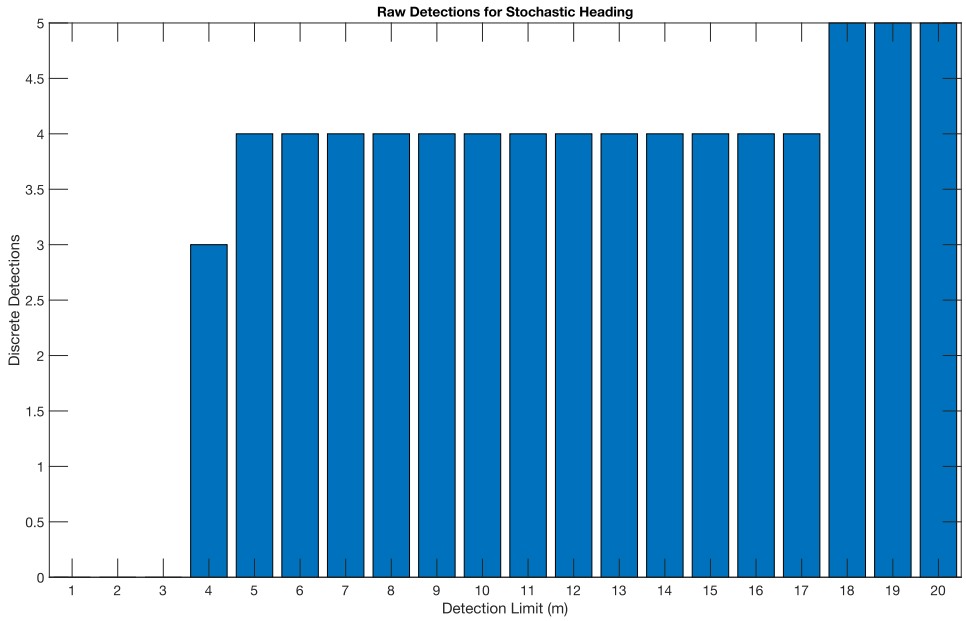


Figure 4.39: Bar Graph of Detections for Stochastic Heading Method on Hardware

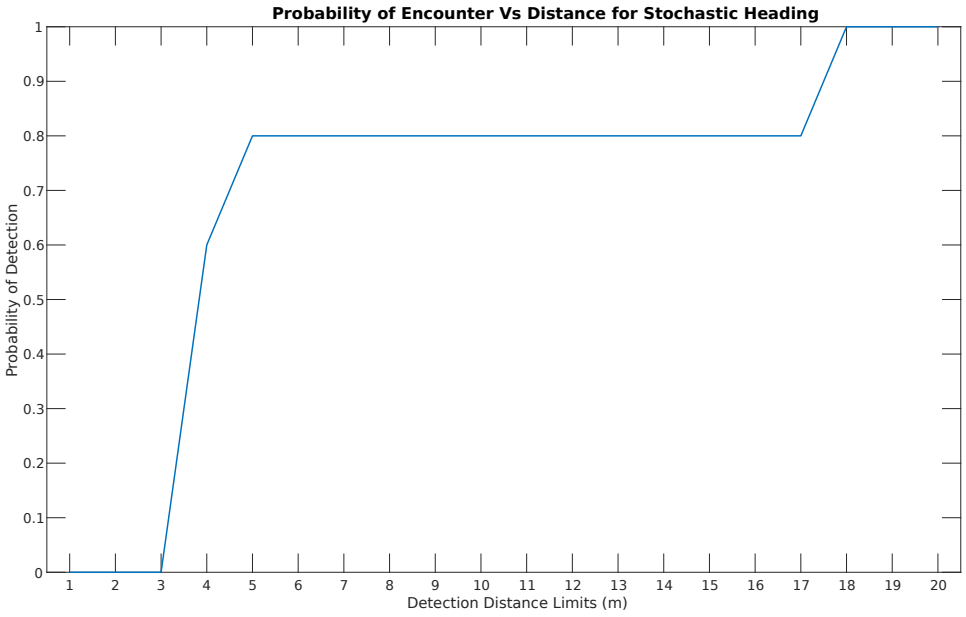


Figure 4.40: Detection Probability plot for Stochastic Heading Method on Hardware

`Sonar Detections`	2
`Number of Solutions`	1
`Heading Error`	-13.11651 degrees
`Speed Error`	0.382 m/s

Table 4.7: Simulated Sonar Sensor Results for Stochastic Heading Hardware Test.

4.5.3 Stochastic Free

The image in fig. 4.41 shows the actual GPS position history of the 7 vehicle swarm on the Charles River while executing the Stochastic Free search method. The green vehicle trails are shown in the image to demonstrate the path each vehicle maintained during testing. The region was first partitioned into a CVT diagram based off vehicle position. Then, each vehicle executed the stochastic free search method within its associated partition. Any boundary that was penetrated got quickly corrected. The overall behavior was executed successfully.

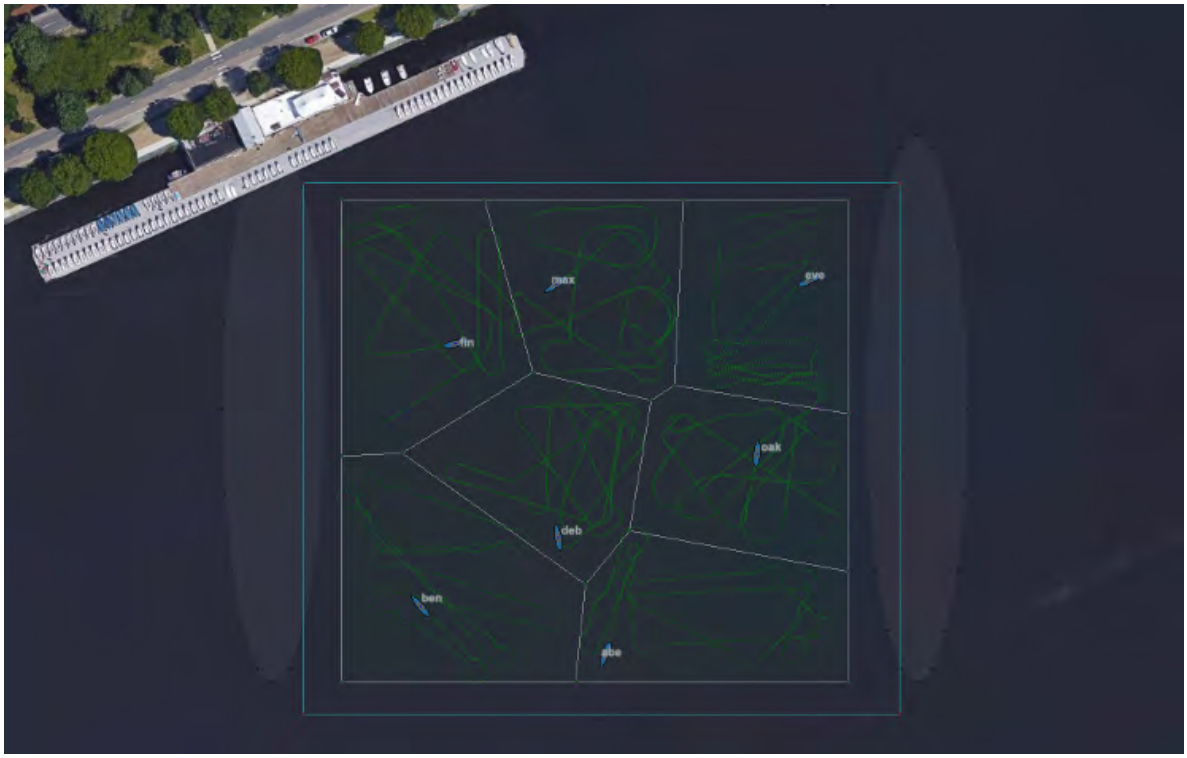


Figure 4.41: Stochastic Free search method operating on the Heron USVs as seen from pMarineViewer during the hardware test. The inner square region is the established 150m by 150m cover region. The Outer square region is the operational safety boundary established for the hardware test. The east and west ovals are the start and finish regions for the simulated adversary vehicle.

Figs. 4.42 and 4.43 show the detection counts and detection probability of the simulated adversary over five region crossings. A similar trend can be seen in the simulated data. Table 4.8 shows the results for the simulated sonar detection and solution generation.

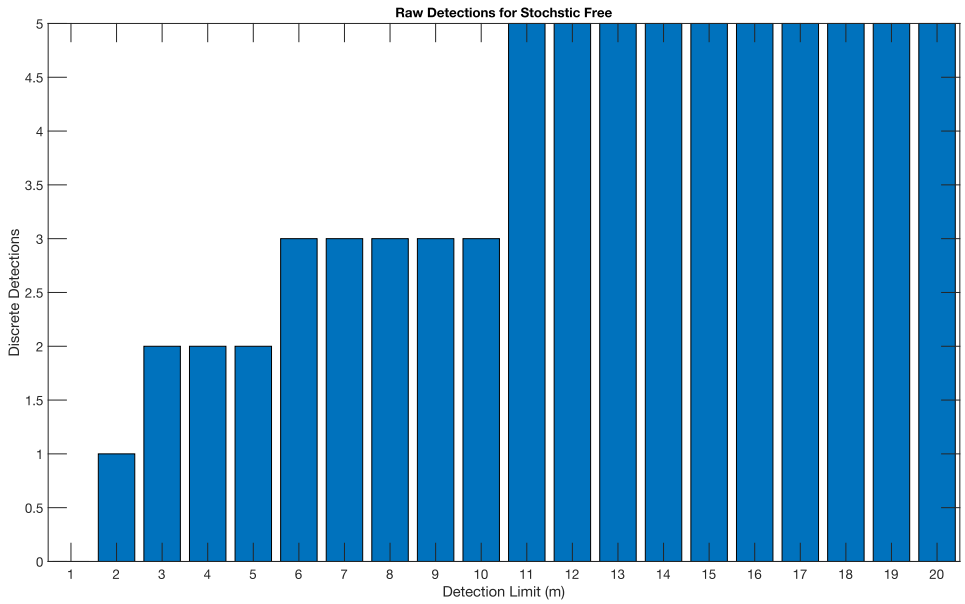


Figure 4.42: Bar Graph of Detections for Stochastic Free Method on Hardware

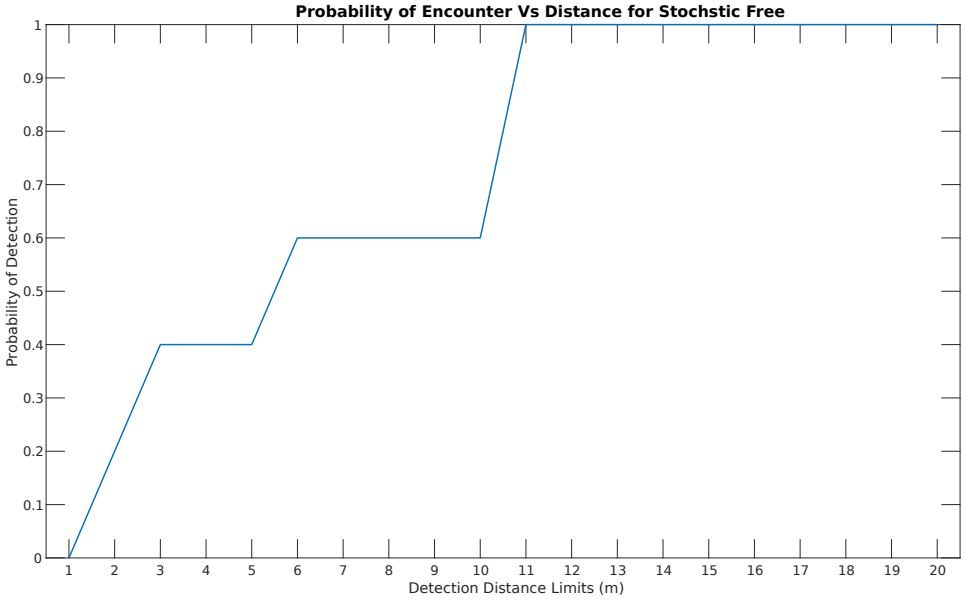


Figure 4.43: Detection Probability plot for Stochastic Free Method on Hardware

Sonar Detections	2
Number of Solutions	1
Heading Error	49.47068 degrees
Speed Error	-.22034 m/s

Table 4.8: Simulated Sonar Sensor Results for Stochastic Free Hardware Test.

4.5.4 Vector Field

The image in fig. 4.44 shows the actual GPS position history of the 7 vehicle swarm on the Charles River. The green vehicle trails show the path the vehicle has taken to its current location. The image shows the Voronoi behavior operating alongside the Vector Field behavior. An example of what occurs when the vector field behavior pushes a swarm vehicle beyond the activation radius of the Voronoi behavior can be seen on the vehicle Fin. Fin is in the southwest region and is steering against the vector field to regain its Voronoi partition. Overall, the Vector Field search method operated identically on the hardware as it did in simulation.

Figure 4.44: Vector field and Voronoi behavior operating on the Heron USVs as seen from pMarineViewer during the hardware test. The inner square region is the established 150m by 150m cover region. The Outer square region is the operational safety boundary established for the hardware test. The east and west ovals are the start and finish regions for the simulated adversary vehicle.

Figs. 4.45 and 4.46 show the detection counts and detection probability of the simulated adversary over five region crossings. A similar trend can be seen in the simulated data. Table 4.9 shows the results for the simulated sonar detection and solution generation.

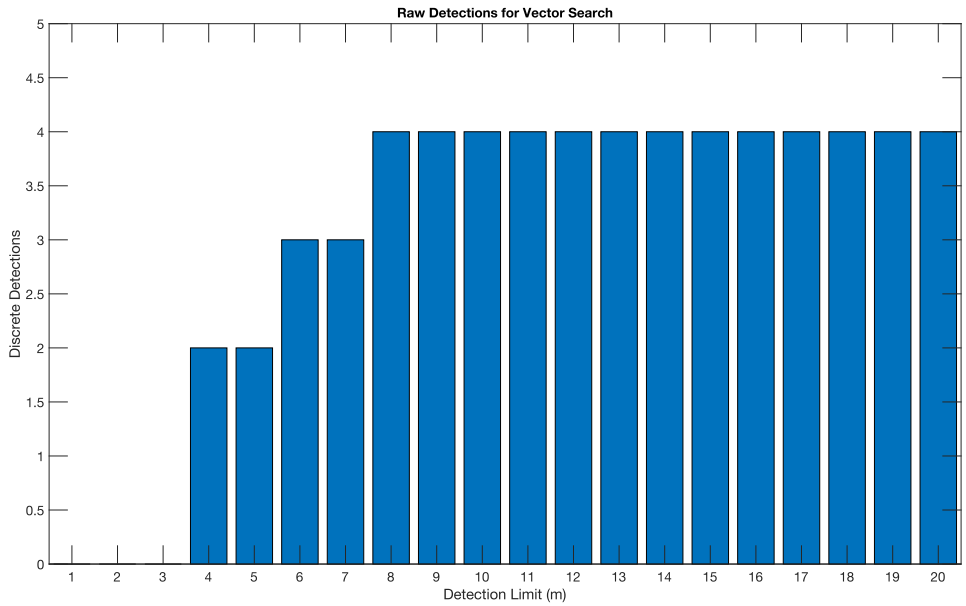


Figure 4.45: Bar Graph of Detections for Vector Field Method on Hardware

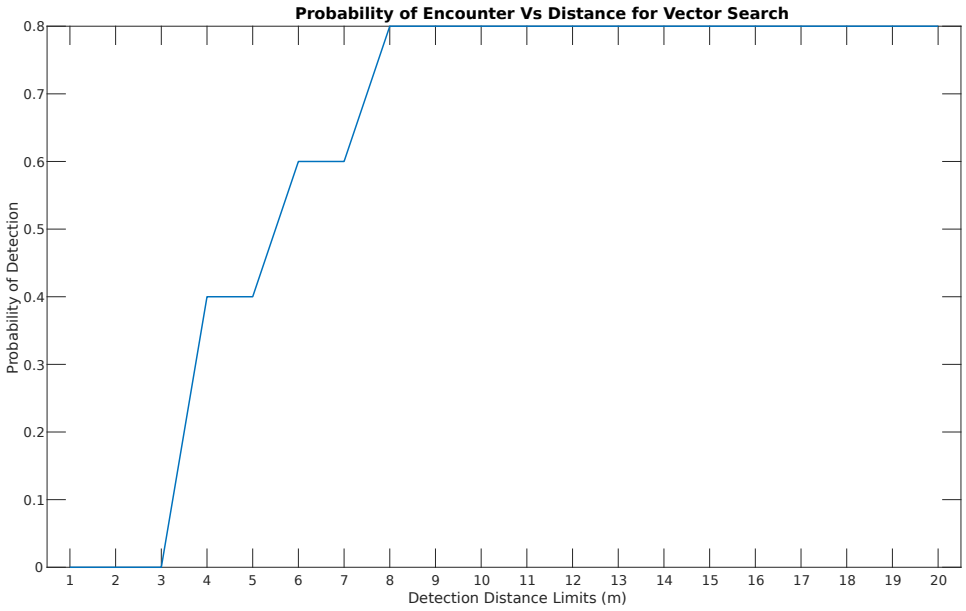


Figure 4.46: Detection Probability plot for Vector Field Method on Hardware

Sonar Detections	3
Number of Solutions	0
Heading Error	N/A
Speed Error	N/A

Table 4.9: Simulated Sonar Sensor Results for Vector Field Hardware Test.

4.5.5 Hardware Data Conclusions

The hardware test successfully demonstrated the deployability of the algorithms presented in this thesis. While the detection results are limited, the similarity to the results achieved through simulation can be seen. One notable difference is that the Vector Field method performed the worst in detection capability. However, this is likely due to the limited number of crossings and the asymmetry of the CVT for the other methods. Therefore, the hardware test cannot be relied upon to determine search method effectiveness. Rather, the hardware test demonstrated that a swarm of real world ASVs can be effectively be deployed as a cover and search team. Further, it was demonstrated that developing swarm algorithms, testing through Monte Carlo simulation, and deploying aboard real world vehicles can efficiently be done in the MOOS-IvP framework.

5 Conclusions

5.1 Review and Conclusion

The work in this thesis ultimately proposed and evaluated the effectiveness of a swarm of autonomous vehicles using different Voronoi based search methods in an area coverage and tracking problem. The methods were analyzed using Monte Carlo simulation, which determined the probability of detection for each method. Moreover, the ability to generate target solutions from a simulated sensor with a normal distribution detection model was presented. Lastly, the search algorithms were deployed on autonomous surface vessels to demonstrate the deployability and feasibility of the simulated mission. Through these methods of evaluation, the potential for a swarm of USVs to serve as a detection and alert mechanism in a harbor defense setting is better understood.

Through Monte Carlo simulation, it was shown that the probability of the swarm to detect a passing intruder is increased when search methods involving swarm motion is included. When the measurement criteria is compared between each proposed search method, there is not a clear definitive best. However, the Vector Field search method demonstrated the highest possible mean detection probability in simulation. Thereby, without any prior knowledge of the threat vessel, the Vector Field search method was determined to be the method with the greatest potential. While the determination of the best search method for a setting with limited prior knowledge lacks complete

conviction, this experiment demonstrated that every search method proposed is more effective at detecting an intruder than a stationary swarm. Therefore, a swarm of mobile surface vessels enacting Voronoi based search methods will increase the probability of detecting a threat passing through a cover region.

The use of traditional search methods for a moving target often rely on a time dependent target location probability. Optimally, a swarm employed in an area coverage and tracking scenario would provide this level of information. This thesis showed that a swarm with a simulated active sonar application was able to produce an actionable target solution on average. This was accomplished using crude error prone detection reports. Therefore, a swarm equipped with inexpensive simple detection mechanisms can provide useful target information.

The Monte Carlo simulations used for this thesis provided useful data to predict the effectiveness for each of the tested swarm search methods. However, simulations do not always capture the complexity of algorithm execution on real world platforms. This thesis demonstrated that the Voronoi search methods proposed and tested in simulation, in fact, can be employed effectively on autonomous surface vessels. The hardware test was not used to determined the best search method. This is because the asymmetry of the 7 vehicle distribution and the limited number of target vessel crossings prevented a definitive conclusion. However, this was the purpose of the Monte Carlo Simulations. All in all, the MOOS-IvP environment provided a mature ecosystem that enabled the smooth transition from algorithm development, to simulation validation, and ultimately to hardware execution.

As the rapid development in robotic systems and artificial intelligence flow into adversarial marine vehicles, the need for equally capable defense systems is required. This thesis examined and quantified the effectiveness of a swarm of autonomous surface vehicles in detecting and tracking an adversarial AUV. Further, the work in this thesis

demonstrated a novel approach to increase the probability of detecting a threat vehicle, to generate an actionable target solution, and to employ these methods on a robotic platform. The use of Voronoi based methods proved to be a deployable and effective method for area coverage and tracking.

5.2 Future Work

While the work in this thesis proposes and evaluates novel autonomous marine vehicle control methods, further testing and data collection would provide a more complete assessment of Voronoi based search methods. The work in this thesis focuses on the speed ratio between adversary and swarm. However, another useful metric would include evaluating the ratio of swarm coverage to region area. Varying the number of vehicles inside the cover region would provide a measurement of this ratio. Then, the combination of the speed ratio and area coverage ratio data would provide a method to estimate swarm detection probability given a region, detection range, number of vehicles, and speed ratio. In fact, this additional data would provide a mechanism to estimate any one of the above variables given the others. This would provide a mission designer a tactical decision aid with data backed estimates.

The hardware employment conducted for this thesis accurately demonstrated the deployability of the proposed search algorithms; however, the ability of the swarm to detect a vessel was estimated using a simulated active sonar and simulated target vehicle. For future work, it would be beneficial to execute the search algorithms using a single beam echo sounder and a UUV platform to validate the sensor model. The tools developed for this thesis are ready to be evaluated with this real world upgrade. Further, the conclusions drawn from the simulated active sonar could be appropriately challenged and compared.

The Monte Carlo simulations were conducted with a 9 vehicle swarm and tested using a target crossing in one direction. The symmetry of the 9 vehicle swarm permitted this assumption. However, in order to properly report the detection probability of a swarm with an adversary crossing from any direction, the simulation must be altered. The simulation scenario would be more robust and would not require a symmetry assumption, if the adversary could cross the region from any direction. Therefore, the simulation would benefit from this change in the future.

Acronyms

ASV	Autonomous Surface Vehicle
ASW	Anti-Submarine Warfare
AUV	Autonomous Underwater Vehicle
AxV	Unmanned Autonomous Vehicle
BHV	Behavior
CVT	Centroidal Voronoi Tesselations
IvP	Interval Programming
MOOS	Mission Oriented Operating Suite
MOOSDB	MOOS Data Base
SCAT	Simultaneous Coverage and Tracking
UUV	Unmanned Underwater Vehicle

Bibliography

1. Aurenhammer, F. Voronoi Diagrams—a Survey of a Fundamental Geometric Data Structure. *ACM Comput. Surv.* **23.** doi: 10.1145/116873.116880, 1991.
2. Ben Slimane, N. and Tagina, M. Proposition of a Distributed Voronoi Partitioning Approach Enhanced with a Dispersion Phase for a Multirobot System. *International Journal of Social Robotics* **13.** doi: 10.1007/s12369-020-00677-2, 2021.
3. Benjamin, M. R., Newman, P. M., Schmidt, H., and Leonard, J. J. An Overview of MOOS-IvP and a Brief Users Guide to the IvP Helm Autonomy Software, 2009.
4. Benkoski, S. J., Monticino, M. G., and Weisinger, J. R. A survey of the search theory literature. *Naval research logistics* **38,** 1991.
5. Berg, M., Kreveld, m., Overmars, M., and Schwarzkopf, O. *Computational Geometry* Springer Berlin Heidelberg, 1997.
6. Bureau of Naval Personnel, N. *Naval Sonar* United States Government Printing Office, 1953.
7. Butler, J. L. and Sherman, C. H. *Transducers and Arrays for Underwater Sound. 2nd ed.* Springer International Publishing, 2016.
8. Cortes, J., Martinez, S., Karatas, T., and Bullo, F. Coverage control for mobile sensing networks. *IEEE Transactions on Robotics and Automation* **20,** 2004.

9. D' Acunto, M. Optimized Dislocation of Mobile Sensor Networks on Large Marine Environments Using Voronoi Partitions. *Journal of Marine Science and Engineering* **8.** doi: 10.3390/jmse8020132, 2020.

10. Du, Q., Faber, V., and Gunzburger, M. Centroidal Voronoi Tessellations: Applications and Algorithms. *SIAM Review* **41.** doi: 10.1137/S0036144599352836, 1999.

11. Ferri, G., Munafo, A., Tesei, A., Braca, P., Meyer, F., Pelekanakis, K., Petroccia, R., Alves, J., Strode, C., and Lepage, K. Cooperative Robotic Networks for Underwater Surveillance: an Overview. *IET Radar, Sonar & Navigation* **11.** doi: 10.1049/iet-rsn.2017.0074, 2017.

12. Guruprasad, K. R. and Ghose, D. Automated Multi-Agent Search Using Centroidal Voronoi Configuration. *IEEE transactions on automation science and engineering* **8,** 2011.

13. Healey, A. J., Horner, D., Kragelund, S., Wring, B., and Monarrez, A. Collaborative Unmanned Systems for Maritime and Port Security Operations. *IFAC Proceedings Volumes* **40,** 2007.

14. Ketter, T. Anti-Submarine Warfare in the 21st Century, 2004.

15. Newbolt, H. J. *Submarine and anti-submarine* Longmans, Green and co., New York, London [etc, 1919.

16. Newman, P. MOOS - Mission Orientated Operating Suite, 2003.

17. Pimenta, L. C. A., Schwager, M., Lindsey, Q., Kumar, V., Rus, D., Mesquita, R. C., and Pereira, G. A. S. *Simultaneous Coverage and Tracking (SCAT) of Moving Targets with Robot Networks* in *Algorithmic Foundation of Robotics VIII* Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

18. Robin, C. and Lacroix, S. Multi-robot target detection and tracking: taxonomy and survey. *Autonomous robots* **40,** 2015.

19. Simetti, E., Casalino, G., Turetta, A., Storti, E., and Cresta, M. Towards the Use of a Team of USVs for Civilian Harbour Protection: USV Interception of Detected Menaces. *IFAC Proceedings Volumes* **43.** doi: https://doi.org/10.3182/20100906-3-IT-2019.00027, 2010.

20. Stone, L., Royset, J., and Washburn, A. *Optimal Search for Moving Targets* doi: 10.1007/978-3-319-26899-6. 2016.

21. Terracciano, D. S., Bazzarello, L., Caiti, A., Costanzi, R., and Manzari, V. Marine Robots for Underwater Surveillance. *Current Robotics Reports* **1,** 2020.

22. Teruel, E., Aragues, R., and López-Nicolás, G. A distributed robot swarm control for dynamic region coverage. *Robotics and autonomous systems* **119,** 2019.

23. Toti, W. J. The Hunt for Full-Spectrum ASW. *Proceedings* **140/6/1,336,** 2014.

24. Turchin, P., Hoyer, D., Korotayev, A., Kradin, N., Nefedov, S., Feinman, G., Levine, J., Reddish, J., Cioni, E., Thorpe, C., Bennett, J. S., Francois, P., and Whitehouse, H. Rise of the war machines: Charting the evolution of military technologies from the Neolithic to the Industrial Revolution. *PLOS ONE* **16.** doi: 10.1371/journal.pone.0258161, 2021.

25. Vencatasamy, K., Jaulin, L., and Zerr, B. In *Marine Robotics and Applications* (eds Jaulin, L., Caiti, A., Carreras, M., Creuze, V., Plumet, F., Zerr, B., and Billon-Coat, A.) Springer International Publishing, Cham, 2018.

26. Wagner, D., Mylander, W., and Sanders, T. *Naval Operations Analysis* Naval Institute Press, 1999.

27. XIONG, C., Chen, D., Lu, D., Zeng, Z., and Lian, L. Path planning of multiple autonomous marine vehicles for adaptive sampling using Voronoi-based ant colony optimization. *Robotics and Autonomous Systems* **115.** doi: 10.1016/j.robot.2019.02.002, 2019.

28. Yongjie, S. Underwater Acoustic Transducers, 2019.

29. Zekavat, R. and Buehrer, R. M. In *Handbook of Position Location: Theory, Practice, and Advances* 2019. doi: 10.1002/9781119434610.ch5.

Appendix

1 Batch Simulation Instructions

Below is the instructions for executing the batch simulations that provided the results for all simulations used to obtain the results of this thesis.

First all source code dependencies are required to be downloaded and built.

1. MOOS-IvP - found at https://oceanai.mit.edu/moos-ivp

2. moos-ivp-swarm extend tree

3. moos-ivp-voronoi-batch extend tree - found at https://github.com/ncevans/moos-ivp-voronoi-batch.git

A copy of the README associated with the moos-ivp-voronoi-batch tree is provide below

##
FILE: moos-ivp-voronoi-batch/README
DATE: 2022/06/22
#Modified from README provided from the MOOS-IVP extend tree checkout README
##

#===
Introduction
#===
This extend tree for MOOS-IvP contains all the source code and scripts to execute
the batch simulations ran for the thesis entitled A Practical Underwater Search with Voronoi Distributed Autonomous Swarms.

#===
Voronoi Batch Description and Usage details
#===
To execute the batch simulations the missions/VORONOI_BATCH_SIM/zlaunch.sh is used.

#===
    # zlaunch usage and parameters

#===
    COMMAND LINE ARGUMENTS

    SWARM RELATED ARGUMENTS
    --mode #Sets the swarm behavior mode to use the Vector Field or Region Search BHV
      --mode=VECTOR #Vector Field
      --mode=SEARCH #Region Search
    --search #Sets the mode of the Region Search BHV
      --search=ROTATE #Enacts the CVT Rotation method
      --search=STOCHASTIC_HEADING #Enacts the center oriented Stochastic Method
      --search=STOCHASTIC_FREE #Enacts the CVT complete Stochastic Method
      --search=STOCHASTIC #Enacts the center oriented Stochastic Method with changing random speeds
      --search=STOCHASTIC_FREE_SPEED #Enacts the complete stochastic method with random changing speeds
    --startvehicle #Sets the number of starting swarm vehicles to run the scenario with
      --startvehicle=5 #Runs the simulation with 5 swarm vehicles
    --endvehicle #Sets the amount of swarm vehicle to stop simulating.
      mission runs for startvehicle to endvehicle incremented by 1.
      --endvehicle=10 with start=5 the scenario will run for 5,6,7,8,9,10 swarm vehicles
        the data will be stored in data/veh_num0, data/veh_num1,... and so on
    --minspd #Sets the initial starting speed of the swarm

--spdsteps #sets the number of times swarm speed will be increased ZERO INDEXED
--spdsteps=1 # this produce 2 simulations minspd, minspd+(1*spdsteps)
--scalespeed #is the multiple each time step will increase speed by

ADVERSARY RELATED ARGUMENTS
MISSION LENGTH
The mission length is determined by the number of region crosses that the Adversary vehicle
completes. Therefore, the number of crosses is the variable uQueryDB uses to terminate the simulation

--cross #designates the number of adversary crosses
--cross=100 #The mission will stay running until the adversary crosses the op region 100 times

Adversary Speed: the simulations are ran for (speedstart) to (speedend) in (numsteps). Each speed
is held constant for the designated number of crosses.
Therefore if speedstart=.5, numsteps=6, and stepsize=.5 the sim will run 6 times for adversary speeds=
.5, 1.0, 1.5, 2.0, 2.5, 3.0. Each speed the adversary will cross the region for designate #of crosses

--speedstart # Initializes the Adversary speed for the starting simulation
--speedstart=.5 #The adversary speed will start at .5
--numsteps #sets the number of times the adversary changes speed THIS IS ZERO INDEXED
--numsteps=1 # this produce 2 simulations speedstart, speedstart+(1*stepsize)
--stepsize #The value of the speed increase for each step

MISSION BASED ARGUMENTS

--loc #designates whether the sailing pavilion mission or saxis mission is executed
--loc=saxis #Runs the saxis mission
--headless #takes no argument, will run simulations without pMarineViewer
Time warp is designate with a int value on command line with flag required
example. ./zlaunch.sh 10 #will run time warp 10

#==
Directory Structure

#==
The directory structure for the moos-ivp-extend is described below:

bin   - Directory for generated executable files
build   - Directory for build object files
build.sh   - Script for building moos-ivp-extend
CMakeLists.txt - CMake configuration file for the project
lib   - Directory for generated library files
missions   - Directory for the thesis mission
README   - Contains helpful information - (this file).
src   - Directory for source code

#==
Build Instructions
#==

To build on Linux and Apple platforms, execute the build script within this
directory:

  $ ./build.sh

To build without using the supplied script, execute the following commands
within this directory:

  $ mkdir -p build
  $ cd build
  $ cmake ../
  $ make
  $ cd ..

#==
Environment variables
#==
The moos-ivp-extend binaries files should be added to your path to allow them
to be launched from pAntler.

In order for generated IvP Behaviors to be recognized by the IvP Helm, you
should add the library directory to the "IVP_BEHAVIOR_DIRS" environment
variable.

##
END of README

2 Comparison Data

Table 1: Probability of Interaction at Discrete Distances for Adversary Speed=.5m/s.

Distance m	Stochastic %	Stochastic Heading %	Rotate %	Vector %	Stationary %
1.000	23.108	18.327	21.557	29.482	6.673
2.000	40.438	36.853	42.116	52.390	15.671
3.000	55.976	51.992	57.285	65.538	22.908
4.000	67.331	62.351	70.060	75.299	30.445
5.000	74.502	70.518	80.240	83.267	35.923
6.000	81.474	77.888	85.629	88.845	40.637
7.000	85.458	83.068	90.818	91.833	46.282
8.000	89.243	87.649	94.012	94.223	50.432
9.000	91.434	90.438	95.808	95.817	54.349
10.000	93.227	91.633	98.004	97.410	58.300
11.000	94.821	94.024	98.802	97.610	61.753
12.000	96.614	94.821	99.601	97.809	65.704
13.000	97.211	95.219	99.800	98.207	69.190
14.000	97.809	96.414	99.800	98.805	72.244
15.000	98.406	96.614	100.000	99.203	75.764
16.000	99.203	98.008	100.000	99.402	78.552
17.000	99.402	98.207	100.000	99.402	81.142
18.000	99.402	98.406	100.000	99.402	83.300
19.000	99.402	98.406	100.000	99.402	85.392
20.000	99.402	98.606	100.000	99.402	87.849

Table 2: Probability of Interaction at Discrete Distances for Adversary Speed=1.0m/s.

Distance m	Stochastic %	Stochastic Heading %	Rotate %	Vector %	Stationary %
1.000	14.143	11.952	13.546	16.932	6.673
2.000	26.295	27.092	25.100	30.279	15.671
3.000	34.263	37.052	37.052	41.235	22.908
4.000	45.219	44.622	44.821	50.797	30.445
5.000	52.390	52.390	54.382	57.968	35.923
6.000	59.761	58.765	61.753	64.940	40.637
7.000	65.538	64.741	66.932	71.116	46.282
8.000	70.518	69.721	70.120	74.900	50.432
9.000	73.904	74.701	73.108	79.482	54.349
10.000	78.685	77.689	77.092	83.068	58.300
11.000	83.267	79.880	80.876	86.255	61.753
12.000	86.056	84.661	84.064	88.446	65.704
13.000	87.649	87.052	87.052	90.837	69.190
14.000	90.040	88.845	88.048	92.829	72.244
15.000	91.633	90.438	89.044	94.223	75.764
16.000	93.227	92.231	90.637	95.817	78.552
17.000	95.020	93.625	92.231	96.614	81.142
18.000	96.414	95.219	93.825	97.012	83.300
19.000	96.813	95.817	94.223	98.008	85.392
20.000	97.012	97.211	95.219	98.606	87.849

Table 3: Probability of Interaction at Discrete Distances for Adversary Speed=1.5m/s.

Distance m	Stochastic %	Stochastic Heading %	Rotate %	Vector %	Stationary %
1.000	9.163	11.753	11.155	10.359	6.673
2.000	21.713	24.303	21.912	23.904	15.671
3.000	31.275	32.470	30.478	33.665	22.908
4.000	38.845	39.841	36.853	43.825	30.445
5.000	45.418	45.219	45.020	50.398	35.923
6.000	53.386	53.187	51.992	57.171	40.637
7.000	57.171	58.566	57.371	63.347	46.282
8.000	64.143	62.749	62.948	68.526	50.432
9.000	67.729	67.331	66.335	73.506	54.349
10.000	71.713	72.311	70.916	76.096	58.300
11.000	76.295	75.100	73.904	80.279	61.753
12.000	79.681	78.884	78.685	81.474	65.704
13.000	82.869	81.474	80.876	85.657	69.190
14.000	85.259	84.861	84.064	88.247	72.244
15.000	89.044	87.450	86.255	90.239	75.764
16.000	90.040	89.442	87.649	91.633	78.552
17.000	91.833	90.239	89.243	92.430	81.142
18.000	93.227	91.434	90.837	93.825	83.300
19.000	94.622	92.829	91.833	95.020	85.392
20.000	95.817	94.024	94.024	95.817	87.849

Table 4: Probability of Interaction at Discrete Distances for Adversary Speed=2.0m/s.

Distance m	Stochastic %	Stochastic Heading %	Rotate %	Vector %	Stationary %
1.000	8.549	8.748	9.960	9.742	6.673
2.000	17.694	19.682	22.510	21.074	15.671
3.000	27.833	27.833	30.279	32.803	22.908
4.000	35.586	35.388	36.454	41.352	30.445
5.000	41.352	44.135	44.024	48.907	35.923
6.000	46.123	50.099	50.996	54.871	40.637
7.000	51.491	56.461	56.375	61.431	46.282
8.000	57.853	61.431	62.351	66.402	50.432
9.000	63.419	66.600	68.127	71.769	54.349
10.000	68.191	69.583	72.112	75.149	58.300
11.000	73.559	73.559	76.494	78.529	61.753
12.000	76.938	78.330	79.482	81.113	65.704
13.000	80.915	81.511	82.072	85.885	69.190
14.000	84.095	84.294	84.462	87.078	72.244
15.000	86.083	87.276	86.653	88.270	75.764
16.000	89.861	89.861	88.845	89.662	78.552
17.000	91.451	91.252	91.833	91.650	81.142
18.000	93.042	93.638	93.227	93.241	83.300
19.000	94.036	95.229	94.223	93.837	85.392
20.000	94.632	95.229	95.618	95.229	87.849

Table 5: Probability of Interaction at Discrete Distances for Adversary Speed=2.5m/s.

Distance m	Stochastic %	Stochastic Heading %	Rotate %	Vector %	Stationary %
1.000	7.769	7.570	8.167	7.371	6.673
2.000	17.729	16.733	19.522	17.331	15.671
3.000	27.689	24.701	28.884	27.092	22.908
4.000	36.454	33.466	37.849	34.462	30.445
5.000	43.426	43.227	43.825	41.633	35.923
6.000	49.801	48.805	50.996	46.813	40.637
7.000	57.371	55.378	56.574	54.781	46.282
8.000	63.147	59.163	59.960	60.558	50.432
9.000	68.127	63.147	66.335	67.729	54.349
10.000	72.709	68.725	69.522	72.311	58.300
11.000	75.697	72.908	72.709	76.494	61.753
12.000	78.287	76.295	76.295	79.880	65.704
13.000	82.072	79.880	78.685	83.466	69.190
14.000	84.462	82.669	80.677	85.657	72.244
15.000	86.653	84.064	83.267	87.649	75.764
16.000	88.048	86.255	85.458	89.641	78.552
17.000	90.040	88.247	87.450	90.837	81.142
18.000	91.434	90.438	88.247	91.235	83.300
19.000	91.833	92.032	90.239	92.829	85.392
20.000	92.829	93.825	91.633	93.825	87.849

Table 6: Probability of Interaction at Discrete Distances for Adversary Speed=3.0m/s.

Distance m	Stochastic %	Stochastic Heading %	Rotate %	Vector %	Stationary %
1.000	6.773	8.367	6.175	5.976	6.673
2.000	18.924	16.733	19.522	17.729	15.671
3.000	26.295	24.502	30.279	27.888	22.908
4.000	33.865	33.865	38.446	38.247	30.445
5.000	40.637	40.438	45.020	47.012	35.923
6.000	48.207	47.410	50.996	54.382	40.637
7.000	55.976	52.590	55.578	60.159	46.282
8.000	60.558	57.371	61.355	65.936	50.432
9.000	65.139	63.147	65.737	71.315	54.349
10.000	70.717	67.131	68.725	75.697	58.300
11.000	74.303	71.912	71.912	79.283	61.753
12.000	77.490	76.693	74.701	82.470	65.704
13.000	79.283	78.685	78.685	83.466	69.190
14.000	82.669	81.275	82.271	86.653	72.244
15.000	84.861	84.064	84.661	88.446	75.764
16.000	86.653	86.653	86.653	89.641	78.552
17.000	89.641	89.243	89.243	91.633	81.142
18.000	91.434	90.837	90.239	93.028	83.300
19.000	91.833	92.629	91.434	93.825	85.392
20.000	92.829	93.625	92.629	94.622	87.849

2.1 Speed Ratio of 3.0

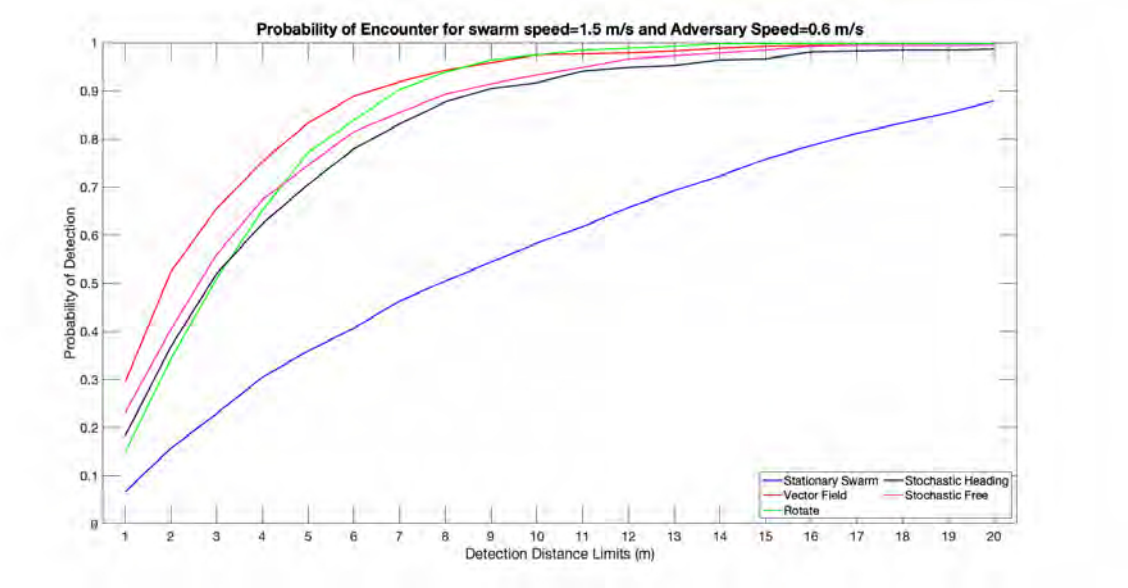


Figure 1: Probability of Interaction at Discrete Distances

Table 7: Probability Improvement from Stationary Swarm at Discrete Distances for Adversary Speed=.5m/s.

Detection Distance m	Stochastic %	Stochastic Heading %	Rotate %	Vector Field %
1.000	16.434	11.653	14.884	22.809
2.000	24.768	21.182	26.445	36.720
3.000	33.068	29.084	34.377	42.629
4.000	36.886	31.906	39.615	44.854
5.000	38.579	34.595	44.317	47.344
6.000	40.837	37.251	44.991	48.207
7.000	39.177	36.786	44.537	45.551
8.000	38.811	37.218	43.580	43.792
9.000	37.085	36.089	41.459	41.467
10.000	34.927	33.333	39.704	39.110
11.000	33.068	32.271	37.049	35.857
12.000	30.910	29.117	33.897	32.105
13.000	28.021	26.029	30.610	29.017
14.000	25.564	24.170	27.556	26.560
15.000	22.643	20.850	24.236	23.440
16.000	20.651	19.456	21.448	20.850
17.000	18.260	17.065	18.858	18.260
18.000	16.102	15.106	16.700	16.102
19.000	14.011	13.015	14.608	14.011
20.000	11.554	10.757	12.151	11.554

Table 8: Probability Improvement Average from Stationary Swarm for Adversary Speed=.5m/s.

Stochastic Free %	Stochastic Heading %	Rotate %	Vector Field %
28.068	25.847	30.551	32.012

2.2 Speed Ratio of 1.5

Data for Adversary Speed = 1.0

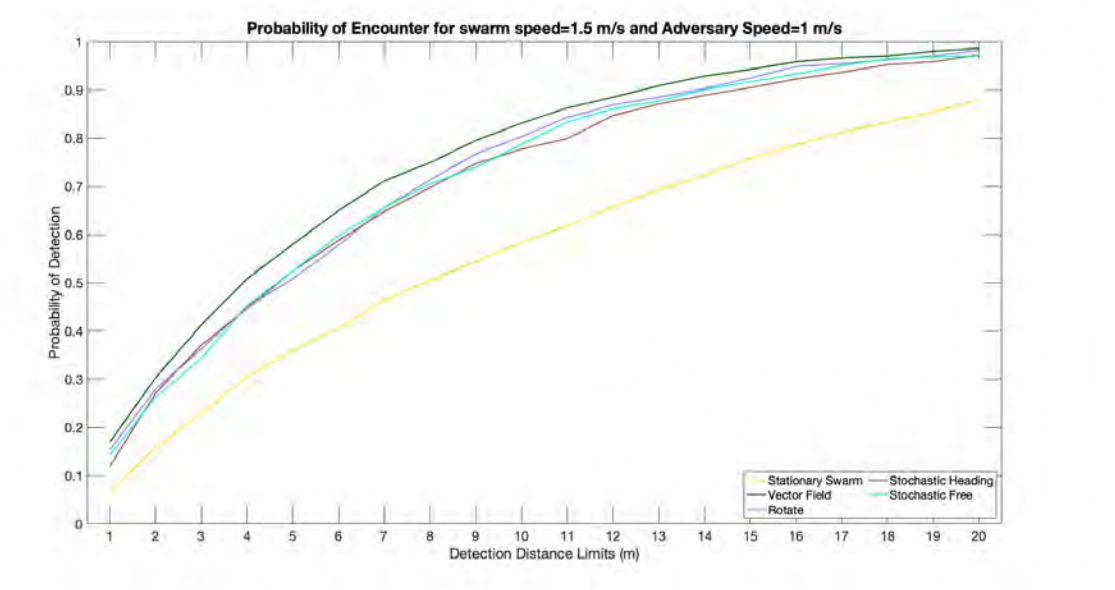


Figure 2: Probability of Interaction at Discrete Distances

Table 9: Probability Improvement from Stationary Swarm at Discrete Distances for Adversary Speed=1.0m/s.

Detection Distance m	Stochastic %	Stochastic Heading %	Rotate %	Vector Field %
1.000	7.470	5.279	6.873	10.259
2.000	10.624	11.421	9.429	14.608
3.000	11.355	14.143	14.143	18.327
4.000	14.774	14.177	14.376	20.352
5.000	16.467	16.467	18.459	22.045
6.000	19.124	18.127	21.116	24.303
7.000	19.256	18.459	20.651	24.834
8.000	20.086	19.290	19.688	24.469
9.000	19.555	20.352	18.758	25.133
10.000	20.385	19.389	18.792	24.768
11.000	21.514	18.127	19.124	24.502
12.000	20.352	18.958	18.360	22.742
13.000	18.459	17.862	17.862	21.647
14.000	17.795	16.600	15.803	20.584
15.000	15.870	14.675	13.280	18.459
16.000	14.675	13.679	12.085	17.264
17.000	13.878	12.483	11.089	15.471
18.000	13.114	11.919	10.525	13.712
19.000	11.421	10.425	8.831	12.616
20.000	9.163	9.363	7.371	10.757

Table 10: Probability Improvement Average from Stationary Swarm for Adversary Speed=1m/s.

Stochastic %	Stochastic Heading %	Rotate %	Vector Field %
15.767	15.060	14.831	19.343

2.3 Speed Ratio of 1

Data for Adversary Speed = 1.5

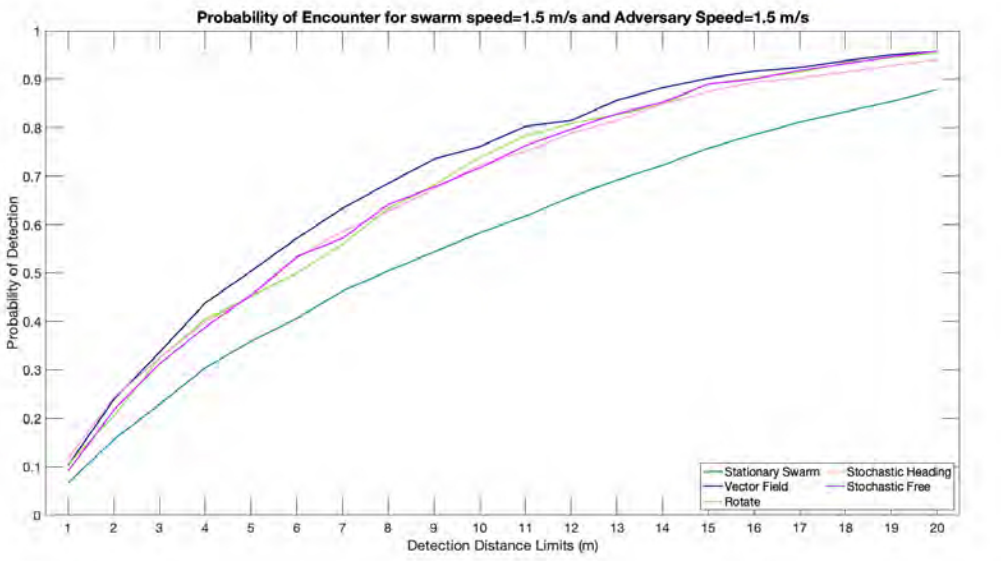


Figure 3: Probability of Interaction at Discrete Distances

Table 11: Probability Improvement from Stationary Swarm at Discrete Distances for Adversary Speed=1.5m/s.

Detection Distance m	Stochastic %	Stochastic Heading %	Rotate %	Vector Field %
1.000	2.490	5.080	4.482	3.685
2.000	6.042	8.632	6.242	8.234
3.000	8.367	9.562	7.570	10.757
4.000	8.400	9.396	6.408	13.380
5.000	9.495	9.296	9.097	14.475
6.000	12.749	12.550	11.355	16.534
7.000	10.890	12.284	11.089	17.065
8.000	13.712	12.317	12.517	18.094
9.000	13.380	12.981	11.985	19.157
10.000	13.413	14.011	12.616	17.795
11.000	14.542	13.347	12.151	18.526
12.000	13.977	13.181	12.981	15.770
13.000	13.679	12.284	11.687	16.467
14.000	13.015	12.616	11.819	16.003
15.000	13.280	11.687	10.491	14.475
16.000	11.487	10.890	9.097	13.081
17.000	10.691	9.097	8.101	11.288
18.000	9.927	8.134	7.537	10.525
19.000	9.230	7.437	6.441	9.628
20.000	7.968	6.175	6.175	7.968

Table 12: Probability Improvement Average from Stationary Swarm for Adversary Speed=1.5m/s.

Stochastic %	Stochastic Heading %	Rotate %	Vector Field %
10.837	10.548	9.492	13.645

2.4 Speed Ratio of .75

Data for Adversary Speed = 2.0

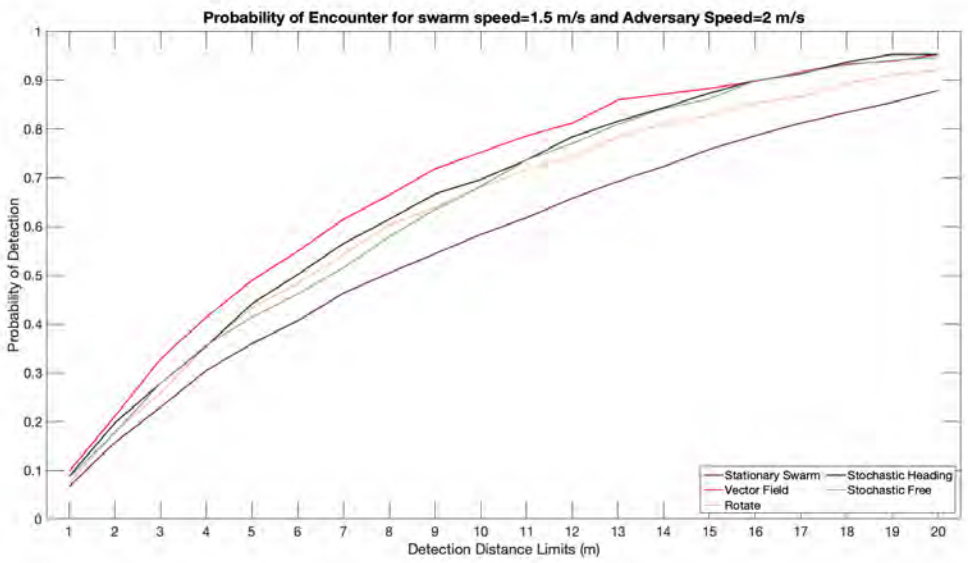


Figure 4: Probability of Interaction at Discrete Distances

Table 13: Probability Improvement from Stationary Swarm at Discrete Distances for Adversary Speed=2.0m/s.

Detection Distance m	Stochastic %	Stochastic Heading %	Rotate %	Vector Field %
1.000	1.875	2.074	3.287	3.068
2.000	2.023	4.011	6.839	5.403
3.000	4.925	4.925	7.371	9.895
4.000	5.142	4.943	6.009	10.907
5.000	5.429	8.212	8.101	12.984
6.000	5.486	9.462	10.359	14.233
7.000	5.210	10.180	10.093	15.150
8.000	7.421	11.000	11.919	15.970
9.000	9.070	12.251	13.778	17.420
10.000	9.891	11.282	13.811	16.849
11.000	11.806	11.806	14.741	16.776
12.000	11.235	12.626	13.778	15.409
13.000	11.725	12.321	12.882	16.695
14.000	11.851	12.050	12.218	14.833
15.000	10.320	11.513	10.890	12.507
16.000	11.308	11.308	10.292	11.110
17.000	10.309	10.110	10.691	10.508
18.000	9.742	10.338	9.927	9.940
19.000	8.644	9.837	8.831	8.445
20.000	6.784	7.380	7.769	7.380

Table 14: Probability Improvement Average from Stationary Swarm for Adversary Speed=2.0m/s.

Stochastic %	Stochastic Heading %	Rotate %	Vector Field %
8.010	9.381	10.179	12.274

2.5 Speed Ratio of .6

Data for Adversary Speed = 2.5

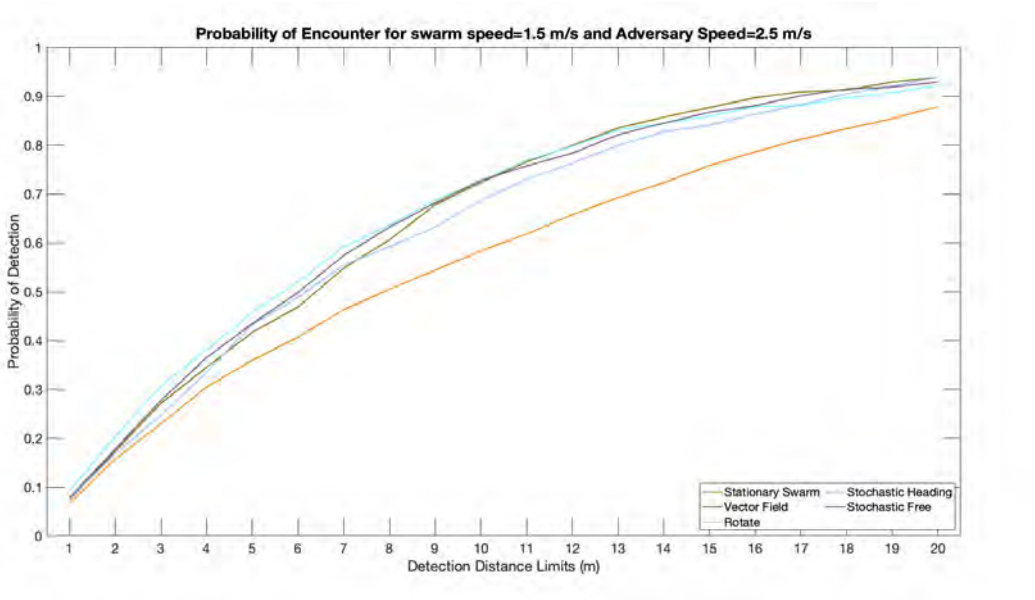


Figure 5: Probability of Interaction at Discrete Distances

Table 15: Probability Improvement from Stationary Swarm at Discrete Distances for Adversary Speed=2.5m/s.

Detection Distance m	Stochastic %	Stochastic Heading %	Rotate %	Vector Field %
1.000	1.096	0.896	1.494	0.697
2.000	2.058	1.062	3.851	1.660
3.000	4.781	1.793	5.976	4.183
4.000	6.009	3.021	7.404	4.017
5.000	7.503	7.304	7.902	5.710
6.000	9.163	8.167	10.359	6.175
7.000	11.089	9.097	10.292	8.499
8.000	12.716	8.732	9.529	10.126
9.000	13.778	8.798	11.985	13.380
10.000	14.409	10.425	11.222	14.011
11.000	13.944	11.155	10.956	14.741
12.000	12.583	10.591	10.591	14.177
13.000	12.882	10.691	9.495	14.276
14.000	12.218	10.425	8.433	13.413
15.000	10.890	8.300	7.503	11.886
16.000	9.495	7.703	6.906	11.089
17.000	8.898	7.105	6.308	9.695
18.000	8.134	7.138	4.947	7.935
19.000	6.441	6.640	4.847	7.437
20.000	4.980	5.976	3.785	5.976

Table 16: Probability Improvement Average from Stationary Swarm for Adversary Speed=2.0m/s.

Stochastic %	Stochastic Heading %	Rotate %	Vector Field %
9.153	7.251	7.689	8.954

2.6 Speed Ratio of .5

Data for Adversary Speed = 3.0

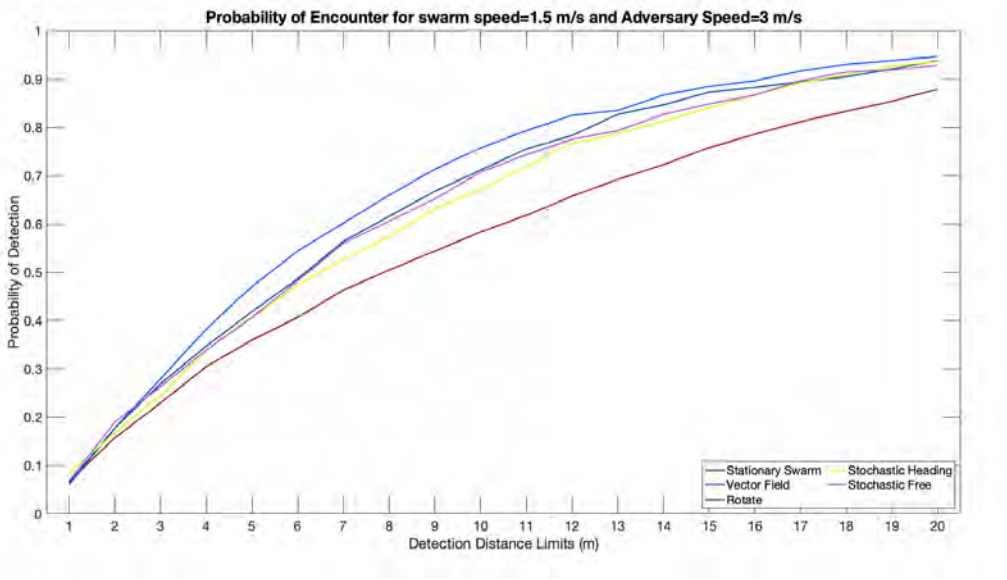


Figure 6: Probability of Interaction at Discrete Distances

Table 17: Probability Improvement from Stationary Swarm at Discrete Distances for Adversary Speed=3.0m/s.

Detection Distance m	Stochastic %	Stochastic Heading %	Rotate %	Vector Field %
1.000	0.100	1.693	−0.498	−0.697
2.000	3.254	1.062	3.851	2.058
3.000	3.386	1.594	7.371	4.980
4.000	3.420	3.420	8.001	7.802
5.000	4.714	4.515	9.097	11.089
6.000	7.570	6.773	10.359	13.745
7.000	9.695	6.308	9.296	13.878
8.000	10.126	6.939	10.923	15.505
9.000	10.790	8.798	11.388	16.965
10.000	12.417	8.831	10.425	17.397
11.000	12.550	10.159	10.159	17.530
12.000	11.786	10.989	8.997	16.766
13.000	10.093	9.495	9.495	14.276
14.000	10.425	9.031	10.027	14.409
15.000	9.097	8.300	8.898	12.683
16.000	8.101	8.101	8.101	11.089
17.000	8.499	8.101	8.101	10.491
18.000	8.134	7.537	6.939	9.728
19.000	6.441	7.238	6.042	8.433
20.000	4.980	5.777	4.781	6.773

Table 18: Probability Improvement Average from Stationary Swarm for Adversary Speed=3.0m/s.

Stochastic %	Stochastic Heading %	Rotate %	Vector Field %
7.779	6.733	8.088	11.245