

**Interception Algorithm for Autonomous Vehicles with Imperfect Information**

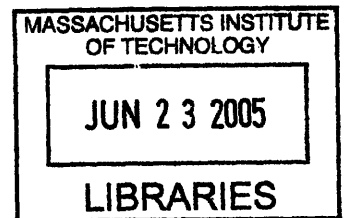
by

**Randal E. Hickman**

B.S., Systems Engineering  
United States Military Academy, 1997

SUBMITTED TO THE SLOAN SCHOOL OF MANAGEMENT IN PARTIAL  
FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

**MASTER OF SCIENCE**  
at the  
**MASSACHUSETTS INSTITUTE OF TECHNOLOGY**  
June 2005



© 2005 Randal E. Hickman. All rights reserved.

The author hereby grants to MIT permission to reproduce and to distribute publicly paper and electronic copies of the thesis document in whole or in part.

Signature of Author: \_\_\_\_\_

Operations Research Center  
May 6, 2005

Certified by: \_\_\_\_\_

Margaret F. Nervegna  
Charles Stark Draper Laboratory  
Technical Supervisor

Certified by: \_\_\_\_\_

John N. Tsitsiklis  
Professor of Electrical Engineering and Computer Science  
Co-Director, Operations Research Center  
Thesis Advisor

Accepted by: \_\_\_\_\_

James B. Orlin  
Edward Pennell Brooks Professor of Operations Research  
Co-Director, Operations Research Center

**ARCHIVES**

**(This page intentionally left blank)**

# **Interception Algorithm for Autonomous Vehicles with Imperfect Information**

by

**Randal E. Hickman**

Submitted to the Sloan School of Management  
on May 6, 2005, in Partial Fulfillment of the  
Requirements for the Degree of Master of Science in  
Operations Research

## **ABSTRACT**

Autonomous vehicles often operate in environments with imperfect information. This thesis addresses the case of a system of autonomous vehicles and sensors attempting to intercept a moving object of interest that arrives stochastically and moves stochastically after arrival. A sensor array is placed in the area of expected arrivals. As the object of interest moves across the sensor system, the system initially receives perfect information of the object's movements. After the object of interest leaves the sensor system, the algorithm uses statistical estimation techniques to develop confidence intervals about points of expected interception. The algorithm assigns the optimal, autonomous chase vehicle from a set of pre-positioned autonomous vehicles, develops movement commands for the assigned vehicle, and considers reassignment of chase vehicles as appropriate given the stochastic movements of the object of interest. Dynamic programming is employed to optimize system parameters, and the thesis considers a reformulation of the problem that uses dynamic programming as a structural model for the entire algorithm.

Technical Supervisor: Margaret F. Nervegna

Title: Member of the Technical Staff, Charles Stark Draper Laboratory

Thesis Supervisor: John N. Tsitsiklis

Title: Professor of Electrical Engineering and Computer Science

Co-Director, Operations Research Center

## ACKNOWLEDGEMENTS

I am especially thankful for the many friends, advisors, and fellow researchers who have assisted me in the past two years. My wife Courtney, my parents, and the rest of my family have been especially supportive throughout the process. Members of the technical staff of Draper Laboratory have provided numerous important perspectives and insights during this research, and I appreciate the support and mentoring provided by Margaret Nervegna, Mike Ricard, and Steve Kolitz. I am also grateful for the suggestions and theoretical input from my MIT advisor, Professor John Tsitsiklis. Finally, I should also thank my peers at Draper Laboratory for their suggestions, support, and friendship. The input from Darryl Ahner, Matt Wroten, and Caleb Earnest was especially helpful.

This thesis was prepared at The Charles Stark Draper Laboratory, Inc., under Internal Company Sponsored Research Project 12601-001, Offensive UUV Operations. Publication of this thesis does not constitute approval by Draper or the sponsoring agency of the finding or conclusions contained herein. It is published for the exchange and stimulation of ideas.

---

Randal E. Hickman, CPT, U.S. Army, May 6, 2005

(This page intentionally left blank)

# Table of Contents

<b>ABSTRACT</b> .....	<b>3</b>
<b>ACKNOWLEDGEMENTS</b> .....	<b>4</b>
<b>TABLE OF CONTENTS</b> .....	<b>6</b>
<b>LIST OF FIGURES</b> .....	<b>9</b>
<b>LIST OF TABLES</b> .....	<b>10</b>
<b>CHAPTER 1 INTRODUCTION</b> .....	<b>11</b>
1.1 PROBLEM MOTIVATION.....	11
1.2 PROBLEM STATEMENT.....	12
1.3 THEORETICAL CONTRIBUTIONS.....	14
1.4 ORGANIZATION OF THE THESIS.....	14
1.5 SUMMARY OF INTERCEPTION ALGORITHM AND KEY RESULTS.....	15
<b>CHAPTER 2 SIMULATION ENVIRONMENT</b> .....	<b>18</b>
2.1 DESIGN OF THE SIMULATION.....	18
2.2 RANDOM NUMBERS AND SIMULATION.....	20
2.2.1 <i>Concept of Randomness</i> .....	20
2.2.2 <i>Use of Random Numbers in Simulation</i> .....	22
2.3 AREA OF OPERATIONS.....	25
2.4 SENSOR ARRAY.....	26
<b>CHAPTER 3 THREAT VEHICLES</b> .....	<b>30</b>
3.1 FOUNDATIONAL ASSUMPTIONS.....	30
3.2 MOVEMENT ASSUMPTIONS.....	31
3.2.1 <i>Threat Vessel's Intent</i> .....	31
3.2.2 <i>Arrival Distribution Across the Chokepoint</i> .....	31
3.2.3 <i>Initial Bearing of Threat Vessel</i> .....	32
3.2.4 <i>Initial Speed of Threat Vessel</i> .....	33
3.2.5 <i>Changes in Threat Vessel Bearing</i> .....	34
3.2.6 <i>Changes in Threat Vessel Speed</i> .....	36
3.3 RESULTS OF THREAT VESSEL MODELING.....	37
<b>CHAPTER 4 FRIENDLY VEHICLES</b> .....	<b>41</b>
4.1 SWARM INTELLIGENCE.....	41
4.2 UUV ASSUMPTIONS.....	42
4.3 DEFINITION OF THE ENGAGEMENT AREA.....	44
4.3.1 <i>Definition of the Desired Arc of Interception</i> .....	44
4.3.2 <i>Definition of the Initial Locations of the UUVs</i> .....	45
4.4 OUTLINE OF THE INTERCEPTION ALGORITHM.....	46
4.5 DETERMINING THREAT VESSEL SPEED AND BEARING.....	47
4.6 INITIAL POINT OF DESIRED INTERCEPTION.....	49
4.7 DEFINITION OF THE TIME HORIZON " $\tau$ ".....	50
4.8 ASSIGN A UUV.....	50
4.9 INFORMATION REQUIREMENTS AT EACH TIME STEP.....	53
4.10 CONTROLLING THE ASSIGNED UUV.....	53
4.10.1 <i>UUV Speed Assumptions</i> .....	54
4.10.2 <i>Update the UUVs' Positions</i> .....	55
<b>CHAPTER 5 CONFIDENCE INTERVALS &amp; THE REASSIGNMENT ALGORITHM</b> .....	<b>59</b>
5.1 THEORY OF CONFIDENCE INTERVALS.....	59

5.1.1	<i>Frequentist vs. Bayesian Perspectives of Confidence Intervals</i> .....	60
5.1.2	<i>Calculating a Confidence Interval</i> .....	61
5.1.3	<i>Optimality of the Confidence Intervals</i> .....	63
5.2	<b>THEORY APPLIED TO THE SIMULATION</b> .....	64
5.2.1	<i>Use of Confidence Intervals</i> .....	64
5.2.2	<i>Confidence Intervals Defining Scoring Regions</i> .....	65
5.2.3	<i>“Bent Ellipse” Shape</i> .....	68
5.2.4	<i>Effects of the “Bent Ellipse” Phenomenon</i> .....	70
5.2.5	<i>Application of Scoring Regions in the Simulation</i> .....	70
5.2.6	<i>Approximating the Scoring Regions in the Simulation</i> .....	72
5.3	<b>USING THE SCORING REGION DATA</b> .....	75
5.4	<b>REASSIGNMENT OF UUVs</b> .....	77
5.4.1	<i>When Should the Algorithm Consider Reassignment</i> .....	77
5.4.2	<i>3-Part Test for Reassignment</i> .....	81
5.4.3	<i>Interception Algorithm / Reassignment Example</i> .....	82
<b>CHAPTER 6 OPTIMIZING THE RADIUS OF THE INTERCEPTION ARC</b> .....		<b>85</b>
6.1	<b>MOTIVATION TO OPTIMIZE THE RADIUS OF THE INTERCEPTION ARC</b> .....	85
6.2	<b>PROBLEM FORMULATION</b> .....	88
6.2.1	<i>Assumptions</i> .....	88
6.2.2	<i>Optimization Problem</i> .....	89
6.3	<b>SOLUTION TO OPTIMIZATION PROBLEM</b> .....	91
6.3.1	<i>Examination of the UUV Coverage Constraint</i> .....	91
6.3.2	<i>Re-formulation of the UUV Coverage Constraint</i> .....	92
6.3.3	<i>Solution to the Optimization Problem</i> .....	94
<b>CHAPTER 7 OPTIMIZING INITIAL UUV LOCATIONS THROUGH DYNAMIC PROGRAMMING</b> .....		<b>96</b>
7.1	<b>MOTIVATION TO OPTIMIZE THE UUV LOCATIONS</b> .....	96
7.2	<b>DYNAMIC PROGRAMMING THEORY</b> .....	97
7.3	<b>APPLIED DYNAMIC PROGRAMMING ALGORITHM</b> .....	99
7.3.1	<i>Dynamic Programming Modeling Methodology</i> .....	100
7.3.1.1	<i>Speed of Threat Vehicles</i> .....	100
7.3.1.2	<i>Initial UUV Locations within the Sensor Array</i> .....	101
7.3.1.3	<i>Optimize Half of the Arc and Reflect Data to the Other Side</i> .....	101
7.3.1.4	<i>Define the list of Candidate UUV Coordinates</i> .....	102
7.3.1.5	<i>Defining the Discretization of Candidate Points</i> .....	105
7.3.1.6	<i>Weighting of the Interception Arc</i> .....	105
7.3.1.7	<i>No Added Value for Overlapped Coverage</i> .....	105
7.3.2	<i>Design of the Dynamic Program</i> .....	106
7.3.2.1	<i>State Variables</i> .....	106
7.3.2.2	<i>Control Variable</i> .....	110
7.3.2.3	<i>System Equation</i> .....	110
7.3.2.4	<i>Cost Function</i> .....	110
7.3.2.5	<i>Implementing the Dynamic Programming Algorithm</i> .....	112
7.3.3	<i>Results of the Dynamic Program</i> .....	114
7.3.3.1	<i>Optimal Regions for Initial UUV Placements</i> .....	114
7.3.3.2	<i>The Curse of Dimensionality</i> .....	115
7.3.3.3	<i>The Discretization Paradox</i> .....	116
7.3.3.4	<i>Optimal Initial Coordinates for Scenarios with Four UUVs</i> .....	116
7.3.4	<i>Optimizing Coverage Positions</i> .....	118
7.3.4.1	<i>Three UUVs Remaining</i> .....	118
7.3.4.2	<i>Two UUVs Remaining</i> .....	122
7.3.4.3	<i>One UUV Remaining</i> .....	123
<b>CHAPTER 8 RESULTS, ANALYSIS, AND ALTERNATE STRATEGIES</b> .....		<b>125</b>
8.1	<b>NUMERICAL RESULTS</b> .....	125
8.2	<b>ANALYSIS OF RESULTS</b> .....	126

8.3	ALTERNATE MODELING METHODOLOGY – IMPERFECT STATE INFORMATION AND CERTAINTY	
	EQUIVALENT CONTROL .....	128
8.3.1	<i>Imperfect State Information</i> .....	129
8.3.2	<i>Certainty Equivalent Control</i> .....	134
<b>CHAPTER 9</b>	<b>CONCLUSION</b> .....	<b>137</b>
9.1	RESEARCH SUMMARY .....	137
9.2	AREAS FOR FUTURE RESEARCH.....	138
9.2.1	<i>Addition of Known Terrain Features</i> .....	138
9.2.2	<i>Unplanned Obstacles</i> .....	139
9.2.3	<i>Intelligent Adversary</i> .....	140
<b>APPENDIX A</b> .....		<b>141</b>
<b>REFERENCES</b> .....		<b>144</b>



## List of Figures

FIGURE 1.1 – TYPICAL USE OF A UUV IN AN ISR SCENARIO.....	13
FIGURE 2.1 – AREA OF OPERATIONS FOR THE SIMULATION.....	26
FIGURE 2.2 NUMBERED SENSORS IN BASIC SENSOR ARRAY LAYOUT .....	27
FIGURE 2.3 SENSOR ARRAY WITH RANDOMNESS IN SENSOR LOCATION .....	29
FIGURE 3.1 – 50 TIME STEPS OF ONE THREAT VESSEL WITH RANDOM BEARING CHANGES .....	37
FIGURE 3.2 – 50 TIME STEPS OF ONE THREAT VESSEL WITH NO BEARING CHANGES.....	38
FIGURE 3.3 – 50 TIME STEPS OF ONE THREAT VESSEL – UNLIKELY RANDOM OUTCOME.....	38
FIGURE 3.4 – 20 RANDOM ITERATIONS OF THE THREAT VESSEL ALGORITHM, 50 TIME STEPS PER ITERATION .....	39
FIGURE 3.5 – 50 RANDOM ITERATIONS OF THE THREAT VESSEL ALGORITHM, 50 TIMES STEPS PER ITERATION .....	39
FIGURE 4.1 – ENGAGEMENT AREA DEFINED AND INITIAL LOCATIONS OF THE 4 UUVS .....	45
FIGURE 4.2 – CALCULATING THE BEARING ( $\Theta$ ) OF THE THREAT VESSEL .....	48
FIGURE 4.3 – ASSUMED COVERAGE POSITIONS WITH 3 UUVS REMAINING.....	52
FIGURE 4.4 – ASSUMED COVERAGE POSITIONS WITH 2 UUVS REMAINING.....	52
FIGURE 4.5 – ASSUMED COVERAGE POSITION WITH ONLY 1 UUV REMAINING.....	52
FIGURE 4.6 – MOVEMENT OF THE ASSIGNED UUV DURING 1 TIME STEP (1 MINUTE) .....	56
FIGURE 5.1 – CONCEPTUAL FIGURE FOR SCORING REGION CONSTRUCTION .....	66
FIGURE 5.2 – POSSIBLE SHAPES FOR THE SCORING REGIONS ABOUT $(x_\tau, y_\tau)$ .....	67
FIGURE 5.3 – TWO POSSIBILITIES OF ROTATED SCORING REGIONS .....	67
FIGURE 5.4 – DEPICTION OF THE BEARING CONFIDENCE INTERVAL AS AN ARC.....	69
FIGURE 5.5 – APPROXIMATION OF THE BENT ELLIPSE SHAPE FOR SCORING REGIONS.....	70
FIGURE 5.6 – TWO VISUAL EXAMPLES OF THE SCORING REGIONS AROUND $(x_\tau, y_\tau)$ .....	73
FIGURE 5.7 – WIDER SCORING REGIONS TO ACCENTUATE THE “BENT ELLIPSE” EFFECT.....	74
FIGURE 5.8 – PROGRESSIVELY SMALLER SCORING REGIONS AS THE NUMBER OF TIME STEPS INCREASES ...	75
FIGURE 5.9 – VERY SMALL SCORING REGIONS WHEN $(x_\tau, y_\tau)$ IS INSIDE A SENSOR RADIUS.....	79
FIGURE 5.10 – EXAMPLE OF SUCCESSFUL REASSIGNMENT .....	84
FIGURE 6.1 – FIRST EXAMPLE OF THREAT VESSEL CHANGING BEARING AFTER LEAVING SENSOR ARRAY..	86
FIGURE 6.2 – SECOND EXAMPLE OF THREAT VESSEL CHANGING BEARING AFTER LEAVING SENSOR ARRAY .....	86
FIGURE 7.1 – INITIAL UUV LOCATIONS AS ASSUMED IN THE BASE CASE SCENARIO.....	96
FIGURE 7.2 – CURVE OF CANDIDATE POINTS FOR ONE-HALF OF THE SENSOR ARRAY .....	102
FIGURE 7.3 – REFINEMENT PROCESS FOR DYNAMIC PROGRAMMING OUTPUT .....	104
FIGURE 7.4 – COVERAGE REGION ON THE INTERCEPTION ARC FROM ONE UUV LOCATION .....	107
FIGURE 7.5 – UUV COVERAGE RADIUS EXCEEDS THE BOUNDS OF THE PROBLEM.....	108
FIGURE 7.6 – TRANSLATING INTERCEPTION POINTS TO RADIAN MEASURES .....	109
FIGURE 7.7 – CALCULATED OPTIMAL AREAS & INITIALLY ASSUMED STARTING POINTS .....	115
FIGURE 7.8 – OPTIMAL UUV LOCATIONS WITH FOUR UUVS.....	118
FIGURE 7.9 – OPTIMAL UUV LOCATIONS WITH THREE UUVS .....	121
FIGURE 7.10 – OPTIMAL UUV LOCATIONS WITH TWO UUVS.....	123
FIGURE 7.11 – OPTIMAL UUV LOCATION WITH ONE UUV .....	124
FIGURE 8.1 – CERTAINTY EQUIVALENT CONTROLLER IMPLEMENTED IN FEEDBACK FORM (FROM [5]).....	135

## List of Tables

TABLE 5.1 – PROXIMITY SCORES FOR UUVs.....	76
TABLE 5.2 – WHICH CONFIDENCE INTERVAL IS CLOSEST TO 500 M FROM $(x_r, y_r)$ WITHOUT EXCEEDING 500 M.....	78
TABLE 5.3 – MODIFIED DATA, AFTER ELIMINATING CASES WITH UNREALISTICALLY SMALL CONFIDENCE INTERVALS.....	80
TABLE 6.1 – CRITICAL RATIO VALUES FOR EQUATION 6.5 GIVEN THREAT VEHICLE SPEED.....	91
TABLE 6.2 – SIMULATION RESULTS WITH DIFFERENT INTERCEPTION ARC RADIUS LENGTHS.....	95
TABLE 7.1 – OPTIMAL INITIAL UUV LOCATIONS VS. ASSUMED VALUES.....	117
TABLE 7.2 – OPTIMAL PLACEMENT OF THREE UUVs & ORIGINALLY ASSUMED VALUES .....	121
TABLE 7.3 – OPTIMAL PLACEMENT OF TWO UUVs & ORIGINALLY ASSUMED VALUES .....	122
TABLE 7.4 – OPTIMAL UUV PLACEMENT AND ASSUMED VALUES FOR ONE UUV .....	123
TABLE 8.1 – SIMULATION RESULTS .....	126

# Chapter 1 Introduction

The objective of this thesis is to develop an interception algorithm for a system of sensors and Unmanned Undersea Vehicles (UUVs) tasked with finding and intercepting a threat vessel in an aquatic environment. The base scenario uses simple shortest path algorithms for assignment and control calculations. Improvements on the base case use both applied confidence intervals to motivate a reassignment algorithm and principles of optimization and applied dynamic programming to optimize system parameters. A simulation of the interception algorithm is developed and is used to validate the performance of the algorithm.

## **1.1 Problem Motivation**

The rapid increase in computer technology over the last half-century has allowed for the creation and expansion of the fields of robotics, unmanned vehicles, and autonomous systems. What started as large machines accomplishing relatively simple tasks has grown into the applied use of miniature robotics with diverse uses ranging from manufacturing, to medicine, to military systems. Dramatic improvements in the size and speed of microprocessors have encouraged a corresponding growth in the fields of artificial intelligence and vehicular autonomy. Modern research in the areas of reinforcement learning, machine learning, and neuro-dynamic programming offer the potential of truly autonomous vehicles accomplishing complex and dangerous assignments in changing or unknown environments [16].

Autonomous systems of this type have a vast array of potential military uses. An autonomous system could potentially respond to rapidly changing stimuli more quickly and effectively than a human operator, while minimizing the exposure of friendly forces to the inherent dangers of the modern battlefield. An autonomous system could operate in areas contaminated with chemical or biological weapons, or in areas with extreme environmental conditions that would make human presence very uncomfortable or

completely impossible. An autonomous vehicle can also operate for long periods of time without growing tired or hungry.

The U.S. Army is already experimenting with unmanned robotic systems for assistance with very dangerous missions such as cave clearing and urban warfare. Both the Army and the Air Force have a fleet of unmanned aerial vehicles (UAVs) used for long-range reconnaissance in hostile areas. Similarly, the U.S. Navy is interested in developing its own fleet of unmanned undersea vehicles (UUVs) for a variety of difficult and dangerous missions in an aquatic environment.

The U.S. Navy publicly released its “Unmanned Undersea Vehicle Master Plan” in the Spring of 2005 [30]. This document outlines a diverse group of potential missions for maritime autonomous systems, ranging from reconnaissance and inspection to payload delivery. It includes both benign uses such as oceanography, communications, and navigation assistance, as well as potentially dangerous tasks such as mine countermeasures and anti-submarine warfare.

Each of these “high-level” missions has numerous sub-tasks that must be accomplished in order for the high-level mission to succeed. One of the sub-tasks shared by many of the high-level missions is for the UUV to find or intercept another vehicle in a large body of water. This may be accomplished through interaction with other UUVs or with a sensor system. This finding / interception sub-task is particularly relevant for the high-level missions of intelligence, surveillance, and reconnaissance (ISR), and anti-submarine warfare (ASW). The requirements of this interception sub-task are the motivation for this research.

## ***1.2 Problem Statement***

The specific problem addressed by this thesis is to develop an interception algorithm for a system of autonomous UUVs and sensors. The sensors will initially provide the UUVs perfect information of threat vessel arrivals and movements in the vicinity of the area of interest. As the threat vehicles leave the sensor array, the model

loses its perfect information, and the UUVs must attempt to intercept the threat vessel through the applied use of expected values and confidence regions. The algorithm also employs optimization methodology and the applied use of dynamic programming to optimize system parameters.

This research offers a very successful algorithm for threat vehicle interception by UUVs, and a working simulation to test the existing algorithm as parameter values or assumptions change. The interception algorithm will directly support the stated subordinate tasks of “near-land and harbor monitoring,” “persistent and tactical intelligence collection,” “object detection and localization,” and “submarine track and trail” consistent with the high-level missions of ISR and ASW [30]. Figure 1.1 below is from the Navy’s UUV Master Plan, and it depicts the applied use of the interception algorithm in support of an ISR mission [30].



Figure 1.1 –Typical Use of a UUV in an ISR Scenario

### ***1.3 Theoretical Contributions***

This research provides two important theoretical contributions that advance the research field of autonomous vehicle control. The first contribution is to define the area of desired interception as an arc around a specific area of interest whose radius is adjustable by the user. This is in contrast to other research of this type that typically uses metrics such as intercepting the threat vessel as quickly as possible or closing the distance between the threat and the chaser as much as possible during each time step [24]. The approach in this research allows a human-in-the-loop interface that may take advantage of terrain or other known factors to influence the specific radius of the interception arc. In the absence of human intervention, the methodology of Chapter 6 will optimize the radius of the interception arc in a way that optimally balances the competing constraints of the UUV / threat vehicle speed differential and the sensing limitations of the sensor array.

The second theoretical contribution in this research is the applied use of confidence intervals around the expected point of desired interception. The confidence intervals are employed as a scoring metric to quantify whether or not a reassignment of the chaser UUV is desirable in response to the detected changes in threat vehicle speed or bearing. Since the number of UUVs is limited and the energy supply of each UUV is quickly depleted in a chase sequence, reassignments should be restricted to cases when such a reassignment is essential to the success of the model. Chapter 5 explores the details of the reassignment algorithm that uses confidence regions around the expected point of desired interception as a basis for the reassignment strategy.

### ***1.4 Organization of the Thesis***

The thesis is divided into nine chapters. The first chapter provides background information and problem motivation. The next three chapters describe the assumptions and underlying algorithmic structure of the simulation. Chapter 2 explains the underlying assumptions of the simulation structure, the area of operations, and the sensor network.

Chapter 2 also considers the concept of randomness and random numbers in simulation. Chapter 3 describes the underlying assumptions that govern the stochastic arrivals and behaviors of the threat vehicles. Chapter 4 explains the underlying assumptions of UUV behavior, and it describes the base-case algorithm for threat vehicle interception by the UUV / sensor system.

The next three chapters describe improvements to the base-case algorithm. Chapter 5 describes the applied use of confidence intervals as a tool to determine when UUV reassignment is desirable. Chapter 6 uses optimization principles to determine the optimal radius of the interception arc in the absence of human input. Chapter 7 applies dynamic programming to determine the optimal initial locations for the UUVs relative to the sensor array and the area of operations.

The last two chapters show the results of the algorithm and offer suggestions for future work. Chapter 8 includes a summary and analysis of results from the interception algorithm, and it also includes a description of a different modeling framework that solves the interception problem as a dynamic program with imperfect state information. Chapter 9 offers a summary of the research and provides recommendations for future work that could further develop or improve the interception algorithm.

## ***1.5 Summary of Interception Algorithm and Key Results***

The complete interception algorithm is developed starting with Section 4.4 of this thesis, after considering important assumptions and problem solving methodologies in Chapters 2-4. This section will provide a brief preview of the interception algorithm developed in this thesis and a summary of the key results.

1. The algorithm assumes the existence of a sensor system and 4 available UUVs in the area of operations. There is a desired interception region within the area of operations, approximated by an arc of desired interception. A threat vehicle enters the area of operations according to assumed probability distributions that independently describe its location, speed, and bearing.

2. The interception algorithm begins after the sensor system has observed the threat vessel for the second time. Using these two observations, the algorithm computes the threat vessel's speed and bearing.
3. Given the threat vessel's location and bearing, the algorithm determines the point where the threat vessel would cross the desired interception arc. Given the threat vessel's speed, the algorithm also calculates the number of discrete time steps (1 minute per time step) until the threat vessel reaches this point. This value becomes the parameter  $\tau$ , and all future interception attempts will be designed to occur at  $\tau$  time steps.
4. On the second visible time step, the algorithm also assigns a UUV as the chaser vehicle. This is accomplished based on which UUV is closest to the expected point of desired interception  $(x_\tau, y_\tau)$ , defined as the expected coordinates of the threat vessel at  $\tau$  time steps.
5. The assigned UUV begins movement towards  $(x_\tau, y_\tau)$  and the unassigned UUVs begin movement towards coverage positions that redistribute the remaining UUVs within the area of operations.
6. The threat vehicle's movements are stochastic, based on assumed distributions that allow for changes in threat vehicle speed and bearing. During each time step in which the threat vessel is seen by a sensor, the algorithm recalculates the speed and bearing of the threat vehicle and the coordinates  $(x_\tau, y_\tau)$ . The algorithm recalculates the control information for the assigned UUV in response to threat vehicle disposition changes. If the threat vehicle changes its disposition dramatically, an improvement to the basic algorithm allows for reassignment of the chaser UUV based on which UUV is closest to the newly calculated point  $(x_\tau, y_\tau)$ .
7. The algorithm terminates as a successful iteration if the chaser UUV intercepts the threat vehicle (defined as  $\pm 500$  meters between the chaser UUV and the threat vehicle.) The algorithm terminates as an unsuccessful iteration if more than  $\tau$



time steps have occurred and the model determines that there is no reasonable probability of successful interception.

The basic interception algorithm without the reassignment algorithm has a success rate of about 74%. The reassignment algorithm in this thesis uses the confidence intervals around  $(x_r, y_r)$  as a scoring metric to determine whether or not reassignment of the chaser UUV is appropriate. The inclusion of the reassignment algorithm increases the rate of successful interception to about 86%. The optimization process in Chapter 6 determines the optimal interception arc radius. Since the optimal value is different from the previously assumed value, use of the optimized interception arc radius increased the rate of successful interception to approximately 89%. Chapter 7 uses dynamic programming to optimize the initial UUV locations and coverage positions. Since the optimal locations differ from the previously assumed locations, the use of the optimized UUV locations further increases the algorithm's success rate. After implementing all the improvements discussed above, the final version of the interception algorithm successfully intercepts the threat arrivals more than 92% of the time.

## Chapter 2 Simulation Environment

The focus of this chapter is to describe the underlying simulation environment by addressing the basic framework of the simulation, the theory and use of random numbers in simulation, and the assumptions used to create the simulation environment. Later chapters will focus on the threat vessels, the UUVs, the interception algorithms, and the optimization problems involved.

### 2.1 Design of the Simulation

The simulation framework is designed as a tool to evaluate the UUV's ability to find and intercept threat vehicles. This is a fundamental sub-task for the ISR and ASW high-level missions of the UUV Master Plan. While the simulation remains intentionally generic, this research specifically supports numerous subordinate tasks for ISR and ASW as described in Chapter 1. The underlying design of the simulation must accurately model the environments in which the UUVs would likely perform these missions. This includes realistic modeling representations of the threat systems, the friendly systems, the doctrinal process of decision support, and the relevant aquatic environments.

The interception algorithm in this simulation is run in discrete time, and a simulation time step is defined as one minute. The one-minute time step is the smallest measurable increment of time in this simulation, and all movements of the threat vehicles and the UUVs are based on the one-minute time step. In contrast, the term *iteration* will be used to describe a complete simulation run that may result in successful interception or a failure to intercept the threat vessel.

The threat systems in this simulation may include a variety of aquatic vessels ranging from large surface ships, to small patrol boats, to submarines. The terms "threat vehicle" or "threat vessel" are used generically throughout the thesis to describe this threat. The important assumption is that the threat is a ship or boat of some kind, and that the UUV system has a mission of monitoring, following, tracking, or intercepting the threat vessel. Since any of these specific missions requires the UUV to find the threat

vessel, the goal of the algorithm is to locate or intercept the threat vessel in a large body of water. Since the threat vessel could be a variety of systems with varying capabilities, the simulation requires the use of applied stochastics to accurately model threat capabilities. Chapter 3 will focus on the specific assumptions and mathematical formulations that describe the arrival processes and stochastic movements of the threat vehicles.

The friendly systems in this simulation are a collection of UUVs and sensors, working in conjunction with each other to monitor an area of interest for threat vehicle activity. The simulation structure must accurately describe the energy, communications, and longevity limitations of the UUV / sensor system. The simulation assumes typical values for these parameters, since specific UUVs or sensors have varying capabilities. Chapter 4 will focus on the detailed assumptions and mathematical models behind friendly system capabilities.

The underlying simulation must also accurately reflect the doctrinal usage and decision support tools for UUV deployment. Although specific doctrine for UUV deployment is still in developmental form, the assumptions for this simulation are consistent with the U.S. Navy's UUV Master Plan, publicly released in March 2005 [30]. Doctrinal consistency is relevant throughout the thesis, and all modeling assumptions are designed to be consistent with each other and in support of the emerging UUV doctrine.

Finally, the simulation structure must accurately model the aquatic environment in which the UUVs would doctrinally operate. To support the specific missions mentioned above, the UUV is likely to conduct many of its missions in relatively shallow water in the vicinity of land, the area often referred to as the littorals [30]. The specific design of the sensor system is tailored to this environment, and includes randomness in sensor placement due to unpredictable tides, currents, and navigational error in sensor placement systems. The second half of Chapter 2 will focus on the specific assumptions and models that describe the aquatic environment of the simulation.

Many of the simulation components described above are fundamentally stochastic in nature. For the simulation to model the real-world system within these design objectives, it models the stochastic processes through the use of random variables with

assumed distributions. This process requires a reliable source of random numbers. The following section discusses randomness and the application of random numbers in greater detail.

## ***2.2 Random Numbers and Simulation***

In order to model system characteristics realistically, the simulation employs a variety of assumed distributions to describe the inherently stochastic arrivals and movements of threat vehicles. A fundamental question in stochastic simulation is how to interject realistic randomness into the simulation. This simulation models stochastic phenomena through relevant probability expressions. However, the probability expressions require random numbers as input. This section will focus on the concept of random numbers and then address the use of random numbers in a simulation.

### **2.2.1 Concept of Randomness**

A natural definition of randomness is the degree to which an observation is unpredictable. One author appealed to intuition by characterizing the process of random number generation as being similar to many successive spins of an unbiased “wheel of fortune [21].” Mathematically speaking, a string of numbers is said to be random if there is no shorter way to describe the string than to state the entire string itself [10]. This means that there is no mathematical expression to characterize the string other than to state the full string, and that no algorithm or formula can describe the string more simply. More specifically, knowledge of any one number or any string of numbers should have no bearing on the ability to predict future numbers [28]. There are statistical tests (such as the well-known chi-squared tests) that can measure randomness, and the extent to which a series of numbers can be shown to be statistically random is its measurement of entropy [17]. Higher entropy corresponds to higher randomness.

In order to interject sufficient randomness into the simulation, the objective is to achieve a high level of entropy in a set of random numbers. A variety of methods are

proposed by cryptographic experts. One expert suggested using human inputs such as the time intervals between typed characters on a keyboard or mouse movements. The same expert also suggested using characteristics of the computer hardware such as the system clock or the air turbulence in the disk drive [10]. Several web sites offer lists of “truly random numbers” that come from a variety of sources. One site claims to achieve statistically entropic random numbers through the atmospheric noise on unused radio frequencies [17]. Another site uses the inherent uncertainty of quantum mechanics to produce highly entropic random numbers through the timing of successive pairs of radioactive decay [28].

Since computer simulations inherently require an algorithm to generate the “random” numbers, the objective is to find a highly entropic stream of numbers that can serve as the seed values for the “random” algorithm [10]. This simulation uses the random number generator in MATLAB. By design, this random number generator is not truly random because it relies on a starting seed and uses a deterministic algorithm. In MATLAB, the starting seed is either set by the user directly or allowed to default to a known (fixed) seed that resets when starting the program. Consequently, the numbers produced by the MATLAB algorithm are predictable (as a result of the deterministic algorithm used), and not truly random [13]. The proper term for such data is *pseudo-random* [13]. The specific algorithm used by MATLAB is [29]:

$$seed = (7^7 \text{ seed}) \bmod (2^{31} - 1) \quad (2.1)$$

Based on the characteristics of the *modulo* (mod) function, algorithms of this form can have at most the same number of random output quantities as the value of the *modulo* term [21]. In this case, the MATLAB algorithm will produce

$$(2^{31} - 1) = 2,147,483,647 \quad (2.2)$$

pseudo-random values before repeating. This is sufficiently high that there should never be a repeated pseudo-random number in the course of the simulation.

The pseudo-random results should behave similarly to independent, identically distributed random variables from a  $U(0, 1)$  distribution, given the starting seed [18]. Although the predictability of this model would make the algorithm unacceptable for

some purposes (such as cryptographic work), the linear design makes it especially fast and therefore useful for simulation purposes [18].

Despite its deterministic structure, the MATLAB algorithm actually produces surprisingly good “random” results. Some interesting research performed by Robert Boehringer at Virginia Tech demonstrated the randomness of the MATLAB random number generator. Boehringer’s experiment used 1 million random numbers generated by MATLAB, and sorted them into 100 uniformly spaced bins. Boehringer found that MATLAB’s random number generator produced “random” output that varied from the calculated, expected values by  $\pm 3\%$  per bin [9]. Although Boehringer expected a spread somewhat less than 3%, this is consistent with good “randomness” because the data followed the basic distribution  $U(0, 1)$  while demonstrating a reasonable level of entropy through the variance between bins. Since MATLAB’s random number generator produces data that is acceptably random for simulation purposes, this research uses the pseudo-random output from MATLAB’s random number generator in order to take advantage of the high-speed, linear design of the algorithm.

A deterministic algorithm like MATLAB’s will produce repeatable results given a known starting seed. Repeatability may be desirable in order to compare simulation runs from the same “random” set, but with different values for some of the parameters. This methodology is important to compare the simulation results using different parameter values, as shown in Chapter 8 of this thesis.

## 2.2.2 Use of Random Numbers in Simulation

Given a source of acceptably “random” numbers, the next step is to use the random numbers to generate random samples from known distributions. Many methods exist to solve this problem. Larson and Odoni list four methods in their book, *Urban Operations Research* [21]:

- Invert the Cumulative Distribution Function (CDF).

- Exploit relationships of probability distributions whose subcomponents are easily inverted.
- The “Rejection Method,” (based on the geometric interpretations of probability density functions).
- The “Method of Approximations” (relevant to this research in reference to the Gaussian distribution).

This section will focus on the first and fourth methods.

If the CDF is easily invertible, then inverting the CDF is the most fundamental approach for generating random observations of known distributions. This method uses randomly generated,  $U(0, 1)$  random numbers. An explanation of the process is below.

The Probability Integral Transformation Theorem states the following [12]:

*TH: Let  $X$  have continuous cdf  $F_X(x)$  and define the random variable  $Y$ , as  $Y = F_X(X)$ . Then  $Y$  is uniformly distributed on  $(0, 1)$ , that is  $P(Y \leq y) = y$ ,  $0 < y < 1$ .*

The proof of the theorem is in [12], and hinges on the fact that the graph of the CDF is a non-decreasing function.

This theorem allows a transformation of any distribution with a well-defined CDF to the uniform distribution. Since the model starts with a random value obtained from a  $U(0, 1)$  distribution, it can use the inverse transformation of the CDF to generate the random value from the desired probability distribution in the model [12]. Specifically, if the algorithm generates a random number  $\theta$  from the  $U(0, 1)$  distribution, then the desired random value  $\lambda$  from the applicable distribution  $X$  with well-defined CDF  $F_X(x)$  will be [21]:

$$\lambda = F_X^{-1}(\theta) \tag{2.3}$$

The principal limitation in this methodology is that the distribution for the random value in the model must have a well-defined CDF. The classic example of this limitation is when the desired distribution is Gaussian. No closed-form expression exists for the cumulative distribution function of a Gaussian random variable [21]. For this reason, the

algorithm must use alternative methods of generating a random value from a Gaussian distribution.

One famous method of doing this is the Box-Muller transformation. This technique starts with two independent, identically distributed (IID), randomly generated numbers from the  $U(0, 1)$  distribution,  $\theta_1$  and  $\theta_2$ . Using the equations below, the Box-Muller transformation generates two random values from a  $N(0, 1)$  distribution,  $\lambda_1$  and  $\lambda_2$  [11]:

$$\lambda_1 = \sqrt{-2 * \ln(\theta_1)} * \cos(2\pi\theta_2) \quad (2.4)$$

$$\lambda_2 = \sqrt{-2 * \ln(\theta_1)} * \sin(2\pi\theta_2) \quad (2.5)$$

Many real-world systems are modeled best with Gaussian distributions. However, the actual distribution rarely has mean 0 and standard deviation of 1. Consequently, the final step is to transform the random values  $\lambda_1$  and  $\lambda_2$  from the  $N(0, 1)$  distribution into the desired  $N(\mu, \sigma)$  distribution. The conversion algorithm is below:

- Generate a random value ( $\lambda$ ) from the  $N(0, 1)$  distribution as described above.
- Transform the random value ( $\lambda$ ) into an output ( $\xi$ ) from the desired  $N(\mu, \sigma)$  distribution as follows:  $\xi = \mu + (\sigma * \lambda)$ .
- Use the random value  $\xi$  in the relevant section of the simulation code.

This simulation uses the *randn* function in MATLAB to generate Gaussian random numbers useful to the simulation. In particular, the *randn* function obtains a  $U(0, 1)$  random number (as described before) and then transforms this number into a random value from a  $N(0, 1)$  distribution [29]. The simulation then converts this random value into the desired  $N(\mu, \sigma)$  distribution as discussed above. This technique is particularly useful in the simulation because many of the assumptions will be modeled



as Gaussian distributed random variables with known means and variances.

Consequently, this basic methodology will be used throughout the simulation to generate random values of normally distributed random variables.

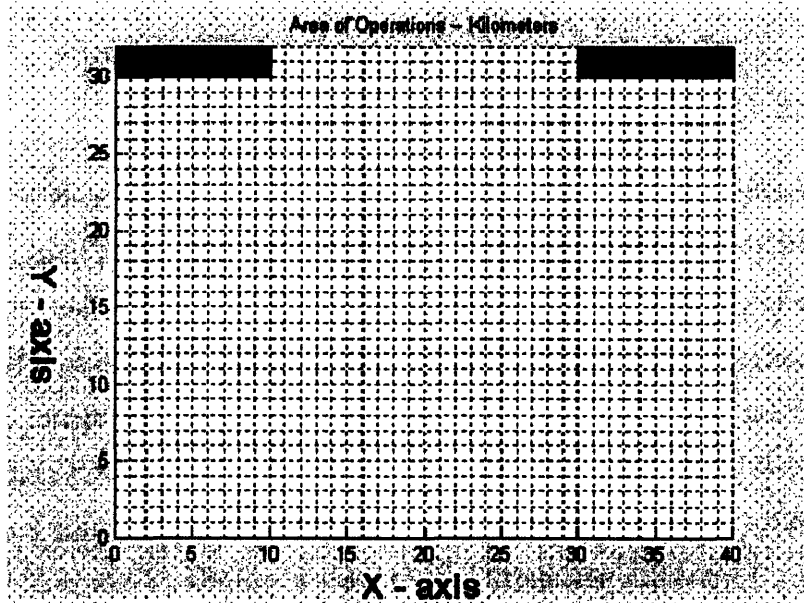
### **2.3 Area of Operations**

After establishing a legitimate source of random numbers, the next task in creating this simulation is defining the area of operations. The capabilities of current and future UUVs allow for a wide variety of operational environments. These may range from open-ocean, search and survey missions to clandestine operations in threat waters [30]. In this research, the focus will be on monitoring a naval chokepoint. By definition, this chokepoint will be any area that restricts maritime movement into a channel. Obvious examples could include the entry to an enclosed bay, the natural passages between islands, deep-water passages between areas of comparatively shallow water, or man-made obstacles such as canals.

For simulation purposes, the algorithm uses a water passage between two land masses that is 20 kilometers wide. Given the energy restrictions and relatively slow speeds of modern UUVs, the simulation restricts the area of operations for the UUV(s) to be a rectangle of open ocean adjacent to the naval chokepoint that is 40 kilometers wide and 30 kilometers long. The simulated UUV(s) must be able to intercept enemy vessels within this area of operations, in support of the projected future doctrine for UUV employment [30].

Figure 2.1 below shows the area of operations for the simulation. The two rectangles at the top represent land, and the 20 kilometers of open water between the two land masses represents the naval chokepoint. Consequently, the area of operations is defined in the figure below where each grid represents one square kilometer and the 20 kilometer wide area between the land masses represents the extreme southerly end of the naval chokepoint. The UUVs in the simulation will attempt to track and intercept the threat vessel as it leaves the chokepoint. Although the concepts of North, South, East, and West are irrelevant to the conceptual algorithm, this thesis uses the terms to help

orient the reader to the figures and to make the figures more understandable. The figures assume the traditional orientation, with North being at the top of the figure, the positive  $x$ -axis pointing East, and the positive  $y$ -axis pointing North.



**Figure 2.1 – Area of Operations for the Simulation**

## **2.4 Sensor Array**

The basic tool that the UUVs use to monitor the naval chokepoint is an array of sensors. The sensors are emplaced in a known pattern that completely covers the width of the chokepoint and includes several layers of sensors along the length of the area of operations. The multiple layers of sensors are intended to increase the length of time that the system has sensor coverage of the threat vessel. Since sensor assets are limited, the sensor pattern covers the entire width of the chokepoint with two layers of sensors, but then concentrates the remaining sensors on additional layers that cover the center of the chokepoint. This design of the sensor array is consistent with the assumptions used to model the enemy movement, as seen in Section 3.3.

Inherent in the emplacement of the sensor array are the corresponding capabilities of that sensor system. There are a wide variety of existing and proposed sensor systems

that could accomplish this task. This research, uses the following assumptions about the sensors, consistent with information about a common sensor system [1]:

- The sensors have an effective sensing radius of 1000 meters.
- The sensors are fixed at a central point and can detect arrivals in any direction within the sensing radius.
- The sensors only report position. The algorithm must derive speed and bearing information after recording more than one visible observation.

The basic design of the sensor array is in Figure 2.2 below. Each circle represents the effective coverage of one sensor, located at the center of the circle. Note that the sensors in the figure are numbered from 1 to 32. These numbers correspond to the sensor identification number, an administrative assignment used to determine which sensor recorded the threat vessel information. Later output will not include the sensor numbers on the figures.

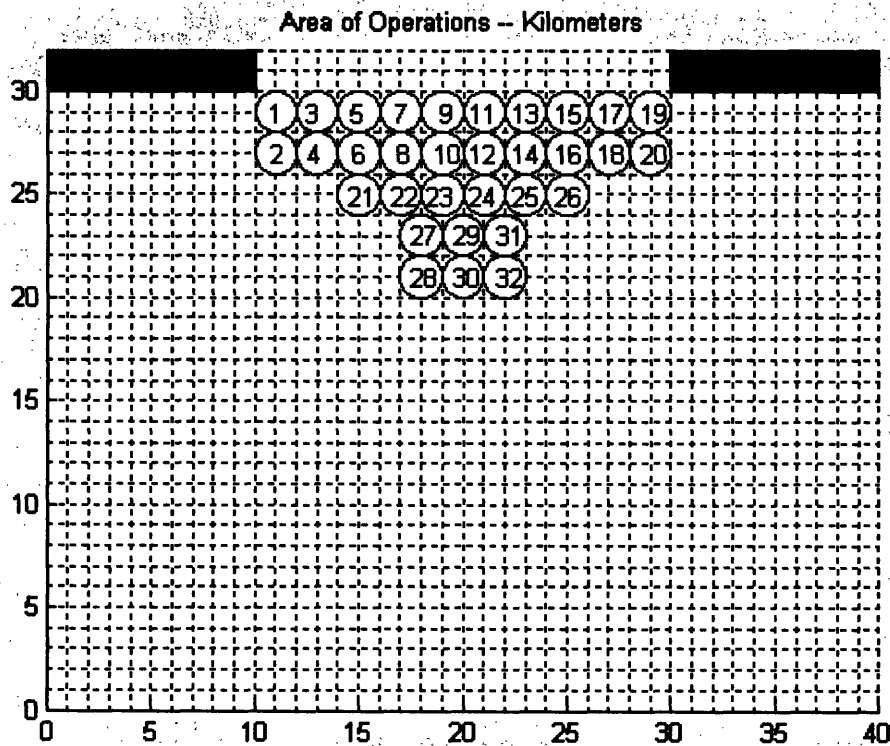


Figure 2.2 Numbered Sensors in Basic Sensor Array Layout

The sensors in Figure 2.2 are emplaced with perfect spacing in both the  $x$  and  $y$  directions, and with no overlap. This is not a reasonable assumption for a system that is emplaced by humans and/or vehicles and must contend with ocean currents, inaccuracies in navigational information, and the inherent displacement from the surface location caused by the sensor sinking to the ocean floor at a non-perpendicular angle to the surface. The simulation takes these inaccuracies into account by allowing the sensor locations to vary in both the  $x$  and  $y$  directions.

Instead of assigning the exact location for each sensor's center, the algorithm makes the center coordinates of each sensor random variables. These random variables follow Gaussian distributions with mean values at the ideal sensor locations (as shown in Figure 2.2 above.) The simulation translates a scaled random output from a  $N(0, 1)$  distribution into a signed shift of the  $x$  coordinate from a  $N(x_{IDEAL}, \sigma_x)$  distribution and then repeats the process with an IID random variable to create an independent, signed shift of the  $y$  coordinate from a  $N(y_{IDEAL}, \sigma_y)$  distribution. For both distributions above, the standard deviations ( $\sigma_x$  and  $\sigma_y$ ) are assumed to be 0.033 kilometers. The simulation uses this value for the standard deviation so that the sensor's center coordinates will not exceed 10% of the sensor radius ( $\pm 100$  meters from ideal) unless the random outcome from the  $N(0, 1)$  distribution corresponds to an event more than 3 standard deviations from the mean.

The modeling assumptions throughout this thesis choose to describe many of the stochastics as Gaussian random variables. When this happens, the assumptions use a common methodology to determine the assumed standard deviation of the Gaussian distribution. The basic properties of the Gaussian distribution are such that approximately 99.7 % of the random outcomes will be within  $\pm 3 * \sigma$  from the mean [23]. If the modeled Gaussian process has well-defined, desired limits (such as  $\pm 100$  meters from the mean in the example above) the model will assume that the standard deviation of the corresponding Gaussian random variable is one-third of the magnitude of the desired limit from the mean. This assumed process provides a common modeling methodology for all assumed Gaussian distributions in this thesis. If the limits are strict, then the distributions are truncated at the limits (corresponding to  $\pm 3 * \sigma$  from the

mean). If the limits are “desired” but not mandatory, then the assumed Gaussian distributions may produce a random outcome that is beyond the desired limits only if the corresponding call to the random number generator produces a value beyond three standard deviations from the mean after transformation to the desired  $N(\mu, \sigma)$  random variable. This should happen approximately 0.3% of the time. [23]

In the example above for sensor center coordinates, the assumed standard deviations in both the  $x$  and  $y$  directions are 0.033 kilometers. This suggests that approximately 99.7% of the realizations of the random variables for the  $x$  and  $y$  shifts will be less than 100 meters. In this case, the 100 meter shift limit is desired but not strict. This means that the shift value will exceed 100 meters approximately 0.3% of the time. The result is a more realistic distribution of sensors. Figure 2.3 below shows one such distribution. The figure is enlarged to emphasize the random placement of the sensors.

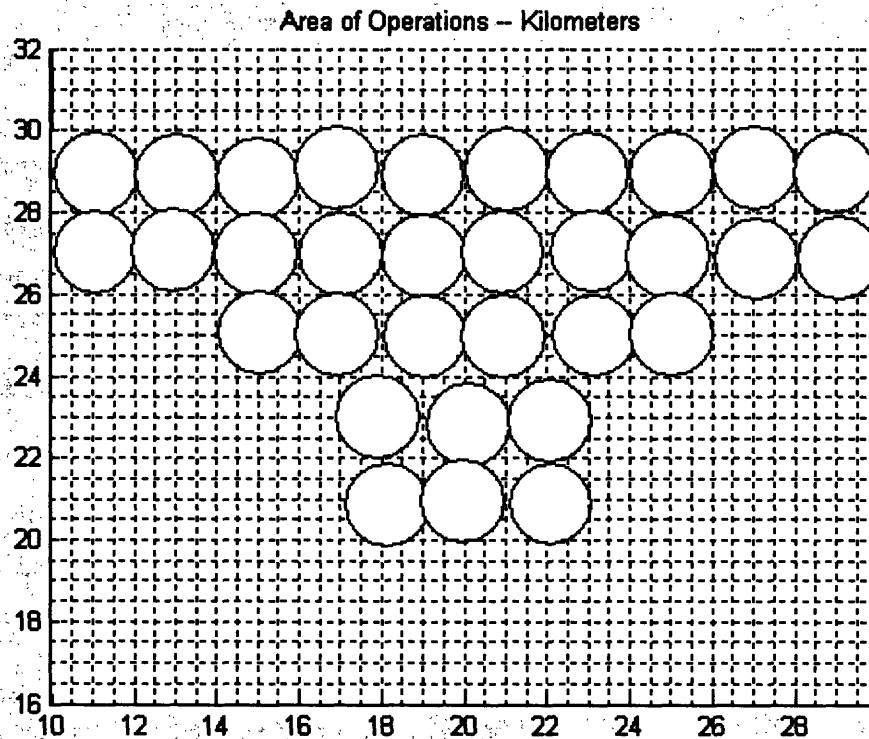


Figure 2.3 Sensor Array with Randomness in Sensor Location

## **Chapter 3 Threat Vehicles**

The previous chapter focused on the description of the underlying simulation environment. This chapter will focus on the threat vehicles and the assumptions behind their movement characteristics. These include foundational assumptions concerning the threat vehicle's capabilities and doctrine, and movement assumptions that described the stochastics governing threat vehicle movements.

### ***3.1 Foundational Assumptions***

Before considering the detailed assumptions that govern specific aspects of threat vehicle movement, it is important to consider foundational assumptions that control the overall behavior of the threat vehicles. These are discussed in the paragraphs below.

Because of the intended clandestine nature of the task, the initial assumption for threat vehicle movement is that the threat vehicles are unaware of the sensor monitoring or the presence of the UUVs. This is a reasonable assumption because the sensors are passive in nature without detectable emissions such as sonar pings or similar signals. Furthermore, the UUVs are small, clandestine vehicles with quiet motors. Consequently, the threat vehicles have no ability to detect either the sensors or the UUVs. This assumption eliminates the need to model the rational responses of an intelligent adversary using game theory or similar disciplines.

The model assumes that the threat vehicle is a naval vessel traveling through the chokepoint and entering the open ocean to the immediate south of the chokepoint. The exact description of the threat vehicle is not important. The combination of sensors and UUVs allow for successful monitoring of the chokepoint against any naval threat ranging from large surface ships, to small patrol boats or submerged vehicles. Since visual identification of threat vessels on the surface may be accomplished through the use of satellite photography or reconnaissance aircraft, the sensor / UUV system would be particularly useful against a submerged threat that is less easily seen with other assets.

The model also assumes that the threat vessel must pass through the chokepoint. This assumption ensures that the sensor array is in the right place to optimally locate the threat vehicles. Specifically, this assumes that the threat vehicles cannot bypass the chokepoint or approach the area of operations from another direction. The physical placement of the sensor/UUV system relative to the entire battle-space is beyond the scope of this research, and is assumed to be optimal.

### **3.2 Movement Assumptions**

The model requires several assumptions concerning the movement of the threat vehicles. Critical to the movement assumptions is the earlier assumption that the threat vehicle is unaware of the friendly sensors and UUVs. In practice, this means that the threat vehicle will not take defensive measures or stray from its intended course. The threat movement assumptions are discussed in the sections below.

#### **3.2.1 Threat Vessel's Intent**

The threat vehicle's intent (mission) is to pass through the open ocean south of the chokepoint en-route to a specific location. This assumes that the threat vessel has a simple, traveling mission to move to a known point assumed to be many kilometers from the chokepoint. Specifically, this assumes that the threat vessel does not have a defensive or patrolling mission in the vicinity of the chokepoint that will result in erratic maneuvers in the area of operations.

#### **3.2.2 Arrival Distribution Across the Chokepoint**

The best understanding of likely threat doctrine is that the threat vessel will prefer to transit through the middle of the chokepoint. The center usually corresponds to deeper water and more overall flexibility of movement for the threat vessel. Based on the

previous assumption of clandestine monitoring of the chokepoint, the threat vessel would have no reason to avoid the generally safer waters in the center of the chokepoint. The center of the chokepoint is at the coordinates (20, 30). Since the y coordinate is 30 for all points along the width of the chokepoint, identifying the initial location of the threat vessel corresponds to identifying the x coordinate. The algorithm assumes that the arrival distribution of the threat vehicle across the width of the chokepoint is a normally distributed random variable (truncated at the ends of the chokepoint) with a mean of 20 kilometers (center of the chokepoint) and a standard deviation of 3.33 kilometers.

$$X \text{ coordinate of initial point} \sim N(20, 3.33) \quad (3.1)$$

The algorithm truncates the new distribution at  $x = 10$  kilometers and  $x = 30$  kilometers by testing the generated value of the x coordinate to ensure that it is between 10 and 30 (the physical ends of the chokepoint in the area of operations). A random variable output that exceeds these dimensions would correspond to an unlikely random event more than 3 standard deviations from the mean. If this happens, the simulation rejects the extreme value and generates a new value using the same procedures as before.

### 3.2.3 Initial Bearing of Threat Vessel

For consistency, the term “bearing” in this thesis is defined as the measurement of degrees East of North. Given the coordinate system for the area of operations described in Figure 2.1, a bearing of 0 degrees is due North, 90 degrees is due East, 180 degrees is due South, and 270 degrees is due West. The initial bearing of the threat vehicle when it leaves the chokepoint is a normally distributed random variable (truncated at 90 degrees and 270 degrees) with a mean of 180 degrees and a standard deviation of 30 degrees.

$$\text{Initial bearing} \sim N(180, 30) \quad (3.2)$$

The intuition behind this assumption is that the threat vessel is heading generally South (generally away from the chokepoint and into open ocean.) The algorithm truncates the distribution at 90 degrees and 270 degrees by testing the output. This ensures that the threat vessel is leaving the chokepoint and heading into open ocean. Once again, a random variable output that exceeds these bounds would correspond to an unlikely



random outcome more than three standard deviations from the mean. If this happens, the simulation rejects the extreme value and generates a new value using the same procedures as before.

### 3.2.4 Initial Speed of Threat Vessel

The initial speed of the threat vehicle when it leaves the chokepoint is a normally distributed random variable (truncated at 4 and 12 knots) with a mean of 8 knots and a standard deviation of 1.33 knots.

$$\text{Initial speed (knots)} \sim N(8, 1.33) \quad (3.3)$$

Section 3.1.2 argued that the sensor / UUV system would be effective against any traditional, naval threat. However, the system would be most useful against submerged threats since surface vessels could be more easily tracked with less sophisticated systems. Consequently, the expected velocities of threat submarines are the benchmark used to justify the initial speed assumption. Although the maximum velocities vary greatly depending on the specific types of submarines, the model assumes that the mean speed is 8 knots with a maximum speed of 12 knots and a minimum speed of 4 knots. Although the maximum possible velocities of threat systems may exceed 12 knots, it would be inconsistent with threat doctrine to exceed 12 knots in the potentially perilous waters of a naval chokepoint. Consequently, the model truncates the initial speed distribution at 4 knots and 12 knots.

Since the simulation uses metric units, the speed must be converted to meters per second using the fact that 1 knot equals 0.514 meters per second. Using this conversion factor, the distribution for the initial speed random variable (in meters per second) becomes  $N(4.112, 0.684)$ . The distribution is truncated at 4 knots (2.058 m/s) and 12 knots (6.173 m/s) by testing the output. If the random variable output exceeds these bounds, the simulation rejects the extreme value and generates a new value for the initial speed using the same procedures as before.

### 3.2.5 Changes in Threat Vessel Bearing

After leaving the chokepoint, the threat vehicle changes bearing according to a Poisson process with an expected rate of 1 bearing change per hour. This assumption relies on the earlier assumption that the vehicle's intent is to travel to a destination many kilometers away and not to patrol in the vicinity of the chokepoint. After setting an initial course (upon leaving the chokepoint), the threat vessel will probably not need to change directions for a long time.

To implement this assumption in the simulation, the algorithm uses a Poisson process with arrival rate  $\lambda = \frac{1}{60}$  over a time interval of length  $\tau = 1$  minute. In this structure, an "arrival" is the occurrence that the threat vessel changes bearing in a given discrete simulation update (1 minute). By the fundamental properties of a Poisson process, the small interval probabilities  $P(k, \tau)$  satisfy the expressions below [6],

$$\begin{aligned} P(0, \tau) &= 1 - \lambda\tau + o(\tau) \\ P(1, \tau) &= \lambda\tau + o_1(\tau) \\ P(k, \tau) &= o_k(\tau), \quad k = 2, 3, \dots \end{aligned} \tag{3.4}$$

where  $o(\tau)$  and  $o_k(\tau)$  are functions of  $\tau$  that satisfy

$$\lim_{\tau \rightarrow 0} \frac{o(\tau)}{\tau} = 0, \quad \lim_{\tau \rightarrow 0} \frac{o_k(\tau)}{\tau} = 0 \tag{3.5}$$

It is important to establish that the value of  $\tau$  is very small relative to the design of the system parameters. The interception algorithm / chokepoint monitoring algorithm is intended to operate autonomously for a period of many days. Given the simulation structure, all updates occur at discrete moments separated by a period of one minute. As such,  $\tau = 1$  minute is the smallest measurable unit of time in this simulation, and it is sufficiently small to use the probability expressions in Equation 3.4 above. Furthermore, the  $o(\tau)$  and  $o_k(\tau)$  terms are negligible compared to  $\tau$  when  $\tau = 1$  minute.

Given the probability expressions in Equation 3.4 and ignoring the negligible terms, the probability of one arrival in a one minute period is exactly  $\lambda$ . Furthermore, the "memorylessness property" or "fresh-start property" of a Poisson process guarantees that

each subsequent arrival (that the threat vehicle changes bearing) will be independent of previous bearing changes and will follow the same probability distribution as before [6]. The simulation runs in discrete time with updates every 1 minute. The objective is to characterize the probability of the Poisson arrival process (with an expected rate of 1 arrival every 60 minutes) in terms of a randomly generated number between 0 and 1. The following algorithm accomplishes this task:

- Given the probability expressions in Equation 3.4, the probability that a bearing change occurs in any given update is equal to  $\lambda$ . Since  $\lambda = \frac{1}{60}$ , this probability is 0.0167.
- Generate a random number between 0 and 1, and compare this random number to the value of  $\lambda$ .
- If the randomly generated number is less than  $\lambda$ , this corresponds to the event that the threat vessel changes bearing.

If the threat vehicle changes bearing after leaving the chokepoint, the change in bearing is a normally distributed random variable with a mean of zero and a standard deviation of 30 degrees.

$$\text{Change in bearing} \sim N(0, 30) \quad (3.6)$$

This assumption is very similar to the assumption used to define the initial bearing of the threat vessel upon exiting the chokepoint. This assumption relies heavily on the earlier assumption that the threat vessel is traveling to a specific destination. Consequently, its initial bearing (set upon exiting the chokepoint) is probably very close to the optimal bearing, and any changes in bearing would probably correspond to minor corrections around the previous bearing. This suggests that the mean value for a change of bearing calculation should be the previous vehicle bearing. The algorithm uses the same standard deviation as before (30 degrees) for consistency. Additionally, the model continues to truncate the newly calculated bearing at 90 degrees and 270 degrees in order to support the initial assumption that the threat vessel's objective is somewhere south of the

chokepoint. (Otherwise, the threat vessel would not have negotiated the chokepoint in the first place.)

### 3.2.6 Changes in Threat Vessel Speed

After leaving the chokepoint, the threat vehicle changes its speed according to a Poisson process with an expected rate of 1 speed change every 2 hours. Again, this assumption rests on the earlier assumption that the threat vehicle's sole intent is to move from one point to another in the most efficient means possible. Under this assumption, it is very unlikely that the threat vehicle would change speed after establishing an initial speed upon exiting the chokepoint.

To implement this assumption about the probability of the threat vessel changing speed at a discrete, one minute update, the algorithm uses a similar procedure as the previous assumption.

- Given the small interval probabilities of the Poisson distribution, the probability that a change in speed occurs in any given update (where  $\tau = 1$  minute) is equal to  $\lambda$ . Since  $\lambda = \frac{1}{120}$ , (one speed change every two hours) this probability is 0.0083.
- Generate a random number between 0 and 1, and compare this random number to the value of  $\lambda$ .
- If the randomly generated number is less than  $\lambda$ , this corresponds to the event that the threat vessel changes speed in that discrete simulation update.

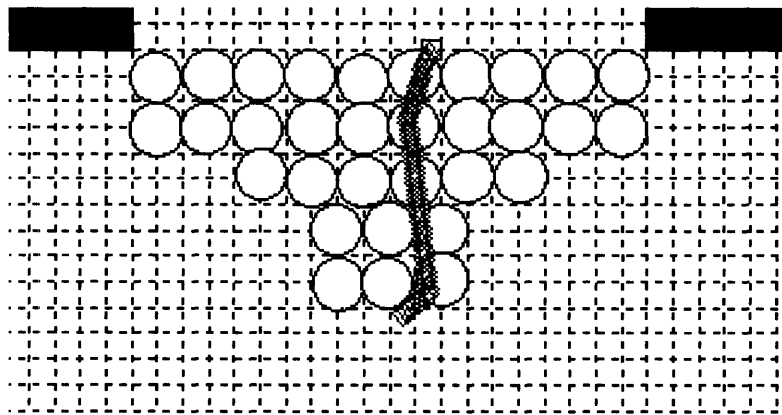
If the threat vehicle changes its speed after leaving the chokepoint, the change in speed is a normally distributed random variable with a mean of zero and a standard deviation of 1.33 knots.

$$\text{Change in speed} \sim N(0, 1.33) \quad (3.7)$$

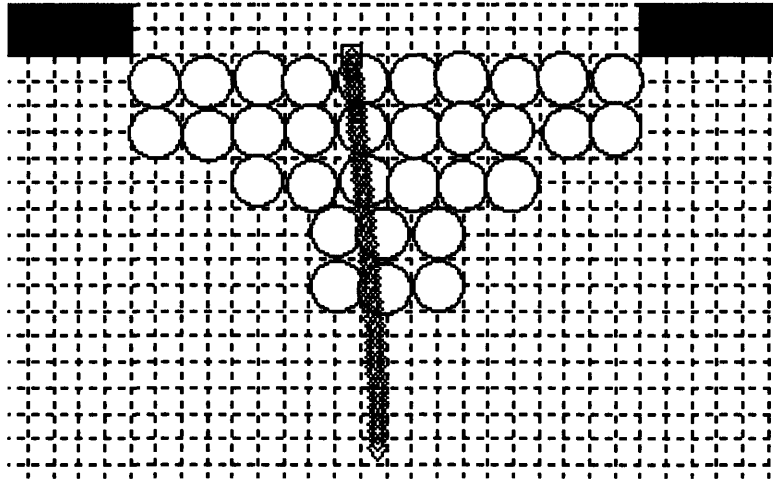
This assumption is very similar to the assumption used to define the initial speed of the threat vessel upon exiting the chokepoint. Again, the model assumes that the initial speed (set upon exiting the chokepoint) is probably very close to the optimal speed given the capabilities and mission specific data of the threat vessel. Any changes in speed would probably correspond to minor corrections. Consequently, the algorithm sets the mean value for the new speed as the previous speed. The model also uses the same standard deviation as before (1.33 knots) for consistency. However, the algorithm continues to truncate the distribution at the maximum and minimum velocities (12 knots and 4 knots) in order to support the earlier assumptions about the expected range of threat vehicle velocities.

### **3.3 Results of Threat Vessel Modeling**

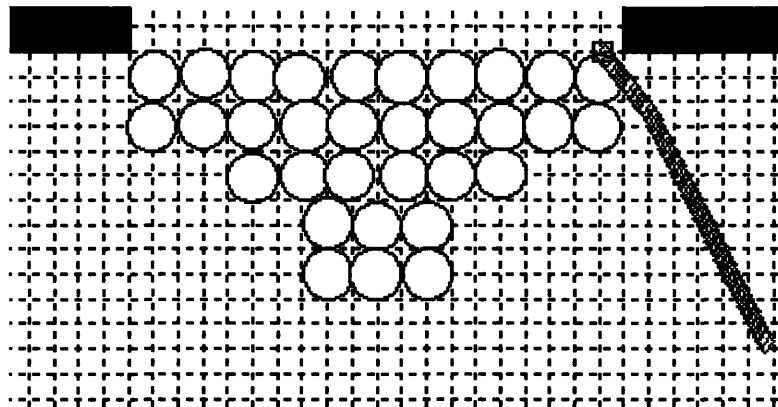
The simulation accurately describes the inherently stochastic nature of the threat vessels' arrival characteristics (position, speed, and bearing), and their probabilities of changing these characteristics at each discrete update. Figure 3.1, Figure 3.2, and Figure 3.3 depict the stochastic arrivals and movements of a single threat vessel over the course of 50 minutes (50 discrete updates at 1 minute intervals). The square represents the initial location of the threat vessels across the width of the chokepoint, and the series of diamonds show the locations of the vessels at each of the 50 discrete updates.



**Figure 3.1 – 50 Time Steps of one Threat Vessel with Random Bearing Changes**



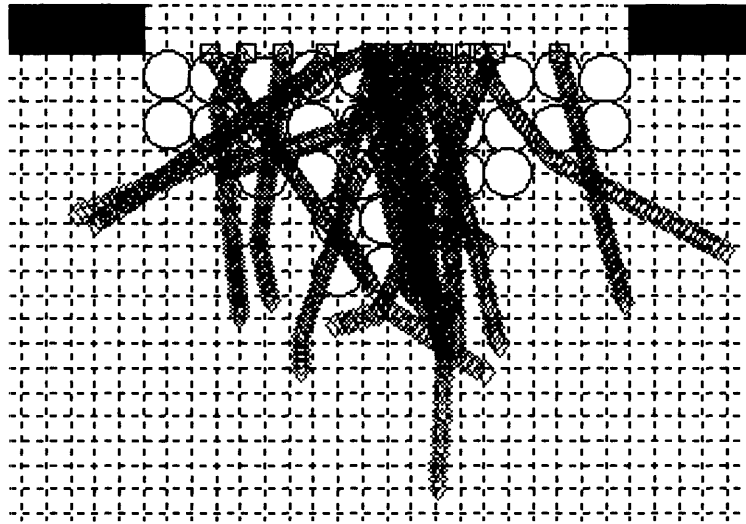
**Figure 3.2 – 50 Time Steps of one Threat Vessel with No Bearing Changes**



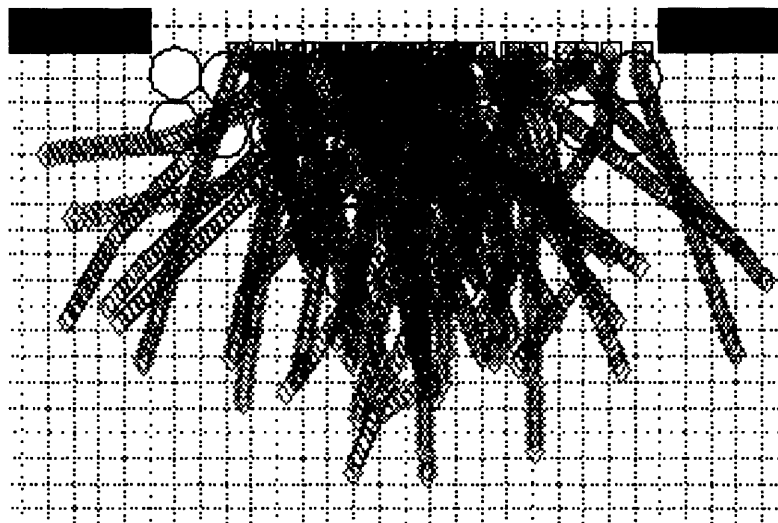
**Figure 3.3 – 50 Time Steps of one Threat Vessel – Unlikely Random Outcome**

Notice that Figure 3.1 and Figure 3.2 show fairly typical movement (relatively centered), while Figure 3.3 shows a statistically unlikely event because the arrival happened at the extreme edge of the chokepoint (the extreme tail of the Gaussian arrival distribution across the chokepoint). However, all three figures show actual random output from the simulation, reflecting the stochastic arrival and movement probabilities of the threat vessels.

Analyzing large numbers of random sample paths should verify that the model accurately reflects the movement assumptions for threat vehicles. Figure 3.4 shows 20 random samples drawn on one graph, and Figure 3.5 shows 50 random samples drawn on one graph. Each random observation tracks 50 minutes of threat vessel movement.



**Figure 3.4 – 20 Random Iterations of the Threat Vessel Algorithm, 50 Time Steps per Iteration**



**Figure 3.5 – 50 Random Iterations of the Threat Vessel Algorithm, 50 Times Steps per Iteration**

By overlapping the output from many random iterations, Figures 3.4 and 3.5 show the long-term trends for threat vessel movements. Note that the threat vehicle paths are of different lengths and have different bearings because the speeds and bearings of each threat vessel are independent random variables. Note also that changes in the bearings or velocities of the vessels appear to happen rarely. When they do happen, the changes appear to be relatively minor. Finally, note that the overall trends are as expected with the majority of the threat activity happening near the center of the chokepoint area. This

is particularly obvious in Figure 3.5, where the pattern of movements somewhat approximates an upside-down, bell-curve shape. This pattern supports the placement pattern of the sensors.



## Chapter 4 Friendly Vehicles

The previous chapter focused on the stochastic processes that describe the threat vehicles in the simulation. This chapter will focus on the friendly vehicles that work in conjunction with the sensor array to monitor the naval chokepoint. The moving components of the sensor system are the Unmanned Undersea Vehicles (UUVs).

### 4.1 Swarm Intelligence

One school of thought argues that problems of the type addressed in this research are best solved through the use of swarm intelligence. Swarm intelligence may be defined as “a form of collective intelligence which relies on the capabilities of several minimally intelligent but autonomous individuals” [22]. Swarm intelligence may be applied to research of this type by employing a collection of simple UUVs that rely on local sensing and reactive behaviors to intercept threat vessels. One such study was conducted by Melissa St. Peter and Ken La Pointe at the Naval Undersea Warfare Center, Division Newport (NUWC) [20].

In their study, presented at the June 2004 Military Operations Research Society Conference, St. Peter and La Pointe attempted to solve a very similar interception problem by “swarming” large numbers of UUVs towards the threat vehicle with the assumption that many, generally oriented UUVs would succeed in successful intercepts. In fact, their hypothesis was correct and the model worked. However, the algorithm rested upon a set of assumptions radically different from the assumptions used in this thesis. St. Peter and La Pointe assumed the existence of a large number of simple UUVs instead of a small number of sophisticated UUVs. Consequently, St. Peter and La Pointe’s model assumed that large numbers of UUVs would be available.

Swarming is probably a useful alternative in some scenarios, particularly if the system is operating in friendly waters near UUV maintenance / recharging facilities. However, swarming UUVs has many disadvantages that this thesis attempts to avoid. In particular, swarming many UUVs uses a tremendous amount of energy. An algorithm

that swarms large numbers of UUVs would cause most or all of the swarming UUVs to deplete their energy supply while only intercepting one threat vessel. This is fundamentally inefficient and should be avoided if there is a method of intercepting the threat using fewer resources. Additionally, swarming large numbers of UUVs in the vicinity of the naval chokepoint would dramatically increase the likelihood that the threat vehicles would learn of the supposedly clandestine monitoring.

The goal of this research is to develop a model that solves the interception problem with a high probability of success while conserving resources and remaining clandestine. In particular, this research will consider algorithms that assign only one UUV to pursue and intercept each threat arrival. This encourages the desired clandestine nature of the operation, while conserving fuel resources. The assumptions below will clarify these important distinctions and motivate the development of the interception algorithm in the later half of this chapter.

## ***4.2 UUV Assumptions***

The chapter begins with some assumptions about the UUVs and the definition of success in this model. All of these assumptions are consistent with current or projected technologies.

### **1. Definition of Successful Interception**

The purpose of this research is to design an autonomous, interception algorithm that controls the UUVs in their effort to intercept threat vehicles in the vicinity of the naval chokepoint. The algorithm defines a successful intercept as any time that the UUV is within 500 meters of the threat vessel. After the chasing UUV is within this range, the internal sensors on the UUV can control further movements to ensure that the UUV remains in contact with the threat vessel over time. This thesis does not model the specific behaviors that happen after the UUV is within 500 meters since these actions would differ depending on the specific mission that the UUV was told to perform.

## **2. Number of UUVs Available**

The algorithm assumes that the simulation starts with 4 UUVs to monitor the sensor array. Since UUVs are an expensive, developing technology, it is unlikely that there would be large numbers of UUVs in the water at any given naval chokepoint.

## **3. UUV Longevity / Servicing**

By its design, a modern UUV is generally small and has a limited energy supply. After depleting its fuel (typically an electric battery), the UUV must be serviced and refueled. This servicing requires the UUV to be unavailable for assignment, and consequently causes the system to monitor the chokepoint with fewer UUVs available.

Servicing a UUV occurs after it has completed an interception attempt of a threat vehicle. This corresponds to the earlier assumption that a UUV has very limited fuel storage capability. After chasing a threat vehicle to the point of desired interception and conducting any possible follow-on mission, the UUV would not have enough fuel for assignment to another threat arrival.

## **4. Reassignment and Availability**

Any UUV that is not in active pursuit of a threat vehicle or in service is considered available for assignment to an incoming arrival. If a UUV begins a chase sequence, but the algorithm reassigns another UUV to the threat vehicle, the previously assigned UUV is still available for later missions. However, if this process has occurred four times on the same UUV, then the model assumes that the UUV has depleted its fuel and must be serviced. The earlier assumption requires that each iteration starts with 4 UUVs, but a UUV may still require service in the course of one iteration in the unlikely event of four reassignments for the same UUV in one iteration. For more details about the re-assignment algorithm, refer to Chapter 5.

## **5. Disposition of UUVs**

The model assumes that a UUV is either moving or dormant. If it is moving, a UUV travels at a speed of exactly five knots (2.57 meters/second). Admittedly, this is unrealistic because of the transition velocities during the period of positive acceleration as the UUV moves from being dormant to its maximum speed. However, the UUVs can achieve maximum speed very quickly, and this differential is trivial in a system that defines successful interception as  $\pm 500$  meters. UUVs start each mission dormant. This is to conserve fuel. UUVs must have initial (dormant) locations within the sensor radii of one of the thirty-two sensors. This ensures that the UUVs have good communications (assumed to be perfect) with the sensor system at the start of each iteration.

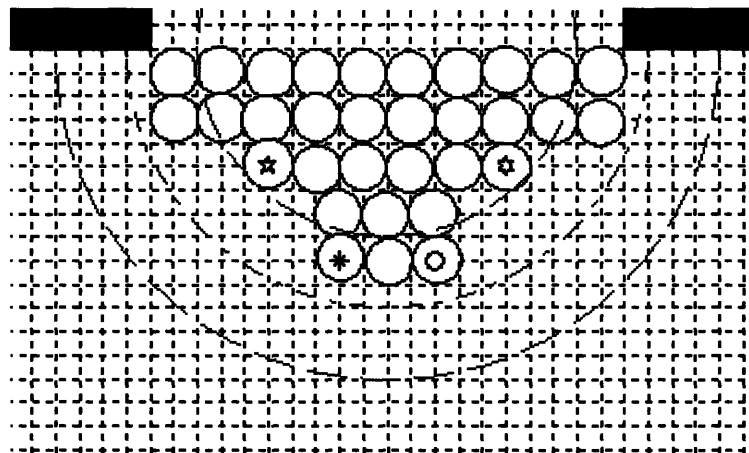
### ***4.3 Definition of the Engagement Area***

A basic tenet of military operations is to select the time and place at which to engage the enemy. By doing so, the tactician accentuates strengths while minimizing the effects of any weaknesses. The primary strength of this model is the clandestine nature of the sensors and the UUVs. However, the UUV's principal weakness is its slow maximum speed (5 knots ~ 2.57 meters/second). The algorithm employs its strengths and minimizes the effects of its weakness in two ways: by defining the desired arc of interception and through the initial locations of the UUVs. The sections below discuss these two areas in greater detail.

#### **4.3.1 Definition of the Desired Arc of Interception**

The first method is to define the desired arc (the physical location in the area of operations) to intercept the threat vehicle, centered on the naval chokepoint. By selecting the preferred interception area, the model employs its strengths (clandestine monitoring) while minimizing its weakness (slow UUV speed). In Figure 4.1 below, the interception

arc corresponds to the middle, dashed arc defined as the lower half of a circle centered at the middle of the naval chokepoint with a radius of eleven kilometers. The selection of the initial length of this radius was based on an effort to locate the arc just outside of the sensor system. The optimization problem that determines the best length of this radius in the absence of human intervention is the focus of Chapter 6. The desired arc of interception becomes the basis for defining the expected number of 1 minute time steps needed for the interception algorithm. (For more details on this process, refer to Section 4.6). Since the threat vehicle may change speed or bearing after the initial interception calculations, the model assumes that the desired engagement area is further defined as the area between the two concentric arcs spaced three kilometers from the middle arc. The boundaries of the desired engagement area correspond to the outside, dashed arcs in Figure 4.1 below.



**Figure 4.1 – Engagement Area Defined and Initial Locations of the 4 UUVs**

### **4.3.2 Definition of the Initial Locations of the UUVs**

The second method used by the model to employ strengths and minimize the effects of weaknesses is to position the UUVs in favorable initial locations. In the base case simulation, the model assumes initial locations of the four UUVs at the expected center coordinates of sensors 21, 26, 28, and 32. The initial locations of the four UUVs are shown in Figure 4.1 above, where the 4 UUVs correspond to the 5 pointed star, the 6

pointed star, the asterisk, and the small circle. Given the desired engagement area, these initial locations allow the UUVs to overcome the potential discrepancy between the chaser UUV's speed and the threat vehicle's speed by initially locating near the desired engagement area, but within the sensing radius of one of the sensors.

The initial locations of the four UUVs were not defined through a rigorous optimization process. Rather, they were placed through intuition and experience after simulating many iterations of the threat vehicles' stochastic movements. Chapter 7 of this thesis will challenge the initial placement of the UUVs through a dynamic programming optimization problem.

#### **4.4 Outline of the Interception Algorithm**

This section outlines the basic interception algorithm that will be defined in greater detail in the remainder of Chapter 4.

1. The algorithm starts after the sensor system has observed the threat vessel for the second time. Using these two observations, the algorithm computes the threat vessel's speed and bearing.
2. Given the threat vessel's location and bearing (calculated in step 1), the algorithm determines the initial point of desired interception. This is the point where the threat vessel will cross the interception arc, given its current disposition.
3. Given the threat vessel's speed (calculated in step 1), the algorithm determines the expected number of 1 minute time steps until the threat vessel arrives at the initial point of desired interception. For the remainder of this thesis, this value is called  $\tau$ .
4. During the second time step, the algorithm assigns a UUV based on which UUV is closest to the expected point of desired interception  $(x_\tau, y_\tau)$ . The potential for reassignment of the chaser UUV is based on the stochastics governing threat vessel movement, and this will be considered in Chapter 5.

5. The assigned UUV begins movement towards  $(x_\tau, y_\tau)$ . The remaining UUVs that are not assigned to the threat arrival begin movement towards coverage positions that redistribute the remaining UUVs within the area of operations.
6. During each time step in which the threat vessel is seen by a sensor, the algorithm recalculates the speed and bearing of the threat vessel and the coordinates  $(x_\tau, y_\tau)$ . The algorithm recalculates the bearing information for the assigned UUV, and it moves towards the updated coordinates  $(x_\tau, y_\tau)$ . The unassigned UUVs continue movement towards their coverage positions.
7. The algorithm terminates as a successful iteration if the UUV intercepts the threat vehicle (defined as  $\pm 500$  meters between the chaser UUV and the threat vessel). The algorithm terminates as an unsuccessful iteration if more than  $\tau$  time steps have occurred and the model calculates that there is no reasonable probability of successful interception.

The sections below elaborate on each of these steps in the basic algorithm.

#### **4.5 Determining Threat Vessel Speed and Bearing**

As seen in Chapter 3, the simulation updates the movement of the threat vehicle at each discrete time step (corresponding to 1 minute between updates). However, the system of sensors and UUVs only observes the threat vehicle during a given time step if it is located within the sensing radius of one of the sensors. The sensors in this model only report the  $x$  and  $y$  coordinates of the threat vessel. Consequently, the algorithm requires two visible observations before it can calculate the speed and bearing of the threat vessel.

The speed calculation is simply the distance traveled by the threat vessel divided by the time. Because the first two visible time steps may or may not correspond to two consecutive simulation time steps, the model considers the time between the visible time steps when calculating the threat vehicle's speed:

$$velocity = \frac{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}}{sim\_time_2 - sim\_time_1} \quad (4.1)$$

Trigometric identities are required to determine the threat vehicle's bearing.

The first step is to determine if the threat vehicle is heading generally east or west (right or left) because it changes the trigometric identities needed.

If  $(x_2 - x_1) \leq 0$ , then the threat vehicle is heading generally West (left) and this corresponds to Figure 4.2(a). If  $(x_2 - x_1) > 0$ , then the threat vehicle is heading generally East (right) and this corresponds to Figure 4.2(b) below. In either case,  $\delta$  is the distance between the points (the hypotenuse) and  $\theta$  is the bearing of the threat vehicle.

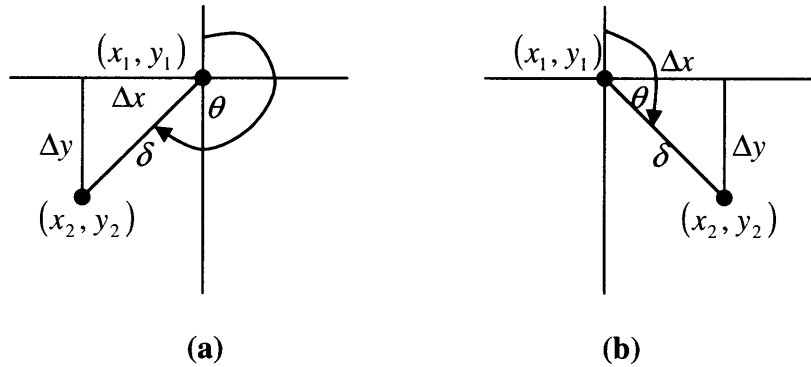


Figure 4.2 – Calculating the Bearing ( $\theta$ ) of the Threat Vessel

Solve for  $\theta$  (in radians) as shown below:

$$\theta = \begin{cases} \frac{3\pi}{2} - \sin^{-1}\left(\frac{\Delta y}{\delta}\right), & x_2 - x_1 \leq 0 \\ \frac{\pi}{2} + \sin^{-1}\left(\frac{\Delta y}{\delta}\right), & x_2 - x_1 > 0 \end{cases} \quad (4.2)$$



## 4.6 Initial Point of Desired Interception

After the algorithm has two visible time steps of the threat vehicle and has completed the calculations of the threat vessel's initial speed and bearing, the model can determine the initial estimate for the desired point of interception. Given the known location of the threat vessel (the most recent visible time step), the algorithm uses the calculated speed and bearing to project the threat vessel's current disposition an arbitrarily large number of time steps into the future. The exact number of time steps is not important as long as the distant point is on the far side of the desired interception arc. Experimentally, the use of 80 time steps works well and is used in this simulation. The algorithm calculates these coordinates, and calls this point  $(x_{dist}, y_{dist})$ . To clarify, the model is not claiming that the threat vessel will arrive at this exact point since the speed and bearing could change at any discrete update. Instead, the algorithm uses  $(x_{dist}, y_{dist})$  as the best estimate of long-term position given the current speed and bearing information. Using the current threat location  $(x_{init}, y_{init})$  and the projected point  $(x_{dist}, y_{dist})$ , the model calculates the slope of the line connecting the two points using the standard slope formula below.

$$slope = \frac{y_{dist} - y_{init}}{x_{dist} - x_{init}} \quad (4.3)$$

The model uses the slope to express the threat vessel's motion as a line.

$$y - y_{dist} = slope * (x - x_{dist}) \quad (4.4)$$

The algorithm then calculates the exact coordinates where this line intersects the arc of desired intersection (the middle arc in Figure 4.1). This point is the solution of two equations with two unknowns:

$$\text{Interception Arc with Radius 11: } (x - 20)^2 + (y - 30)^2 = 121 \quad (4.5)$$

$$\text{Threat Vessel: } y - y_{dist} = slope * (x - x_{dist}) \quad (4.6)$$

This system of equations will have two solutions, but only one of the solutions corresponds to a location in the area of operations. The invalid solution will always be to the North of the chokepoint ( $y > 30$ ), and the valid solution will always be to the South of

the chokepoint ( $y \leq 30$ ). The algorithm defines the valid solution as the initial point of desired interception  $(x_\tau, y_\tau)$ .

#### **4.7 Definition of the Time Horizon “ $\tau$ ”**

As discussed earlier, the algorithm acknowledges that there is very low probability that the threat vehicle will move to exactly the initial point of desired interception without changing speed or bearing. The importance of this point is that the model uses it to define the optimal time horizon ( $\tau$ ). The calculation of  $\tau$  is straightforward except that the earlier expression of speed must be multiplied by 60 to translate from meters per second to meters per minute as shown below:

$$\tau = \frac{\sqrt{(y_{opt} - y_{init})^2 + (x_{opt} - x_{init})^2}}{\text{velocity} * 60} \quad (4.7)$$

The algorithm calculates the value of  $\tau$  only once, after the second visible time step. Successful interception will likely happen at a time different from  $\tau$  (either earlier or later). However, the model uses the calculated value of  $\tau$  as a basis from which to calculate the optimal point of interception for all future time steps of the threat vehicle regardless of any stochastic behavior that the threat vessel may exhibit. Updated calculations of  $(x_\tau, y_\tau)$  will follow the same principles as above except for using  $\tau$  time steps instead of the “arbitrarily large number” of time steps that were used in the initial calculation of  $(x_\tau, y_\tau)$ . In this manner, the algorithm calculates an updated point of desired interception corresponding to each visible time step of the threat vehicle movement.

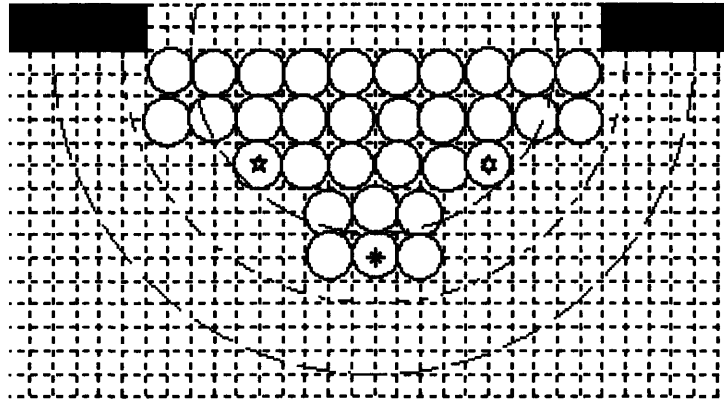
#### **4.8 Assign a UUV**

In the earlier discussion, it was assumed that there would be relatively few UUVs to monitor the sensor array. The model also assumed that additional threat vehicles may cross the sensor array at later times, and that the algorithm should conserve its available

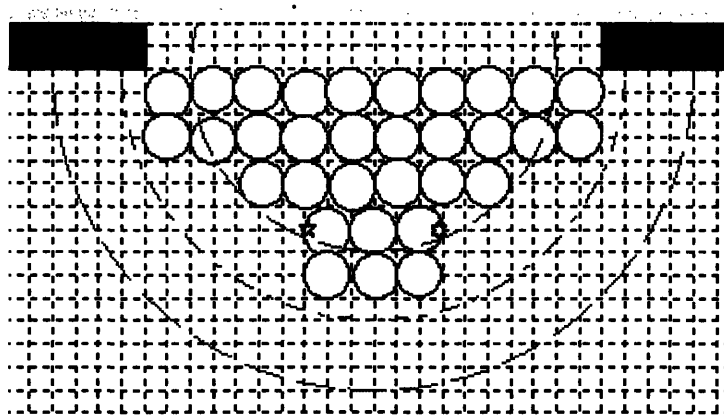
resources to the greatest extent possible so that it could address the later threats. To support these assumptions, the algorithm optimally assigns the UUV that has the highest probability of successfully intercepting each threat arrival. The methodology is to use the initial point of desired interception  $(x_\tau, y_\tau)$  calculated after recording the second visible time step of threat vehicle movement. In the base scenario, the model assumes that there are no obstacles in the water that require special path planning algorithms. Instead, the algorithm calculates the straight-line distance from each UUV to  $(x_\tau, y_\tau)$ , and assigns the UUV with the shortest distance to travel.

Typically, one of the challenges of a sophisticated assignment problem over a long time horizon is to consider the second-order effects of an assignment in an early stage. For example, if the model assigns UUV number 2 to the first threat vehicle, what is the later cost to the model by having assigned UUV 2 to the first threat instead of UUV 1, 3, or 4. This potential “cost to the model” is expressed as the inability of the remaining UUVs to cover the gap once filled by UUV 2. This could allow future threat vessels to pass through the area once covered by UUV 2.

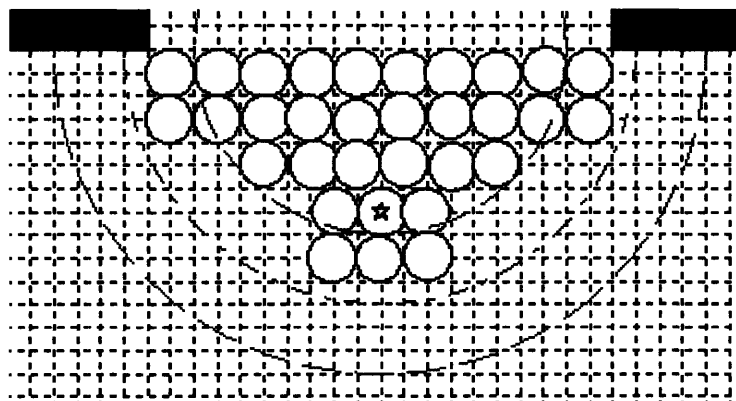
To address this concern, the algorithm in this thesis re-distributes the remaining UUVs after any assignment algorithm is complete. If three UUVs are still available, the UUVs reposition themselves to the expected center coordinates of sensors 21, 26, and 30. If only two UUVs are still available, the UUVs reposition themselves to the vicinity of sensors 27 and 31. If only one UUV is left available, it repositions itself to the expected coordinates of sensor 29. In every case of UUV re-positioning, the algorithm determines the assignment of UUVs to new locations that minimizes the total travel distance for the remaining UUV(s). The figures below show the assumed, re-distributed locations of the unassigned UUVs in the events of 3, 2, or 1 remaining UUV(s).



**Figure 4.3 – Assumed Coverage Positions with 3 UUVs Remaining**



**Figure 4.4 – Assumed Coverage Positions with 2 UUVs Remaining**



**Figure 4.5 – Assumed Coverage Position with only 1 UUV Remaining**

As before, the model assumes that these re-positioned locations are optimal or near optimal. The positions are assumed based on the requirement for full coverage of the sensor array, the known distributions of threat vehicle arrivals and movements, and working experience with the model. Chapter 7 will address this assumption by testing the optimality of the re-positioned UUV locations using dynamic programming.

#### ***4.9 Information Requirements at Each Time Step***

Except for the calculation of  $\tau$  which occurs only on the second visible time step, the algorithm follows the basic procedures described above during each visible time step of threat vehicle movement. The simulation stores a variety of data corresponding to the threat vessel's disposition during that time step. Specifically, it records the coordinates of the threat vehicle, the calculated interception point (at  $\tau$  time steps), the threat vehicle's calculated speed and bearing, and confidence interval data. The confidence interval data is important because it will motivate the possible need for UUV reassignment if the threat vessel has changed its speed or bearing. Reassignment could become necessary if the threat vessel's stochastic movements cause a different UUV to be more capable of successful interception after the initial assignment has already occurred. The simulation allows for such a reassignment, but the details of the algorithm require the use of confidence regions. Chapter 5 will discuss confidence regions and the reassignment criteria in greater detail.

#### ***4.10 Controlling the Assigned UUV***

The assumptions and explanations thus far have described a working algorithm for threat vehicle movement, a working algorithm for sensor / UUV interaction, and have assigned the optimal UUV to intercept the threat vessel given the initial point of desired interception. The next step is to direct the assigned UUV at each discrete update of the simulation in order to support the highest probability of intercepting the threat vessel.

### 4.10.1 UUV Speed Assumptions

Recall the earlier assumption that the UUVs have a maximum speed of 5 knots (2.57 meters per second). This assumption provides the primary challenge to the model since the threat vessels can travel much faster than the UUVs. Previously, the algorithm assumed that a UUV is either dormant or moving at 2.57 meters per second.

A natural question is to speculate if it is ever optimal for the UUV to travel at a speed less than its maximum speed while en route to the desired point of intersection.

The answer to this question is no, as logically demonstrated below:

- For any given discrete time step, there is a desired interception point  $(x_\tau, y_\tau)$ . This point corresponds to the expected value of the distributions that define threat vehicle movement, and the calculated value of  $\tau$ .
- The distributions that define the potential of the threat vehicle to change speed or bearing are symmetric about the expected value.
- Given the symmetry of the threat vehicle's movement distributions, the probability of changing bearing to the right is precisely equal to the probability of changing bearing the corresponding amount to the left. The same can be said of the probability of increasing or decreasing speed. This is true except for extreme values of the speed and bearing changes (that occur with very low probability) that might require truncation given the previous speed or bearing of the threat vehicle.
- The UUV is best able to respond to a change in the threat vehicle's speed or bearing if the UUV is located at the expected point of desired intersection.
- The threat vehicle may change speed or bearing at any given discrete update. Consequently, the UUV should move at its maximum speed to reach the point of optimal interception. This gives the UUV the most flexibility to respond to any changes in the threat vehicle's disposition.

The model also assumes that ocean currents or tides do not affect the UUV's speed while traveling to its destination coordinates. This is almost certainly not true in practice, but these factors are impossible to predict in a general model such as the one in this thesis, that is not catered to specific ocean hydrology. Later models tailored to a specific piece of terrain could employ these effects if needed.

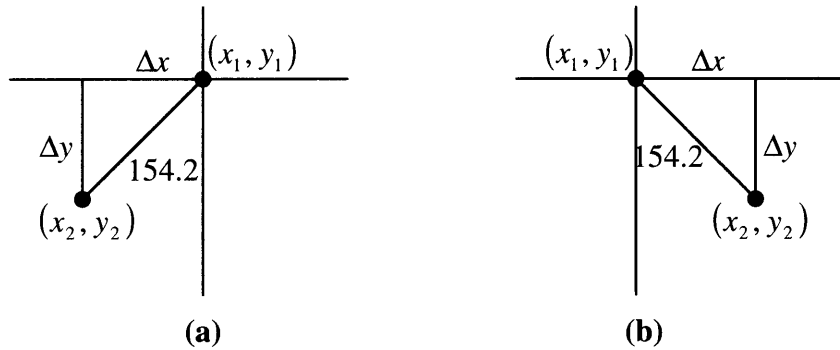
#### **4.10.2 Update the UUVs' Positions**

Calculation of the movement bearings for each of the UUVs in the base scenario assumes that there are no obstacles in the water. Chapter 8 will address a more complicated algorithmic structure using dynamic programming that could address this concern as an area for future research. The intent of this section is to update the UUVs' positions at each discrete time step of the simulation. The calculation determines the bearing that produces the shortest path from each of the UUVs' current locations to their desired coordinates (either  $(x_\tau, y_\tau)$  or the desired coverage position.) Given the known speed of the UUV, this also allows the model to calculate the position along this bearing that the UUV will have attained at the end of each discrete update. The discussion below describes the process of controlling the assigned UUV in its pursuit of the threat vessel. An analogous process controls the movements of the unassigned UUVs to their coverage positions.

Given the location of the assigned UUV  $(x_{UUV}, y_{UUV})$  and the desired intercept coordinates updated after each visible time step of threat vehicle movement  $(x_\tau, y_\tau)$ , the calculation must generate the optimal path between the two points. In the absence of obstacles, this corresponds to a straight line. The argument in Section 4.10.1 established that the UUV will travel at its maximum speed of 5 knots (2.57 meters per second). Consequently, the objective is to determine the coordinates that the UUV should achieve by the end of the next minute and then translate this information to the UUV in terms of a bearing. The methodology is below:

- Calculate the slope  $m$  between  $(x_{UUV}, y_{UUV})$  and  $(x_\tau, y_\tau)$ .

- Let  $\Delta x$  and  $\Delta y$  represent the distances the UUV will traverse relative to the  $x$  and  $y$  axes during the next minute (the next discrete time step).
- Given that the UUV will travel at 2.57 meters per second, it will travel 154.2 meters during the next minute. Consequently, the length of the hypotenuse is 154.2 meters. The figures below describe the two possible scenarios:



**Figure 4.6 – Movement of the Assigned UUV During 1 Time Step (1 Minute)**

- Solve the following system of equations for the values of  $\Delta x$  and  $\Delta y$  :

$$(\Delta x)^2 + (\Delta y)^2 = (154.2)^2 \quad (4.8)$$

$$\Delta y = m * \Delta x \quad (4.9)$$

- To attain positive magnitudes for  $\Delta x$  and  $\Delta y$  , multiply the values of  $\Delta x$  and  $\Delta y$  by  $(-1)$  as necessary so that  $\Delta x, \Delta y > 0$ .
- Depending on which of the graphs in Figure 4.6 (a or b) best describe the relationship between  $(x_1, y_1)$  and  $(x_2, y_2)$  , either add or subtract the magnitudes of  $\Delta x$  and  $\Delta y$  to the current coordinates of the assigned UUV. Determine whether to add or subtract the magnitudes of  $\Delta x$  and  $\Delta y$  by comparing the values of the coordinates  $(x_{UUV}, y_{UUV})$  and  $(x_\tau, y_\tau)$ .
- Given the point  $(x_2, y_2)$  that should be reached by the assigned UUV at the end of the discrete update, calculate the desired bearing for the UUV in the same way discussed in Section 4.5, using  $\tau$  time steps. This bearing is translated to the assigned UUV as control.



The assigned UUV will follow this methodology for each discrete update in the simulation. If the threat vehicle is within range of a sensor during the update, the algorithm calculates a new value of  $(x_\tau, y_\tau)$  that corresponds to the expected, updated data at  $\tau$  time steps. If the threat vehicle is not within the sensing radius of one of the sensors, the algorithm uses the last calculated value of  $(x_\tau, y_\tau)$ . This can occur if the threat vehicle has permanently left the area covered by the sensors, or if the threat vehicle is in one of the gaps in coverage between sensors. The interception algorithm also keeps track of the simulation time step number that corresponds to the last visible time step of the threat vessel. This will become important during the applied use of the confidence region data, as discussed in Section 5.2.5.

The assigned UUV will follow this algorithm until it reaches one of the following termination criteria:

- Successful interception of the threat vessel. Successful interception is defined as +/- 500 meters between the chasing UUV and the threat vessel.
- More than  $\tau$  time steps have occurred and both of the following are true:
  1. The simulation calculates that the assigned UUV cannot reach the current optimal interception coordinate  $(x_\tau, y_\tau)$  before the threat vehicle arrives there.
  2. The “best achievable point” by the assigned UUV fails to be within an acceptably good confidence region around  $(x_\tau, y_\tau)$ , where “acceptably good” is defined by the user. For termination purposes, the model defines “acceptably good” as being within the 10% confidence region around  $(x_\tau, y_\tau)$ . This makes it very unlikely that there is a situation where the simulation terminates prematurely when an actual interception could have happened. Conversely, if the UUV can still attain an “acceptably good” confidence region, the simulation continues even if there have been more than  $\tau$  time steps. In many such cases, this ultimately results in a successful interception of the threat vehicle.

The details of the termination criteria are addressed in the discussion of applied confidence intervals in Chapter 5.

## **Chapter 5 Confidence Intervals & the Reassignment Algorithm**

The algorithm as described in Chapters 2-4 is a fully-functional algorithm that successfully intercepts the threat vessel approximately 74% of the time. However, the description of the algorithm up to this point does not allow for reassignment of the chaser UUV after the initial assignment. The stochastics that govern threat vessel movement often cause the threat vessel to change its disposition in a way that makes a different UUV closer to the new point of optimal interception. In such a scenario, a robust algorithm should have a mechanism to consider reassignment of UUVs after the initial assignment has occurred.

Chapter 5 addresses this concern by developing a reassignment algorithm for the simulation. The reassignment algorithm relies on the applied use of confidence intervals. The first half of Chapter 5 addresses the theory of confidence intervals, and the second half of the chapter describes how to apply this theory to the simulation.

### ***5.1 Theory of Confidence Intervals***

The tools of statistics may be employed for two types of estimation: point estimation and interval estimation. Having already addressed the task of estimating a point of interception for the threat vessel using expected values, the focus of this research turns to interval estimation in the vicinity of the point  $(x_r, y_r)$ . The motivating question becomes: How well does the point estimate represent the underlying, stochastic distributions of threat vessel movement? Confidence regions address this question by providing a range of values which is likely to contain the parameter of interest (the interception coordinates) within a predetermined confidence level  $\psi$  [25].

Throughout this thesis,  $\psi$  will represent the confidence level of the desired confidence region. Strictly speaking,  $\psi = (1 - \alpha) * 100$ , where  $\alpha$  is the significance level

of the corresponding hypothesis test ( $0 \leq \alpha \leq 1$ ) [25]. A  $\psi$  %, two-sided interval estimate of a parameter  $\theta$  is an interval of the form  $\hat{\theta}_1 < \theta < \hat{\theta}_2$ , where  $\hat{\theta}_1$  and  $\hat{\theta}_2$  are values of appropriate random variables  $\hat{\Theta}_1$  and  $\hat{\Theta}_2$  such that  $P(\hat{\Theta}_1 < \theta < \hat{\Theta}_2) = 1 - \alpha = \frac{\psi}{100}$  [15].

For example, if  $\alpha = 0.10$ ,  $\psi = 90$  and  $\hat{\theta}_1$  and  $\hat{\theta}_2$  are the upper and lower confidence limits of the 90%, two-sided confidence interval.

### 5.1.1 Frequentist vs. Bayesian Perspectives of Confidence Intervals

The frequentist and Bayesian perspectives offer two competing schools of thought in theoretical statistics with different perspectives on the interpretations of confidence intervals. This thesis considers the theory behind both methodologies, and uses books by Casella and Berger[12], Cox and Hinkley [14], and Freund[15] as the primary references for statistical theory. Whether considering the frequentist or Bayesian perspectives, the objective is to calculate a  $\psi$  % confidence interval around the point  $(x_r, y_r)$ .

When approaching the problem from a frequentist perspective, a  $\psi$  % confidence interval does **not** mean that there is a  $\psi$  % probability that the interval contains the true population parameter  $\theta$ . The interval either includes the true value of the parameter or it does not. Consequently, it would be inappropriate to think in terms of post-data probabilities unless the number of observations ( $n$ ) approached infinity [19]. Instead, a  $\psi$  % confidence interval means that if the same population is sampled on numerous occasions (“ $n$ ” times) and interval estimates are made on each occasion, the resulting intervals would bracket the true population parameter (the actual interception coordinates) in approximately  $\psi$  % of the cases [25]. In one sense, as  $n \rightarrow \infty$  the interval covers the parameter with  $\psi$  % success.

The Bayesian interpretation allows a more useful application of confidence intervals in this simulation. By assuming a prior probability distribution, the expression

$P(\hat{\Theta}_1 < \theta < \hat{\Theta}_2) = \frac{\psi}{100}$  takes on a clearer form from which to make post-data probability

statements instead of references to the hypothetical repetition of the experiment  $n$  times as  $n \rightarrow \infty$  [14]. Under the Bayesian model, it is valid to assert that the parameter is inside the upper and lower confidence limits with probability  $\psi$  % because the parameters themselves (the interception coordinates) are treated as random variables whose “true” values may change with each repetition [12]. Since the simulation model assumes underlying distributions for both the initial disposition and the later movements of the threat vehicle, the employment of a prior distribution is natural. Since the parameters of interest (the  $x$  and  $y$  coordinates of  $(x_\tau, y_\tau)$ ) will change over time, it is also natural to think of each of them as random variables. The prior distribution offers insight into the coordinates  $(x_\tau, y_\tau)$  through the expected values of the corresponding prior distributions. Consequently, the Bayesian interpretation will be used throughout this research.

Under the Bayesian model, a random quantity  $\theta$  is best described by its posterior distribution:

$$Post(\theta | x) = \frac{f(x | \theta) * \pi(\theta)}{\int f(x | \hat{\theta}) * \pi(\hat{\theta}) d\hat{\theta}} \quad (5.1)$$

where  $\pi(\theta)$  and  $\pi(\hat{\theta})$  are the prior distributions and the functions  $f(x | \theta)$  and  $f(x | \hat{\theta})$  are the stochastic distributions that define the change over the next period of time [12]. Since the denominator is a constant, a simpler relationship for the posterior distribution is below.

$$Post(\theta | x) \propto f(x | \theta) * \pi(\theta) \quad (5.2)$$

### 5.1.2 Calculating a Confidence Interval

In general, a two-sided interval estimate of a real-valued parameter  $\theta$  is any pair of functions,  $L(x_1, \dots, x_n)$  and  $U(x_1, \dots, x_n)$ , of a sample of observations  $x$  that satisfy  $L(x) \leq U(x)$  for all  $x \in X$  [12]. For applied purposes, however, the objective becomes calculating a confidence interval with a specific confidence level  $\psi = (1 - \alpha) * 100$ .

The algorithm in this simulation will calculate confidence intervals from Gaussian distributions since the assumptions of threat vehicle stochastics come from the Gaussian distributions presented in Chapter 3. One Gaussian distribution describes potential changes in the threat vessel speed, and a second Gaussian distribution describes potential changes in threat vessel bearing. The two Gaussian distributions are assumed to be independent of each other, as discussed in Chapter 3. The interval estimate for each Gaussian distribution is of the form described below where the values of  $\pm z_{\frac{\alpha}{2}}$  come from the standard normal distribution based on the value of  $\alpha$ ,  $\bar{x}$  is the assumed mean,  $\sigma$  is the assumed standard deviation for the corresponding Gaussian random variables, and  $n$  is the number of time steps [26].

$$P\left(\bar{x} - z_{\frac{\alpha}{2}}\left(\frac{\sigma}{\sqrt{n}}\right) \leq \theta \leq \bar{x} + z_{\frac{\alpha}{2}}\left(\frac{\sigma}{\sqrt{n}}\right)\right) = 1 - \alpha = \frac{\psi}{100} \quad (5.3)$$

In other words, with probability  $1 - \alpha = \frac{\psi}{100}$ , the population mean  $\theta$  will lie within the region  $\bar{x} \pm z_{\frac{\alpha}{2}}\left(\frac{\sigma}{\sqrt{n}}\right)$  after hypothetically repeating the experiment  $n$  times as  $n \rightarrow \infty$ .

The corresponding  $\psi\%$  confidence interval is below:

$$\left[ \bar{x} - z_{\frac{\alpha}{2}}\left(\frac{\sigma}{\sqrt{n}}\right), \bar{x} + z_{\frac{\alpha}{2}}\left(\frac{\sigma}{\sqrt{n}}\right) \right] \quad (5.4)$$

As described earlier, the Bayesian methodology allows a more convenient form for claiming that the interval estimate contains the population mean with a desired probability without the hypothetical repetition of the experiment  $n$  times as  $n \rightarrow \infty$ . The  $\psi\%$  Bayesian interval estimate that is parallel to the frequentist interval estimate of Equation 5.4 is found in Equation 5.5 below [12].

$$\left[ \delta^B(\bar{x}) - z_{\frac{\alpha}{2}}\sqrt{\text{Var}(\theta | x)}, \delta^B(\bar{x}) + z_{\frac{\alpha}{2}}\sqrt{\text{Var}(\theta | x)} \right] \quad (5.5)$$

Equation 5.5 is the  $\psi\%$  Bayesian credible set, where  $\delta^B(\bar{x})$  is the Bayesian decision that corresponds to  $\bar{x}$  because of the impact of the prior distribution [12]. The full derivation

for this expression is in [12]. While the structure is clearly parallel to that of Equation 5.4, one important difference is that the calculation for the Bayesian confidence interval no longer directly depends on  $n$ . At each time step of the model, the prior distribution accounts for the information from the previous  $n$  time steps through the process of information updating. For each time step, however, the direct computation of the  $\psi\%$  confidence intervals occurs without respect to  $n$  because of the use of the prior distribution in calculating  $\delta^B(\bar{x})$  and  $VAR(\theta | x)$ .

### 5.1.3 Optimality of the Confidence Intervals

The calculation of a  $\psi\%$  confidence interval is not unique. In fact, there are many such confidence intervals that could be calculated with a confidence level of  $\psi\%$ . Using the Bayesian methodology, the optimal  $\psi\%$  confidence interval from the Bayesian credible set is the smallest confidence interval with a specific coverage probability. It is useful to define the Bayesian credible set through a series of regions of high posterior density [14]. Let  $\theta$  be the parameter of interest (the coordinates  $(x_\tau, y_\tau)$  influenced by possible changes in speed or bearing). If  $p(\theta)$  is the posterior distribution of  $\theta$  given the previous observations  $x$ , and  $\psi = (1 - \alpha) * 100$ , the objective is to find the confidence interval  $C(x)$  that satisfies (i) and (ii) below

$$(i) \int_{C(x)} p(\theta) d\theta = \frac{\psi}{100} \quad (5.6)$$

$$(ii) Size(C(x)) \leq Size(C'(x)) \quad (5.7)$$

for any other interval  $C'(x)$  that also satisfies

$$\int_{C'(x)} p(\theta) d\theta = \frac{\psi}{100} \quad (5.8)$$

as shown in [12]. Intuitively, the objective is to find the confidence interval  $C(x)$  with the desired coverage probability ( $\psi\%$ ) whose size is smaller than any other confidence

interval  $C'(x)$  with the same coverage probability. This general definition is most relevant for multimodal distributions or discontinuous distributions. In this research, the distributions of concern for threat vehicle stochastics are both independent, unimodal Gaussian distributions, and the  $Size(C(x))$  is the length of the confidence interval between the two confidence limits. In this case, the optimal (shortest)  $\psi$  % confidence interval for  $\theta$  is the  $\psi$  % confidence interval centered at the maximum point of the bell-curve, the expected point of desired interception.

## **5.2 Theory Applied to the Simulation**

This simulation makes extensive use of confidence intervals. Since the threat vehicle's movements are described by probabilistic distributions, it is impossible to speak in terms of known interception locations. Instead, the algorithm uses the expected values of the movement distributions to define a point of expected interception. Similarly, the model uses confidence regions to describe the probability that the threat vehicle is within some threshold probability deviation from the point of expected interception.

### **5.2.1 Use of Confidence Intervals**

In this simulation, the initial location, bearing, and speed of the threat vehicle are defined in the first time step. From a Bayesian, theoretical perspective, the known initial disposition of the threat vehicle becomes the prior distribution. After the threat vehicle's location and disposition are known (detected and calculated by the sensor system), the threat vehicle's future movements are described by possible changes in bearing and/or speed. By assumption, both of these distributions are Gaussian, with means corresponding to the current speed and bearing, and variances as defined earlier.

Using the Bayesian conceptual framework, the  $x$  and  $y$  values of the coordinate  $(x_t, y_t)$  are defined through their posterior distributions as shown in Equations 5.1 and 5.2. The posterior distribution is proportional to the known disposition of the

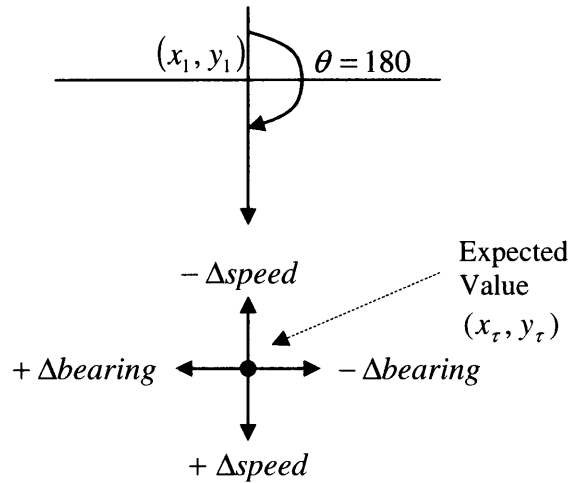


threat (the prior distribution) multiplied by the pdf of the Gaussian distribution that defines the change in disposition during the next discrete time step. At the end of each time-step, the newly-realized disposition of the threat vessel becomes the prior distribution for the following time-step, a process known as information updating [19]. This process repeats itself until the simulation terminates. This conceptual framework is used in the sections below.

### **5.2.2 Confidence Intervals Defining Scoring Regions**

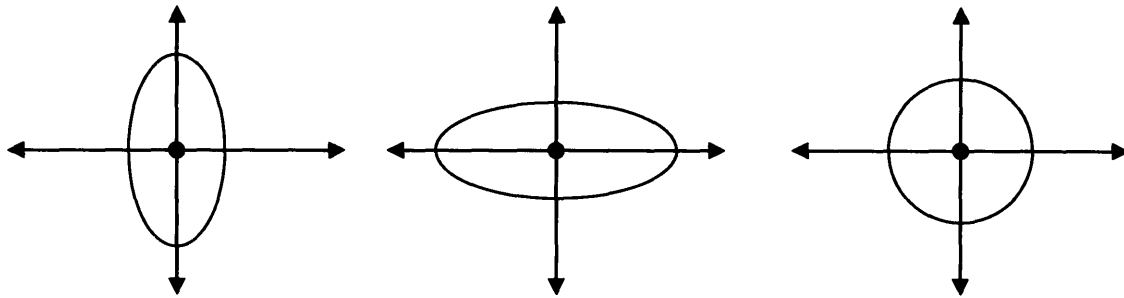
The process of information updating in this research is actually concerned with two independent confidence intervals. One confidence interval addresses the Gaussian distribution that describes the possible changes in threat vessel speed, and the second confidence interval addresses the Gaussian distribution that describes the possible changes in threat vessel bearing. The span of these two, independent confidence intervals describes a confidence region. As shown above, the algorithm combines the joint effects of the stochastic movement distributions during each time step with the “new” prior distribution to calculate the updated posterior distribution. The algorithm in this simulation uses the ranges of the two, independent  $\psi$  % confidence intervals around  $(x_\tau, y_\tau)$  to define a  $\psi$  % scoring region.

Imagine a scenario like in Figure 5.1 below, where the threat vessel is traveling at precisely 180 degrees. In this scenario, one Gaussian distribution affects the possible changes in the  $y$  coordinate (the speed distribution) and the second, independent Gaussian distribution affects the possible changes in the  $x$  coordinate (the bearing distribution).



**Figure 5.1 – Conceptual Figure for Scoring Region Construction**

This simulation uses the two, independent confidence intervals around the expected point of interception as a scoring metric to determine whether or not reassignment of UUVs should be considered. For any given  $\psi$  % confidence level, the simulation calculates the endpoints of the two confidence intervals centered at  $(x_\tau, y_\tau)$ . In the example above, the confidence interval for speed affects only the  $y$  coordinates while the confidence interval for bearing affects only the  $x$  coordinates. For any given confidence level of  $\psi$  %, the algorithm calculates the endpoints of the confidence intervals for both distributions (both centered at  $(x_\tau, y_\tau)$ ). These four points define the endpoints of the  $\psi$  % scoring region about  $(x_\tau, y_\tau)$ . This  $\psi$  % scoring region will be symmetric about both axes and approximately elliptical in shape. The variances of the two distributions will determine which of the axes are the major and minor axes of the ellipse. Three possibilities are shown in Figure 5.2 below.



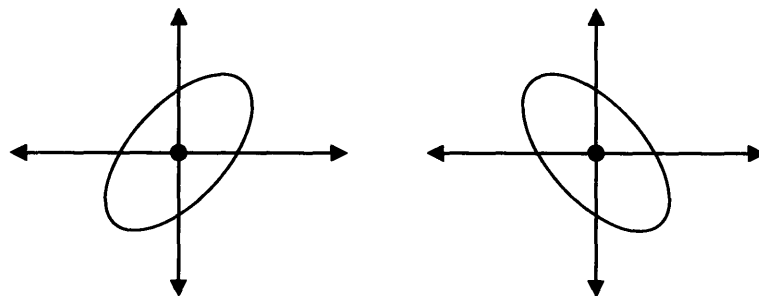
**Figure 5.2 – Possible shapes for the Scoring Regions about  $(x_r, y_r)$**

Given the assumed range of threat vessel bearings in this research, these drawings are only relevant if the threat vessel's bearing is exactly 180 degrees, 90 degrees, or 270 degrees. In this case, the model would use the familiar equation of an ellipse:

$$\frac{(x - x_t)^2}{a^2} + \frac{(y - y_t)^2}{b^2} = 1 \quad (5.9)$$

The center of the ellipse is the coordinate  $(x_r, y_r)$ , the length of the major axis is  $2*a$ , and the length of the minor axis is  $2*b$ , if  $b < a$ .

For all other possible bearings, the algorithm must rotate the ellipse-shaped scoring region through standard transformation equations. Even though the ellipse is rotated through transformation, the ellipse is still centered at  $(x_r, y_r)$ , and it remains symmetric about the major and minor axes. Figure 5.3 below shows two such possibilities:

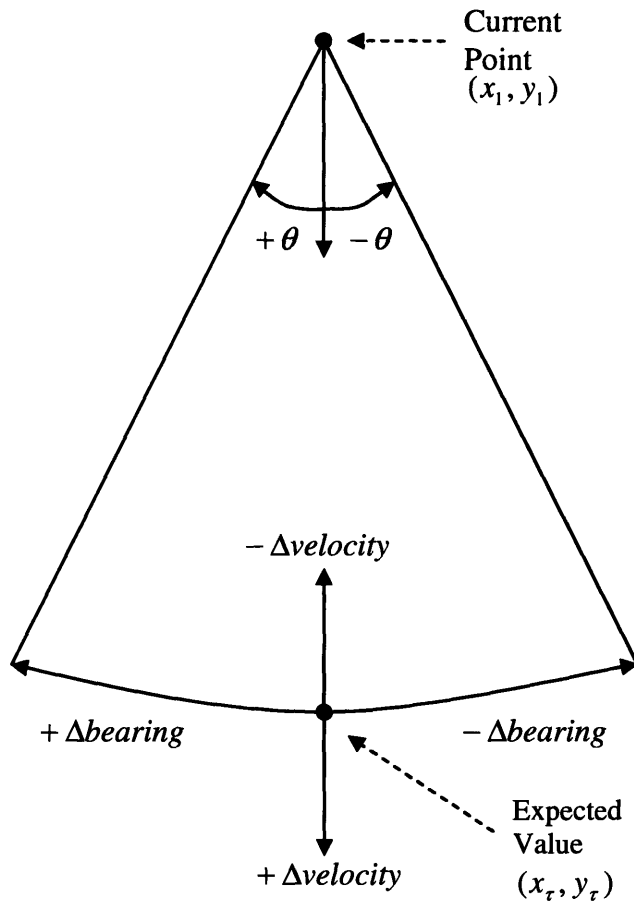


**Figure 5.3 – Two Possibilities of Rotated Scoring Regions**

Using this methodology, the algorithm can approximate the desired  $\psi$  % scoring region by drawing the corresponding ellipse that has the extreme values of the two  $\psi$  % confidence intervals as the endpoints of the major and minor axes.

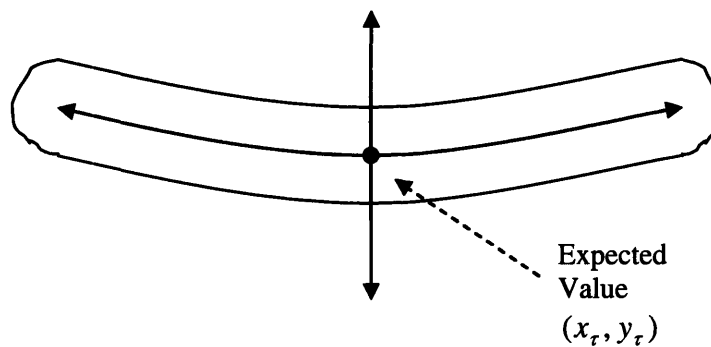
### 5.2.3 “Bent Ellipse” Shape

The discussion in the previous section is valid only if an ellipse is a legitimate approximation to the desired  $\psi$  % scoring region. For an ellipse approximation to be valid, both the major and minor axes of the scoring region must be line segments. In actuality, the confidence interval that describes the probability of a change in bearing lies along an arc (not a line segment). Figure 5.4 below clarifies this phenomenon, where the angles  $\pm\theta$  represent the angles of deviation from the current bearing as calculated to correspond to the desired  $\psi$  % scoring region.



**Figure 5.4 – Depiction of the Bearing Confidence Interval as an Arc**

Because of this phenomenon, the  $\psi$  % scoring region is not strictly described by an ellipse. Instead, the region would appear as a somewhat bent ellipse. Figure 5.5 exaggerates the relative length of the major axis to demonstrate this phenomenon, where the curved shape drawn around the arc represents some  $\psi$  % scoring region.



**Figure 5.5 – Approximation of the Bent Ellipse Shape for Scoring Regions**

#### **5.2.4 Effects of the “Bent Ellipse” Phenomenon**

As seen in the figures above, the “bent ellipse phenomenon” is most noticeable when the length of the confidence interval corresponding to bearing changes is much larger than the length of the confidence interval corresponding to changes in speed given the desired  $\psi$  % scoring region. In the simulation output, the opposite is true. Secondly, the lengths of the  $\psi$  % confidence intervals (for both distributions) will become shorter at each time step of the simulation. This occurs as the algorithm moves closer in time to the originally calculated, desired interception time  $\tau$ . As the length of the  $\Delta$  bearing confidence interval becomes shorter, the phenomenon becomes less noticeable. Consequently, as the importance of the scoring regions to the success of the simulation increases (as the simulation approaches the optimal interception point), the lengths of the corresponding confidence intervals decrease and the “bent ellipse phenomenon” becomes less visible. Although trivial, this phenomenon is noticeable in the simulation output as shown in Section 5.2.5 below.

#### **5.2.5 Application of Scoring Regions in the Simulation**

Just as the *randn* function assisted in the use of random numbers in MATLAB, the *norminv* function in the statistics toolbox of MATLAB assists in the calculation of

confidence intervals. The *norminv* function calculates the inverse of the normal (Gaussian) cumulative distribution function (CDF) given inputs of the mean, standard deviation, and the desired confidence level ( $\psi$  %) for the confidence interval [31]. This is particularly useful in this simulation since both of the distributions for potential changes in threat vehicle movement are Gaussian.

Section 4.9 discussed the stored information requirements for each visible time step of threat vehicle activity during the simulation. Much of this information is the confidence interval data as discussed below.

Once the algorithm calculates the number ( $\tau$ ) of time steps before desired interception, it can calculate the coordinates for the point of desired interception ( $x_\tau, y_\tau$ ) at each discrete update. For each visible update of threat vehicle movement, the algorithm calculates the  $\psi$  % confidence intervals around ( $x_\tau, y_\tau$ ) for both distributions where  $\psi$  corresponds to the 10, 20, 30, 40, 50, 60, 70, 80, and 90 percent confidence intervals. In particular, it calculates the endpoints of the major and minor axes of the corresponding  $\psi$  % “bent” ellipses for each of the confidence levels listed above, for each visible update of threat vehicle movement.

This methodology does not directly consider the independent Poisson arrival probabilities that a change of speed or bearing occurs at all. As such, the model used in this simulation is a worst case scenario that allows for changes in both distributions at every time step. The result is therefore overly conservative and models the most difficult possibility in the pursuit algorithm. Any model that works under these extreme circumstances will be even more effective in real-world scenarios where changes in the speed and bearing of the threat vessel are much less frequent.

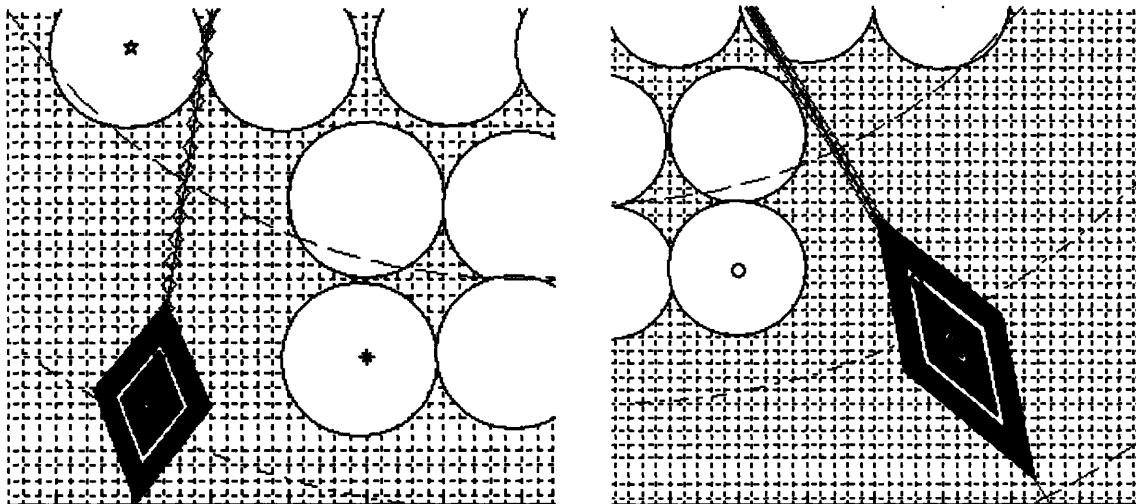
## 5.2.6 Approximating the Scoring Regions in the Simulation

For any given  $\psi$  % confidence level, the algorithm uses the *norminv* function described above to calculate the changes in bearing ( $\pm \theta$  degrees) that correspond to the desired  $\psi$  % confidence interval centered at  $(x_\tau, y_\tau)$ , given the assumed distribution for changes in bearing. This process was previously illustrated in Figure 5.4. The algorithm then projects the bearing changes  $(\tau - k)$  time steps to determine the endpoints of the  $\psi$  % bearing confidence interval around  $(x_\tau, y_\tau)$ . The algorithm follows an analogous process to calculate the endpoints of the speed confidence interval centered at  $(x_\tau, y_\tau)$ , using the assumed distribution for changes in speed.

Using this methodology, the algorithm stores the four critical coordinates of the  $(10 * \phi)$  percent confidence intervals around  $(x_\tau, y_\tau)$ , where  $\phi = 1.9$ . Since the scoring regions cannot be accurately described as true ellipses, the algorithm approximates the “bent ellipse” shape of the scoring region by a four-sided polygon that connects the four critical coordinates for each  $(10 * \phi)$  percent confidence interval.

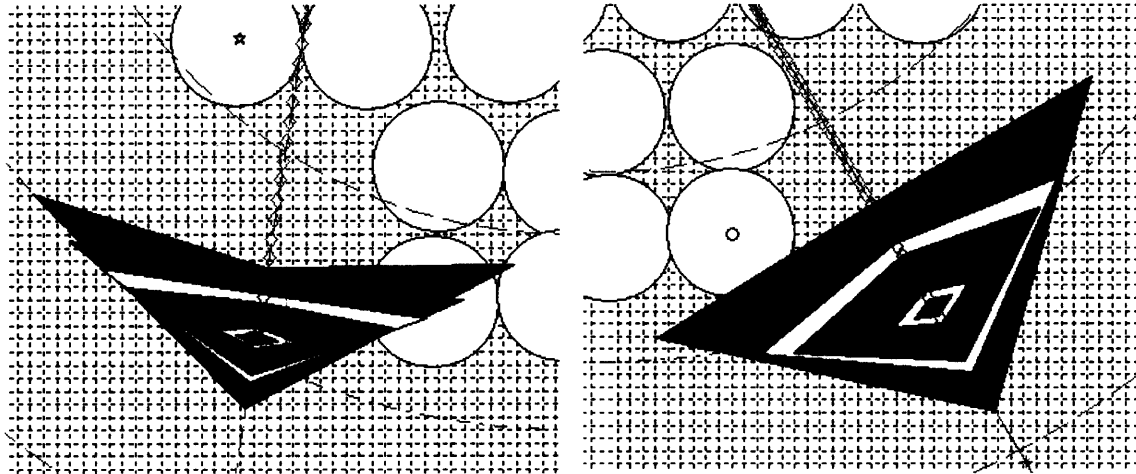
Two visual examples of the resulting scoring regions are in Figure 5.6 below, where the scoring regions are drawn corresponding to the last threat vessel time step that was visible by the sensor system. The figures below are enlarged to emphasize the appearance of the scoring regions and the small diamonds show the actual path of the threat vehicle’s movement. Although the effect is subtle, note that the “bent ellipse” phenomenon along the minor axis of the scoring regions is still noticeable.





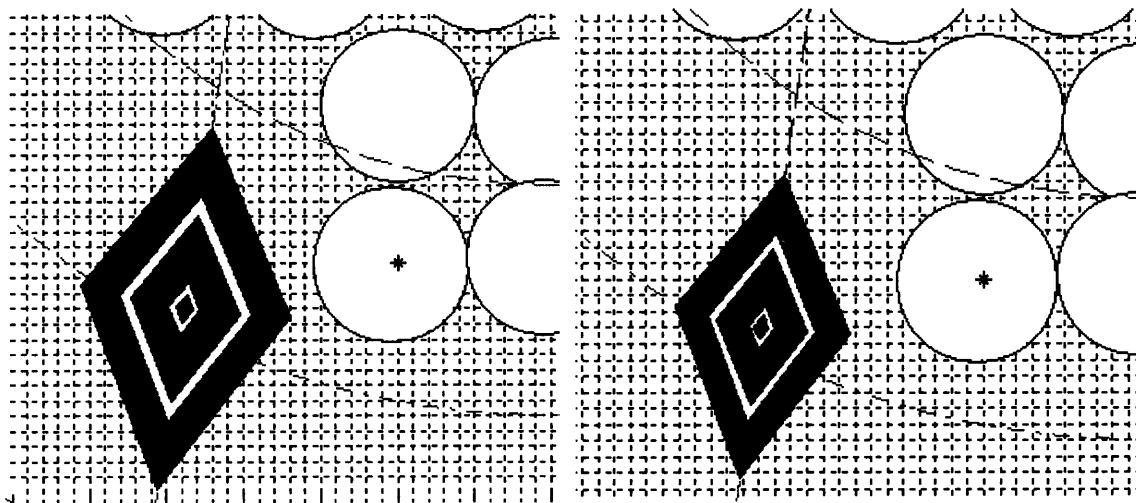
**Figure 5.6 – Two Visual Examples of the Scoring Regions Around  $(x_r, y_r)$**

As the variance of the Gaussian distribution that defines the probability of changing bearing increases, the corresponding bearing confidence intervals are much wider. This point is mentioned to illustrate the increased appearance of the “bent ellipse” effect as the variance of the  $\Delta bearing$  distribution increases. Figure 5.7 below demonstrates this trend by showing the same two scoring regions as in Figure 5.6 above, but with the variance of the  $\Delta bearing$  distribution squared. Note that the four-sided approximations to the ellipses change form for the higher percentage scoring regions. The lower percentage scoring regions remain diamond-shaped and the “bent ellipse” phenomenon is less pronounced.



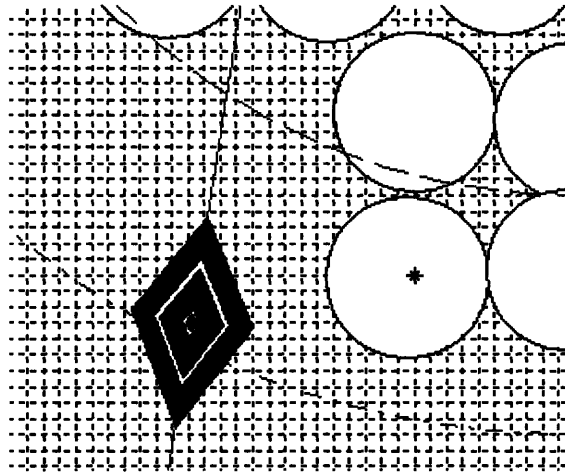
**Figure 5.7 – Wider Scoring Regions to Accentuate the “Bent Ellipse” Effect**

As the simulation progresses, the sizes of the confidence intervals decrease as the Bayesian interval estimate “converges” about the expected point of desired interception. Figures 5.8 (a-c) below show the approximated scoring regions about the expected point of desired interception from the same simulation run, calculated at a progressively larger number of time steps. Notice that the scoring regions decrease in size as the simulation matures.



**(a) – After 4 Time Steps**

**(b) – After 10 Time Steps**



(c) – After 17 Time Steps

Figure 5.8 – Progressively Smaller Scoring Regions as the Number of Time Steps Increases

### 5.3 Using the Scoring Region Data

The algorithm uses the scoring region data to provide a quantitative measure of the likelihood of the assigned UUV intercepting the threat vessel. Before moving the assigned UUV at each discrete step, the algorithm calculates the distance between the assigned UUV's location and the expected point of desired interception  $(x_\tau, y_\tau)$ , calling this distance  $\delta_{required}$ . Since the model assumes that the UUV moves at exactly 5 knots and the time step number ( $k$ ) of the current time step is known, the algorithm can calculate the maximum possible distance  $\delta_{max}$  that the UUV can travel in the remaining  $(\tau - k)$  time steps. If  $\delta_{max} \geq \delta_{required}$ , the algorithm returns a message indicating that the assigned UUV is able to intercept the threat vessel given its current disposition.

If  $\delta_{max} < \delta_{required}$ , the simulation follows the methodology below:

- Calculate the bearing for the assigned UUV according to the procedure discussed in Section 4.5.

- Project the assigned UUV ( $\tau - k$ ) time steps along the optimal bearing and determine the closest achievable point  $(x_{achievable}, y_{achievable})$  to the expected point of desired interception  $(x_{\tau}, y_{\tau})$ .
- Determine which (if any) of the  $(10 * \phi)$  percent scoring regions around  $(x_t, y_t)$  contain the point  $(x_{achievable}, y_{achievable})$ ,  $\phi = 1..9$ . The simulation uses the MATLAB function *inpolygon* to test whether or not  $(x_{achievable}, y_{achievable})$  is within each of the nine polygons defined by the four critical points for the nine,  $(10 * \phi)$  percent scoring regions around  $(x_t, y_t)$ ,  $\phi = 1..9$ .
- Determine the smallest sized scoring region (if any) that contains  $(x_{achievable}, y_{achievable})$ . Note that this refers to the scoring region with the smallest size (area) that contains  $(x_{achievable}, y_{achievable})$ . Assign a value score to the point  $(x_{achievable}, y_{achievable})$  according to the table below.

Smallest Scoring Region Containing $(x_{achievable}, y_{achievable})$	Value
None	0
90 %	1
80 %	2
70 %	3
60 %	4
50 %	5
40 %	6
30 %	7
20 %	8
10 %	9

**Table 5.1 – Proximity Scores for UUVs**

- The algorithm returns a message indicating the achievable value that the assigned UUV can attain given the current threat vehicle disposition.

In either case ( $\delta_{\max} \geq \delta_{\text{required}}$  or  $\delta_{\max} < \delta_{\text{required}}$ ), the algorithm uses the output to determine whether or not to consider reassigning a different UUV to the threat vessel. This issue is discussed in greater detail in the next section.

## **5.4 Reassignment of UUVs**

As mentioned earlier in the section on UUV assumptions, this algorithm assumes the presence of only four UUVs. Furthermore, the UUVs have limited energy stores, and they must leave the sensor system to recharge once their energy is depleted. Consequently, the algorithm should avoid reassigning UUVs if the previously assigned UUV has a “reasonably good” chance of intercepting the threat vessel. The challenge is to quantify what it means to have a “reasonably good chance.”

### **5.4.1 When Should the Algorithm Consider Reassignment**

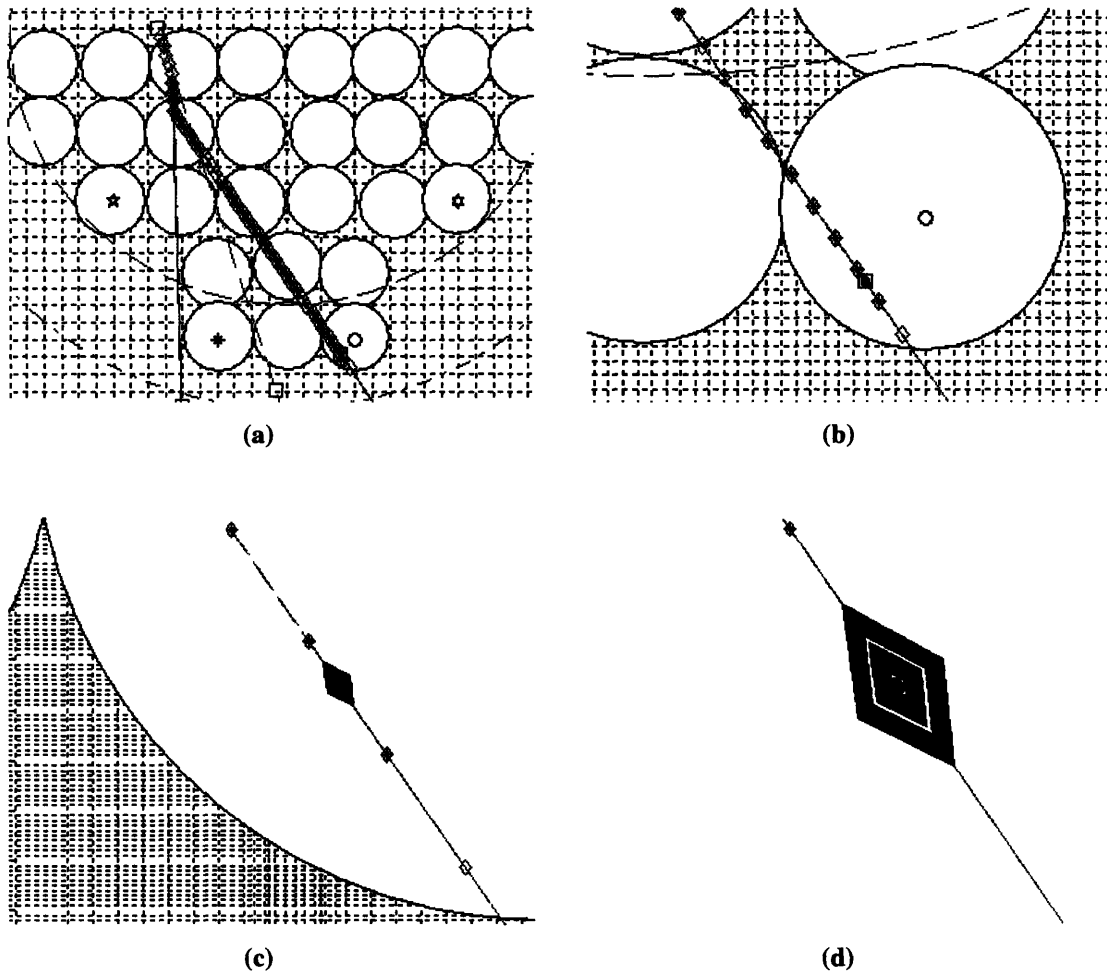
The definition of a successful interception in this simulation is when the UUV is within 500 meters of the threat vehicle. Consequently, the task becomes determining which of the scoring regions (and corresponding value score) most consistently corresponds to approximately 500 meters away from the expected coordinates of optimal interception. The output and graphs from 100 simulation runs were considered to gather data on this subject. For each run of the simulation, the algorithm displayed the scoring regions that corresponded to the last visible time step of threat vehicle movement. It then calculated the distance from the expected interception point to the coordinates defining each of the nine ( $10 * \phi$ ) percent confidence intervals around  $(x_{\tau}, y_{\tau})$ ,  $\phi = 1..9$  for both the bearing and speed confidence intervals. For each run of the simulation, the system then determined which confidence interval (in each of the two directions) was the closest to 500 meters in length without exceeding 500 meters. The results are in Table 5.2 below:

Confidence Level	Results for the Speed Confidence Interval	Results for the Bearing Confidence Interval
90%	32%	63%
80%	7%	14%
70%	3%	18%
60%	15%	4%
50%	14%	0%
40%	17%	0%
30%	9%	0%
20%	2%	1%
10%	1%	0%

**Table 5.2 – Which Confidence Interval is closest to 500 m from  $(x_\tau, y_\tau)$  without Exceeding 500 m**

The first important observation from this data is that the confidence intervals tend to be longer in the  $\pm$  *velocity* direction and shorter in the  $\pm$  *bearing* direction. This is evident in the data above because almost two-thirds of the 90% confidence intervals for bearing were less than 500 meters in length from the expected coordinates to the endpoint of the confidence interval. However, the confidence intervals for speed that corresponded to 500 meters in length from the expected coordinates to the confidence interval endpoints were much more spread out across the spectrum of possibilities.

In one important sense, however, the data in Table 5.2 is misleading. As discussed earlier, the model determines the value of  $\tau$  based on the interception arc, and this value of  $\tau$  is used to calculate  $(x_\tau, y_\tau)$ . However, the threat vehicle often changes bearing, speed, or both bearing and speed during the course of its movement across the sensor array. As the algorithm continuously updates the coordinates of  $(x_\tau, y_\tau)$  at each visible time step, this point often moves extremely close to a sensor or even within the sensing radius of one of the sensors. When this happens, the model still displays the scoring regions for the last visible time step (even if this corresponds to the time step of successful interception). Consequently, these scoring regions are unrealistically small. Figures 5.9 (a-d) below show an example of this phenomenon.



**Figure 5.9 – Very Small Scoring Regions when  $(x_t, y_t)$  is Inside a Sensor Radius**

As seen in Figures 5.9 (a-d), the resulting confidence intervals can be tiny (often as small as 10-15 meters in length for the 90% confidence intervals). In these situations, the 90% confidence intervals for both the speed and bearing directions are clearly within 500 meters. These were also easy interceptions for the chasing UUV(s) because the threat vehicle was caught before it even left the sensor array system. Consequently, it would be inappropriate to use this data in determining the trends for which confidence intervals typically represent 500 meters from the expected point of interception for the “difficult” chase scenarios that leave the sensing system. Considering this data would unrealistically credit the “goodness” of the 90% confidence intervals for reassignment purposes.

In the experiment of 100 simulation runs, 31% of the runs resulted in situations like the one above where the point of interception was ultimately inside a sensing region or extremely close to a sensing region (within 500 meters). Discarding these runs as representing unrealistically small confidence intervals, the remaining results are in Table 5.3 below.

Confidence Level	Results for the Speed Confidence Interval	Percentage	Results for the Bearing Confidence Interval	Percentage
90%	1	1.45%	32	46.38%
80%	7	10.14%	14	20.29%
70%	3	4.35%	18	26.09%
60%	15	21.74%	4	5.80%
50%	14	20.29%	0	0%
40%	17	24.64%	0	0%
30%	9	13.04%	0	0%
20%	2	2.90%	1	1.45%
10%	1	1.45%	0	0%

**Table 5.3 – Modified Data, After Eliminating Cases with Unrealistically Small Confidence Intervals**

This data represents the set of runs corresponding to the “difficult” scenarios where the point of ultimate interception was outside of the sensing array system by at least 500 meters.

Since the confidence intervals for speed still tend to be smaller than the confidence intervals for bearing, the analysis only considers the speed confidence intervals. The majority of these observations lie in the 40, 50, or 60 % confidence intervals. Consequently, the algorithm uses the 60% confidence interval (corresponding to a value score of 4) as the threshold. This means that any time step whose calculated value of  $(x_{achievable}, y_{achievable})$  is at least as close as the 60% scoring region around  $(x_{\tau}, y_{\tau})$  will not allow consideration of UUV reassignment.

Recall that the objective is to quantify the level at which the assigned UUV has a “reasonably good” chance of intercepting the threat vessel, given that the UUV cannot reach the expected point of optimal interception before the threat vessel arrives. Based on the explanation above, the model assumes that the assigned UUV has a “reasonably



good” chance to intercept the threat vessel if the point  $(x_{achievable}, y_{achievable})$  has a value score of 4 or greater, corresponding to the 60% scoring region around  $(x_\tau, y_\tau)$  or better.

### 5.4.2 3-Part Test for Reassignment

Based on the stochastic nature of threat vehicle movement, reassignment of UUVs may be required in some instances. Because the algorithm attempts to preserve the energy in the UUVs’ batteries as much as possible, it requires that the simulation pass a 3-part test before allowing consideration of re-assignment of UUVs.

- The simulation must be mature (at least 5 time steps). Early stages of the simulation often included some instabilities based on the calculation of  $\tau$  and the initial UUV assignment. By the fifth time step, the simulation was always stable.
- The initially assigned UUV must have  $\delta_{max} < \delta_{required}$ . (i.e. The UUV cannot reach  $(x_\tau, y_\tau)$  before the threat vessel gets there.)
- The value score of  $(x_{achievable}, y_{achievable})$  must be less than 4, meaning that the point  $(x_{achievable}, y_{achievable})$  is further away from  $(x_\tau, y_\tau)$  than the extreme limits of the 60% scoring region around  $(x_\tau, y_\tau)$ .

Once the system passes the 3-part test, the algorithm may *consider* reassignment. The word *consider* is stressed because it may be the case that the originally assigned UUV is still the best choice, even though its probability of successfully intercepting the threat vessel is very low.

The re-assignment algorithm operates in the same manner as the original assignment algorithm, by determining which of the UUVs is closest to the updated point  $(x_\tau, y_\tau)$ . If the new “optimal UUV” is different from the previously assigned UUV, the

previously assigned UUV joins the other remaining, unassigned UUVs in shifting to the “coverage positions” as discussed in Chapter 4.

### 5.4.3 Interception Algorithm / Reassignment Example

This section clarifies the interception algorithm and the reassignment algorithm through a detailed example. Figure 5.10 on page 84 of this document follows the narrative and provides a visual representation of the successful interception that includes a reassignment of the chaser UUV.

Step 1.) In this example, the threat vessel starts from a position relatively centered across the naval chokepoint. Its initial speed is 4.0543 m/s, and its initial bearing is 149.3021 degrees. After the second visible time step, the algorithm calculates  $\tau$  and  $(x_\tau, y_\tau)$ , and the algorithm assigns UUV number 4 (starting from the vicinity of sensor number 32).

Step 2.) The algorithm determines that  $\delta_{\max} \geq \delta_{\text{required}}$ , and UUV 4 begins moving towards  $(x_t, y_t)$ . Notice that UUV 3 begins moving to the “coverage position” in the vicinity of sensor number 31 after UUV 4 is assigned the mission.

Step 3.) Using the assumptions for stochastic movement and random numbers discussed earlier, the threat vessel changes speed during time step number 4 (4.6965 m/s) and again on time step number 5 (3.9213 m/s). Although this is statistically unlikely, the model reacts by calculating new values of  $(x_\tau, y_\tau)$  after each time step. Throughout this process,  $\delta_{\max} \geq \delta_{\text{required}}$ .

Step 4.) During time step number 8, the threat vessel changes bearing to 127.9810 degrees. The model re-calculates  $(x_\tau, y_\tau)$ , and determines that  $\delta_{\max} < \delta_{\text{required}}$  for UUV number 4. Furthermore, the model calculates the value score as 0, meaning that UUV 4 would not be able to achieve any of the scoring regions around the new  $(x_\tau, y_\tau)$ . (i.e. The point  $(x_{\text{achievable}}, y_{\text{achievable}})$  fails to reach even the 90% scoring region around the newly calculated  $(x_\tau, y_\tau)$ .)

Step 5.) The model passes the 3-part test for consideration of UUV re-assignment, and the algorithm determines that UUV 2 is now the optimal UUV.

Step 6.) UUV 2 begins movement towards  $(x_\tau, y_\tau)$ , and UUV 4 begins movement to the location where UUV 2 started (this is the “coverage position” for UUV 4 under the new scenario with UUV 2 as the assigned UUV).

Step 7.) In time step 27, the threat vessel changes bearing again to 92.8960 degrees. The algorithm recalculates  $(x_\tau, y_\tau)$  and determines that  $\delta_{\max} < \delta_{\text{required}}$  for UUV 2, but the value score for the chasing UUV is 4. This means that UUV 2 can achieve the 60% scoring region around the new  $(x_\tau, y_\tau)$  with the current value of  $(x_{\text{achievable}}, y_{\text{achievable}})$ . Since the value score is greater than 3, the algorithm does not consider reassignment.

Step 8.) As the number of time steps approaches  $\tau$ , the confidence intervals around  $(x_\tau, y_\tau)$  decrease in size. Consequently, the value score for  $(x_{\text{achievable}}, y_{\text{achievable}})$  drops to 3 in time step number 32. This means that the achievable point is within the 70% scoring region, and allows for the potential of UUV re-assignment. However, the reassignment algorithm determines that UUV 2 is still the best UUV for the mission even though its probability of success has dropped below the desirable threshold.

Step 9.) As the confidence intervals continue to decrease in size, the value score for  $(x_{\text{achievable}}, y_{\text{achievable}})$  drops to 2 during time step number 35 (the 80% scoring region). The model considers reassignment, but determines that UUV 2 is still the best choice. UUV 2 continues towards  $(x_\tau, y_\tau)$ .

Step 10.) In time step 45, UUV 2 successfully “intercepts” the threat vessel by attaining a position within 288 meters of the threat vessel. Coincidentally, the threat vessel also changes bearing to 124.8591 degrees during time step number 45. This is helpful for UUV 2 because it steers the threat vessel closer to the chasing UUV.

Step 11.) UUV 2 again “intercepts” the threat vehicle in time step number 46 by achieving a position within 41 meters of the threat vessel. The algorithm successfully terminates.

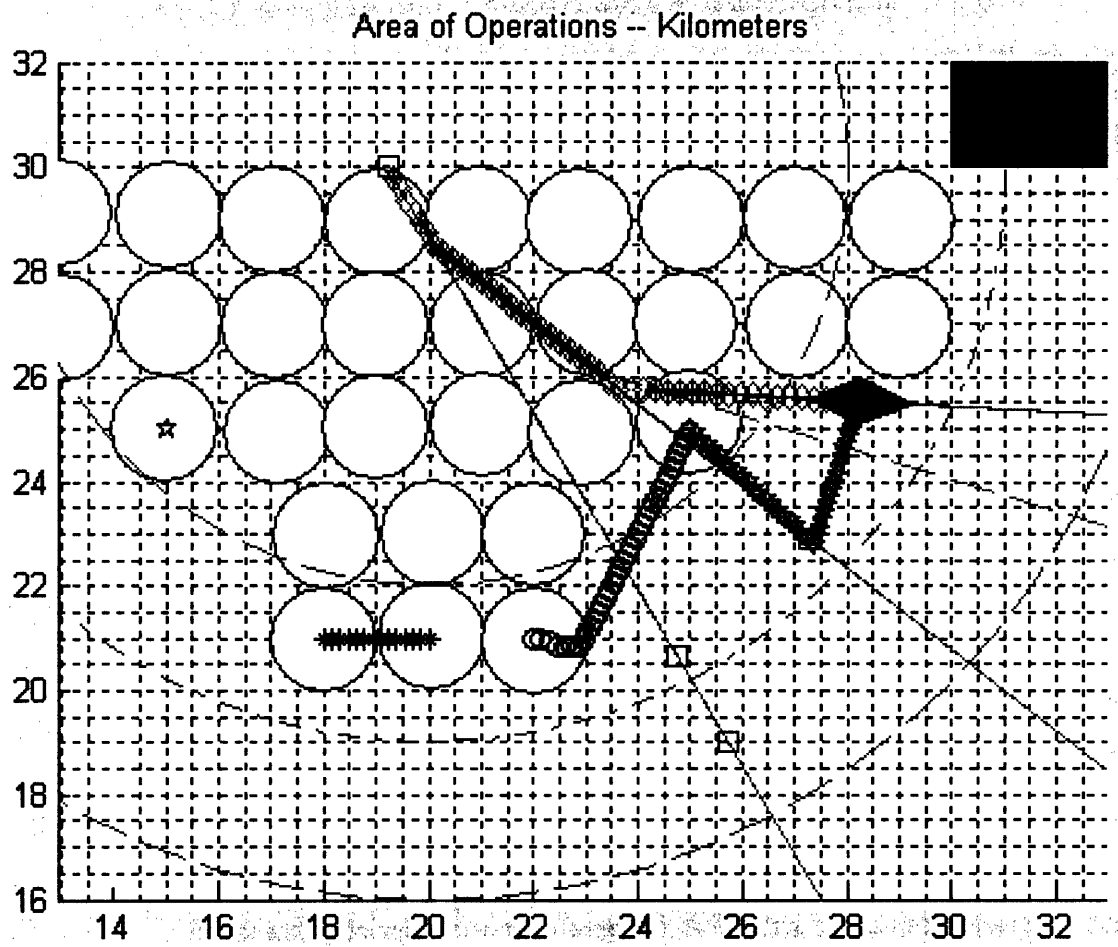


Figure 5.10 – Example of Successful Reassignment

## Chapter 6 Optimizing the Radius of the Interception Arc

The reassignment algorithm from Chapter 5 increased the successful interception rate from about 74% to about 86%. However, the algorithm still has two important parameters with assumed values – the radius of the arc of interception and the initial placement locations of the UUVs. By optimizing these assumed values, the algorithm should be able to achieve even higher success rates. The focus of Chapter 6 is to develop an optimization model that determines the optimal radius for the interception arc in the absence of human guidance. Chapter 7 will address the optimization problem for initial UUV placement.

### ***6.1 Motivation to Optimize the Radius of the Interception Arc***

One of the best ways to motivate future optimization of the algorithm is to analyze the simulation runs that result in failure. Upon consideration of these runs, it becomes apparent that approximately one-third of the unsuccessful iterations of the base-case scenario occurred when the threat vessel changed its disposition (speed or bearing) after leaving the sensor system for the last time.

In this type of scenario, the assigned UUV has a very high probability of missing the threat vessel depending on the time that the speed or bearing change occurs relative to the value of  $\tau$ . In every known example of this scenario, the assigned UUV would have successfully intercepted the threat vessel if the threat vessel had not changed bearing after leaving the sensor system for the last time. The explanation for this phenomenon is clear, as shown below.

The algorithm calculates  $(x_\tau, y_\tau)$  based on the last visible time step. Consequently, the algorithm also bases its final set of confidence regions on the last visible time step. When the threat vessel changes speed or bearing after leaving the sensor system for the last time, the model has no ability to respond. Instead, the assigned UUV diligently moves towards the old coordinate  $(x_\tau, y_\tau)$  with no ability to learn of the

threat vessel's new speed or bearing, or to calculate the new point of interception  $(x_{\tau_{NEW}}, y_{\tau_{NEW}})$ .

Figures 6.1 (a-b) and 6.2 (a-b) display the output from two such instances. Notice that the assigned UUV moves to the center of the confidence regions around the original coordinate  $(x_{\tau}, y_{\tau})$  waiting for the threat vessel, but the threat vessel changes bearing before traveling within 500 meters of the assigned UUV.

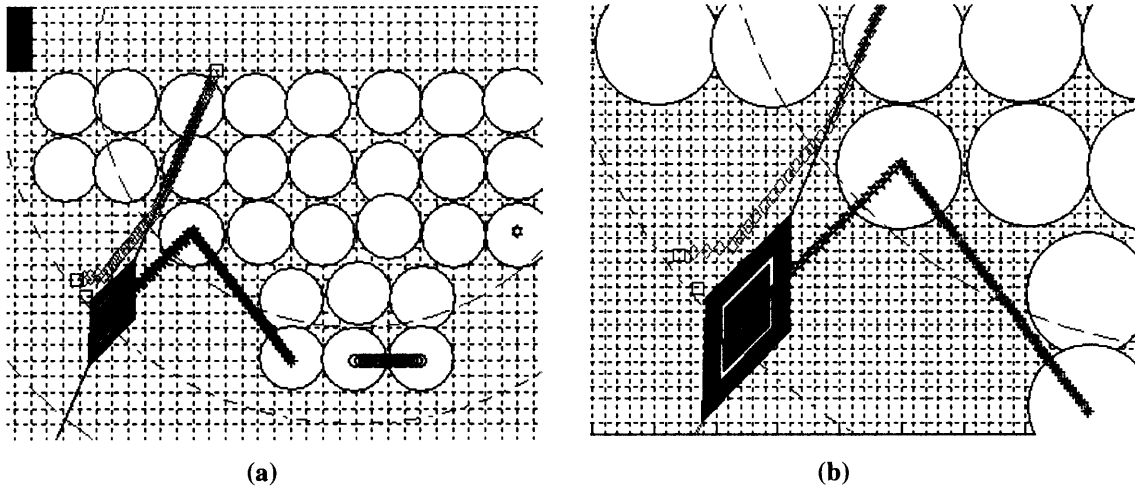


Figure 6.1 – First Example of Threat Vessel Changing Bearing After Leaving Sensor Array

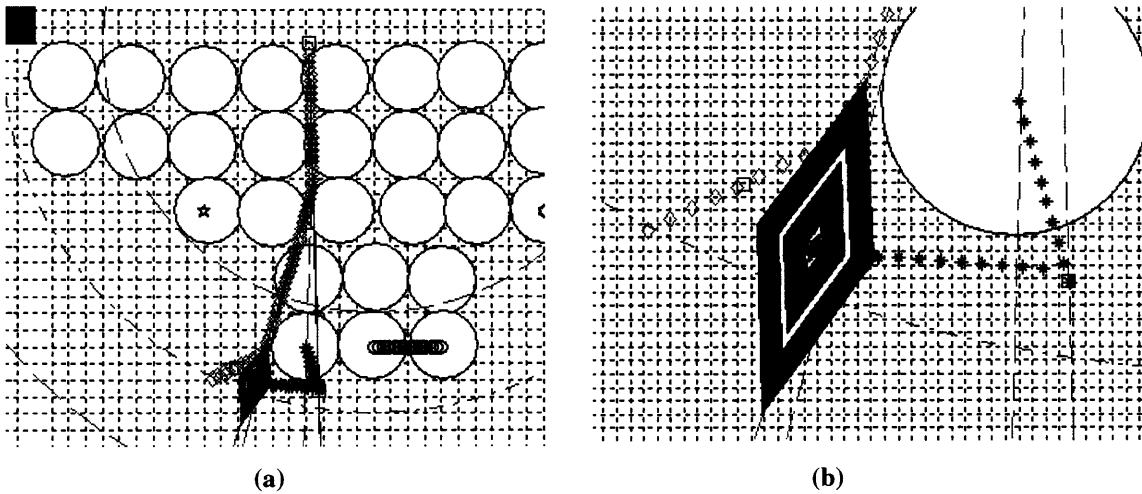


Figure 6.2 – Second Example of Threat Vessel Changing Bearing After Leaving Sensor Array

The possibility of this scenario will always exist to some extent. However, the goal of this section is to minimize its occurrence as much as possible. The section below discusses several potential methods to address this problem.

The first possibility is to increase the number of sensors. While this would be nice, it violates the fundamental assumptions about sensor availability in the real-world environment. Unrealistically increasing the number of sensors would also trivialize the underlying problem in this research. If there were sensors everywhere, there would be total information awareness and there would be little need for chasing the threat vehicles with UUVs in the first place.

The second possibility is to change the initial placement pattern of the sensors. However, the current placement pattern is consistent with the assumptions about the stochastic arrival and movement distributions of the threat vehicles. This was shown previously in Figure 3.5, where the current sensor configuration appears to optimally capture the long-term pattern of threat vehicle movement.

A third possibility is to change the initial placement of the desired arc of interception. Previously, the model attempted to locate this arc just outside of the system of sensors. Specifically, the arc was centered on the naval chokepoint with a radius of 11 kilometers as seen in Figure 4.1. By reducing the radius of the interception arc to something less than 11 kilometers, the results of the algorithm will be fundamentally changed. Specifically, there will be a trend of smaller values of  $\tau$ , which will ultimately result in interception coordinates  $(x_\tau, y_\tau)$  that tend to be closer to the sensor system. Consequently, it would be less likely that the threat vehicles change bearing after leaving the sensor system but before successful interception by the chasing UUV because there will be fewer time steps (on average) after leaving the sensor system before intercept.

The concern is that by reducing the value of  $\tau$ , the algorithm also reduces the UUV's flexibility to make up for its slower speed by forcing the UUV to reach the interception coordinate  $(x_\tau, y_\tau)$  more quickly. By decreasing the radius of the interception arc too much, the algorithm may risk scenarios in which the assigned UUV is unable to reach the point  $(x_\tau, y_\tau)$  before the threat vessel has already arrived and

continued on into open ocean. The problem formulation in Section 6.2 will address these concerns in the process of optimizing the radius of the interception arc.

## **6.2 Problem Formulation**

The purpose of this section is to formulate the arc radius problem as an optimization problem. The section begins with the underlying assumptions that describe the objectives and limitations that shape the problem. The second half of this section transforms the assumptions into the objective function and constraints of an optimization problem.

### **6.2.1 Assumptions**

The first assumption is that the basic design of the arc will not change. Specifically, the arc will be drawn as a semi-circle centered at the middle of the naval chokepoint at coordinates (20, 30). The system algorithms will also remain unchanged. Regardless of the radius of the arc, the underlying system will operate in the same manner as described in Chapters 2-5. This ensures that the only variable that could influence the system output through this optimization process is the radius of the interception arc,  $r$ . As before, it is also assumed that we start each iteration with 4 UUVs available.

The second assumption is that the arc must fully cover the width of the naval chokepoint. This implies that the radius of the arc must be greater than or equal to 10 kilometers. Without this assumption, the algorithm would be unable to respond to threat arrivals that appeared on the extreme ends of the naval chokepoint (beyond the coverage of the arc).

The third assumption is that the algorithm must attempt to cover each section of the interception arc (if possible) with at least one UUV. In this context, the word “cover” means that at least one UUV should be able to reach any point on the arc before a random



threat vessel reaches that point given the expected values of threat vessel speed. Recall the earlier assumption that the initial locations of the UUVs must be within the radius of a sensor for communication purposes. The new assumption describes the problem of the speed differential between the UUVs and the threat vehicles. If the arc radius is too small, the UUVs' slower speed may cause the set of available UUVs to be unable to cover the complete interception arc before the threat vessel arrives.

Within the limitations of the previous assumptions, the model further assumes that it would be beneficial to minimize the arc radius as much as possible. This will minimize the probability that the threat vessel changes speed or bearing after leaving the sensor array and before reaching  $(x_r, y_r)$ .

## 6.2.2 Optimization Problem

The assumptions described above are used to formulate the optimization problem below:

$$\begin{aligned} \text{Objective Function:} \quad & \text{Minimize } r \\ \text{s.t.} \quad & r \geq 10 \\ & (\text{Coverage per UUV}) * (\# \text{ UUVs Available}) \geq \text{Arc Length} \\ & \text{Initial UUV locations must be within the sensor array} \end{aligned}$$

The objective function, the first constraint, and the third constraint are straightforward from the explanations in Section 6.2.1 above. A more complete explanation of the second constraint is below. The analysis below will transform the second constraint into a form that is independent of the arc radius  $r$ .

- For any given radius of the interception arc, there is a corresponding arc length. Since the arc is a semi-circle, the arc length is easily calculated using the expression below:

$$\text{Arc Length} = \pi * r \tag{6.1}$$

- Given the expected value of threat vehicle speed (meters / sec) and any given radius  $r$  (meters), the model can calculate the expected value of  $\tau$ . Since  $\tau$  is the number of minutes until the threat vessel arrives at the arc, the expression below is clear.

$$\tau = \frac{r}{E[velocity]_{THREAT}} * \frac{1}{60} \quad (6.2)$$

- Given the calculated value of  $\tau$  and the known (constant) speed of the UUVs (meters / sec), the model can calculate the UUV coverage (meters) for each UUV. This term will be called the coverage radius of the UUV because it represents the distance (meters) that the UUV can travel in any direction from its starting point in  $\tau$  time steps. This expression is below:

$$Coverage\_radius_{UUV} = \tau * velocity_{UUV} * 60 \quad (6.3)$$

- Equation 6.3 describes the coverage radius of a UUV. If the initial UUV position is assumed to be either on the interception arc or extremely close to the interception arc, multiplying Equation 6.3 by 2 approximates the interception arc coverage per UUV.
- Given Equations 6.1, 6.2, and 6.3 above and the fact that each simulation begins with 4 UUVs, it is valid to substitute terms in the second constraint as follows:

$$2 * \frac{r}{E[velocity]_{ENEMY}} * \frac{1}{60} * velocity_{UUV} * 60 * 4_{UUVS\_Available} \geq \pi * r \quad (6.4)$$

- Equation 6.4 simplifies nicely to the form below:

$$\frac{8 * velocity_{UUV}}{E[velocity]_{ENEMY}} \geq \pi \quad (6.5)$$

- The solution analysis in Section 6.3 uses Equation 6.5 as a restatement of the second constraint.

### 6.3 Solution to Optimization Problem

The purpose of this section is to further analyze the structure of the optimization problem formulated in Section 6.2. Analysis of Equation 6.5 yields insight into the solution of the optimization problem. This section also considers a reformulation of Equation 6.5 that produces the same solution under the modeling assumptions in this scenario. The section concludes with the solution to the optimization problem. The optimal radius is used throughout the remainder of the research.

#### 6.3.1 Examination of the UUV Coverage Constraint

Upon consideration of the UUV coverage constraint (Equation 6.5), it becomes clear that the  $r$  terms cancelled out of the expression. This means that the constraint is either feasible or infeasible based only on the relative velocities of the UUV and the threat vehicle and on the value of the constant multiplied by the  $velocity_{UUV}$  term. The constraint's feasibility is independent of the interception arc radius. Given the known UUV speed and the distribution of the threat vehicle speed, this critical ratio can be calculated from Equation 6.5 and compared to the value of  $\pi$ . Table 6.1 below shows the value of this ratio for the expected threat speed and the extreme values of the threat vehicle's speed distribution:

	(m/s)	Ratio
<b>UUV Speed</b>	2.57222	
<b>Threat Speed</b>		
<b>Expected</b>	4.11552	5.00004
<b>Maximum</b>	6.17333	3.33333
<b>Minimum</b>	2.05778	9.99998

**Table 6.1 – Critical Ratio Values for Equation 6.5 Given Threat Vehicle Speed**

From Table 6.1, it is evident that the value of the critical ratio is always greater than  $\pi$ , even if the threat vehicle is at its maximum speed. This suggests that the second constraint is always satisfied and that the solution to the objective function depends only

on the first and third constraints. This is also supported through experimental spreadsheet calculations that show that the second constraint is always satisfied when varying the interception arc radius among a wide range of reasonable values in this problem. Interesting sections of this spreadsheet output are in Appendix A.

### 6.3.2 Re-formulation of the UUV Coverage Constraint

One point of clarification is important. The third constraint from the optimization formulation requires that the initial location of the UUVs be within the sensing radius of one of the sensors. However, the formulation for the critical ratio has assumed that the UUVs are initially located on the arc itself. This assumption justified multiplying the UUV coverage radius term by 2 to describe the total interception arc coverage area of a single UUV. For very small values of  $r$ , this assumption is clearly accurate. However, as  $r$  increases, there are some sections of the arc that violate this assumption. Consider this reformulation of Equation 6.5 where the value  $c$  is substituted for the constant 2.

$$\frac{4_{UUVs\_Available} * c * velocity_{UUV}}{E[velocity]_{ENEMY}} \geq \pi \quad (6.6)$$

In this formulation,  $c$  represents the multiple of the UUV coverage radius that is used “covering” the interception arc given the radius  $r$  and the constraint that the UUV must be located within the sensing radius of one of the sensors. For example, if the UUV is located on the interception arc as assumed previously, then the UUV can cover an arc length of up to twice its coverage radius, and  $c = 2$ . As the UUV’s initial distance from the interception arc increases, the value of  $c$  decreases as the UUV must spend some of its coverage radius distance traveling to the interception arc from its initial location off the arc.

This effect is only important when considering the pathological examples in which the threat vessel’s speed is extremely close to its maximum values. Using the expected value of threat vessel speed, the data from Table 6.1 shows a comfortable buffer

for the potential values of the constant  $c$  before the ratio would be less than  $\pi$ . Through calculation, this critical value of  $c$  is shown below.

$$c = \frac{\pi * E[velocity]_{ENEMY}}{4_{UUVs\_Available} * velocity_{UUV}} = 1.22135 \quad (6.7)$$

As long as  $c \geq 1.22135$ , the ratio of the UUV's speed to expected threat vehicle speed will be greater than  $\pi$  and the constraint will be satisfied. In practice, the system does not encounter such a low value of  $c$  unless the arc radius is much larger than 10 km.

However, the analysis of Equation 6.6 is continued below because it offers further insight into the behavior of the system.

If a constraint in an optimization problem is met with equality, the corresponding constraint is said to be active or binding [8]. For example, if the second constraint in this formulation was binding, Equation 6.8 would be met with equality as shown below.

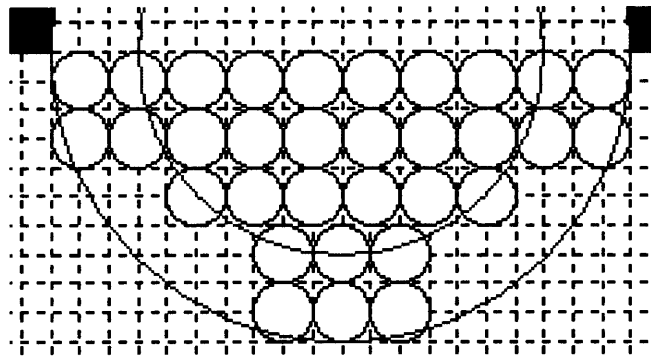
$$\frac{4_{UUVs\_Available} * c * velocity_{UUV}}{E[velocity]_{ENEMY}} = \pi \quad (6.8)$$

The constraint's previous formulation allowed the ratio to be greater than or equal to  $\pi$ .

Careful examination of Equation 6.6 yields an interesting result. By construction, the purpose of the second constraint in the optimization model was to require the model to make up for the speed differential between the UUV and the threat vehicles. The initial assumption was that the second constraint would be a binding constraint in the model that tended to encourage the radius to be larger. This is consistent with the initial assumption that a larger radius for the interception arc would better allow a UUV to compensate for its slower speed en-route to the interception point.

However, the reformulation of the second constraint in Equation 6.6 offers a very different result from what was expected. Since the third constraint limits the initial positions of the UUVs to the sensor field, large values of  $r$  tend to make the value  $c$  in Equation 6.6 smaller. This is because the interception arc will have regions not included in the area of the sensor field as the arc radius gets larger. Figure 6.3 below shows that as the radius of the interception arc exceeds 7 km, the value of  $c$  for some candidate points on the arc begins to decrease because some of the arc is not included in the sensor

coverage areas. The outer arc in Figure 6.3 is the minimum allowable radius of 10 km while the inner arc has a radius of 7 km.



**Figure 6.3 – Radius 7 km (Inner Arc) & Radius 10 km (Outer Arc)**

As the arc radius continues to get much larger than the 10 kilometer minimum, the value of  $c$  in Equation 6.6 will eventually become less than 1 as the UUV requires the majority of its coverage radius just to reach the interception arc from its initial location within a sensor radius.

If the second constraint were binding, the reformulation of the second constraint in Equation 6.6 would motivate the system to decrease the interception arc radius below the 10 km minimum rather than increase the sensor radius. However, after experimentation with the model, it became clear that the second constraint is not binding, even when considering the maximum possible speed of the threat vessel. This will become even more evident in Chapter 7 when discussing the dynamic programming solution to the UUV placement problem. The binding constraint for the interception arc radius problem becomes the first constraint,  $r \geq 10$ , rather than the second constraint as initially assumed.

### **6.3.3 Solution to the Optimization Problem**

Since the goal is to minimize the arc radius in the objective function and the binding constraint is  $r \geq 10$ , the obvious optimal solution for the radius length becomes  $r = 10$  kilometers. This is in contrast to the initial assumption of  $r = 11$  kilometers in the

base case scenario. To verify this solution, 10,000 runs of the simulation were processed for each of 4 radius values ranging from 10 to 13 kilometers. These simulation runs allowed for reassignment of UUVs as discussed in Chapter 5. The results are in Table 6.2 below.

<b>Radius (km)</b>	<b>Success</b>	<b>Failure</b>
10	89.0%	11.0%
11	86.3%	13.7%
12	81.9%	18.1%
13	76.3%	23.7%

**Table 6.2 – Simulation Results with Different Interception Arc Radius Lengths**

As expected from the discussion above, the optimal radius is the minimum allowable value,  $r = 10$  kilometers. Using this optimal radius instead of the assumed value of 11 kilometers, the algorithm decreases the probability of failure by 19.9 %. Conversely, as the value of  $r$  increases, there are an increasingly large number of failures. Given this solution, the algorithm will assume that the optimal interception arc radius is 10 kilometers for the calculations in Chapter 7.

## Chapter 7 Optimizing Initial UUV Locations Through Dynamic Programming

The final set of assumptions that need to be optimized are the initial UUV coordinates. The focus of this chapter is to optimize these coordinates using the powerful tools of dynamic programming.

### 7.1 Motivation to Optimize the UUV Locations

The best insights into improving the initial UUV coordinates may be gained by analyzing the unsuccessful iterations. In the vast majority of unsuccessful iterations, either UUV 1 or UUV 2 was the primary chase vehicle. These are the UUVs on the sides of the sensor array, closest to the edges of the naval chokepoint. Figure 7.1 below shows the assumed initial locations for the UUVs in the base case scenario. In this figure, UUV 1 and UUV 2 are indicated by the five-pointed star and the six-pointed star.

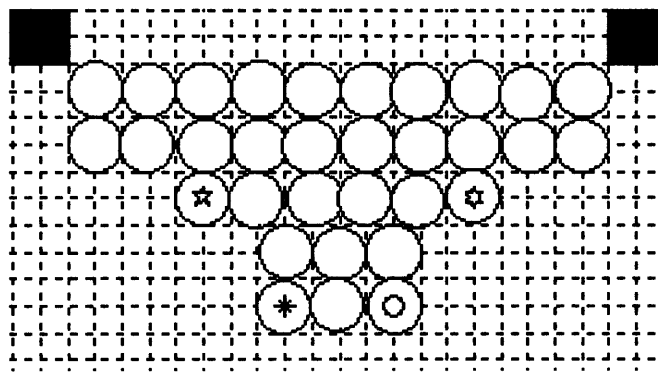


Figure 7.1 – Initial UUV Locations as Assumed in the Base Case Scenario

Iterations that assigned UUV 1 or UUV 2 as the chase vehicle typically corresponded to “tail events” in the distributions defining threat vehicle arrivals or movements. For these “tail events,” the threat vehicle arrived near one of the ends of the



naval chokepoint and / or had an initial bearing close to 90 degrees or 270 degrees. Although the model successfully intercepted many such tail events, almost all of the failures occurred when tail events happened. This suggests that the initial locations of UUV 1 and UUV 2 are probably not optimal.

Specifically, Figure 7.1 suggests that threat arrivals near the ends of the naval chokepoint would have a reasonable probability of passing between the assigned UUV and the end of the chokepoint before the UUV could arrive at the point of desired interception  $(x_i, y_i)$ , given the assumed initial locations for UUVs 1 and 2. This suggests that the model might be improved by moving the initial locations of UUVs 1 and 2 closer to endpoints of the naval chokepoint.

The results of Chapter 6 improved system performance by decreasing the radius of the arc of interception. The goal of Chapter 7 is to further improve system performance through the optimal initial placement of UUVs. Many optimization techniques could solve this problem. This research will solve the problem as a dynamic program.

## **7.2 Dynamic Programming Theory**

The discipline of dynamic programming has a rich theory with applications in many research areas. The purpose of this section is to provide a very brief background into the subject and explain the basic principles used in this simulation. Many popular textbooks offer a more thorough coverage of dynamic programming, including the original text by Bellman [3] and more recent books by Bertsekas [4], and Bertsekas and Tsitsiklis [7].

Bellman developed the term “dynamic programming” to describe the mathematical theory of multi-stage decision processes [3]. The objective is to minimize a cost or maximize a reward through a sequence of optimal decisions – the optimal policy. Each decision results in an immediate cost, but it also changes the context in which future decisions are made thus changing the costs of later stages [7]. Such decision systems are

often large-scale and difficult to analyze. However, the modeling framework of dynamic programming offers insight into the structure of the solution over time. The fundamental elements of the dynamic programming model are described in the paragraphs below, based on the information in [4].

The state variables  $(x_k)$  describe the current state of the system and summarize the past information that is relevant for future optimization, where  $k$  indexes time. The control variable  $(u_k)$  is the decision to make at time  $k$ . The disturbance parameter  $(w_k)$  represents system noise that may interject randomness into the system.

The system equations describe how the state variables change over time given the control variables and the disturbance. The system equations take the form below.

$$x_{k+1} = f_k(x_k, u_k, w_k), \quad k = 0, 1, \dots, N-1 \quad (7.1)$$

The cost function describes the cost incurred at each state given the state variables, the control variable, and the disturbance. Since the disturbance is often modeled as a random variable, the expected cost is determined through the expression below where  $g_N(x_N)$  is the terminal cost. This cost structure is additive over time.

$$E_{w_k} \left\{ g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, w_k) \right\} \quad (7.2)$$

The dynamic programming algorithm is the mechanism through which the model minimizes system cost over time. The basic form of the algorithm is below, where the  $J_{k+1}$  term represents the cost-to-go.

$$J_k(x_k) = \min_{u_k \in U_k(x_k)} E_{w_k} [g_k(x_k, u_k, w_k) + J_{k+1}(f_k(x_k, u_k, w_k))], \quad k = 0, 1, \dots, N-1 \quad (7.3)$$

$$J_N(x_N) = g_N(x_N), \quad k = N \quad (7.4)$$

The principal method for calculating the optimal cost-to-go vector is value iteration [7]. Given the transitional probabilities  $p_{ij}$ , value iteration is described by the expression below [4]:

$$J_{k+1}(i) = \min_{u \in U(i)} \left[ g(i, u) + \sum_{j=1}^n p_{ij}(u) J_k(j) \right], \quad i = 1, \dots, n \quad (7.5)$$

Section 7.3 uses this basic dynamic programming structure to determine the optimal initial locations for the UUVs.

### **7.3 Applied Dynamic Programming Algorithm**

As seen in Section 7.1, the problem with the original assumptions for initial UUV locations was that part of the interception arc was not covered by UUVs near the ends of the chokepoint. This allowed the arrivals that corresponded to tail events in the threat vessel distributions to have a good chance of escaping without interception by the UUVs. This is particularly discouraging if the earlier assumption that the threat vessels had no knowledge of our monitoring in the vicinity of the chokepoint is relaxed. If the threat vehicles become aware of the monitoring, they would certainly learn of this weakness in the UUV / sensor system and exploit it to their benefit.

Consequently, the goal in this section is to use the dynamic programming methodology discussed in Section 7.2 to optimize the initial UUV placement. Specifically, the objective is to optimize the UUV placement such that each section of the entire interception arc is covered by at least one UUV. The same basic methodology will also be used to optimize the coverage positions for the scenarios in which 3, 2, or 1 UUVs remain after the initial UUV assignment.

### 7.3.1 Dynamic Programming Modeling Methodology

The sections below begin the dynamic programming modeling process by stating and explaining the underlying assumptions and modeling requirements. The design of these assumptions is to stress the system to the worst case scenario. Under these assumptions, any solution that covers the complete length of the interception arc will solve the system optimally under any possible distribution of threat vessel disposition.

#### 7.3.1.1 Speed of Threat Vehicles

The first assumption is that the threat vessel is traveling at its maximum speed of 12 knots. This will correspond to the lowest possible value of  $\tau$ , giving the UUV approximately 27 minutes of reaction time after the system initially recognizes a threat arrival. This requires the UUVs to respond as rapidly as they would ever be required given the probabilistic distributions of threat vessel disposition. Given the assumed speed of 5 knots for UUVs (2.5722 m/s), this means that the coverage radius for each UUV is only 4,166.6 meters, as described by the expressions below. Note that all of the velocities have been converted from knots into meters per second using the conversion factor from Section 3.2.4.

$$\begin{aligned}\tau_{MIN} &= \frac{arc\_radius}{threat\_velocity_{MAX}} * \frac{1}{60} \\ &= \frac{10,000(m)}{6.1733(m/s)} * \frac{1(min)}{60(sec)} = 26.9979 \text{ minutes}\end{aligned}\tag{7.6}$$

$$\begin{aligned}UUV \text{ Coverage Radius} &= velocity_{UUV} * \tau_{MIN} \\ &= \frac{2.5722(m)}{1(sec)} * 26.9979(min) * \frac{60(sec)}{(min)} = 4,166.6 \text{ meters}\end{aligned}\tag{7.7}$$

### **7.3.1.2 Initial UUV Locations within the Sensor Array**

The second assumption is that the initial UUV locations must be within the sensing radius of one of the sensors in the sensor array. The purpose of this assumption is to ensure that the UUVs have good communications when the system is initially activated by a threat arrival. The model previously assumed that the sensing radius of each sensor is one kilometer. However, it is important to remember that the placement of each sensor can vary as described in Section 2.4. With probability 99.7%, the sensor center coordinates will vary by no more than 100 meters in each of the four cardinal directions. Consequently, the model attempts to ensure that the initial UUV location is within the sensing radius of one of the sensors by requiring that the initial UUV location is within 900 meters of the optimal sensor coordinates. In this way, it is 99.7% certain that the UUV is within the sensing radius of the sensor, even if the realization of the random variable that defines the sensor's center coordinates is 3 standard deviations from the mean. Given the small UUV coverage radius and the limiting requirements of the second assumption, the difficulty of the underlying dynamic program becomes clear.

### **7.3.1.3 Optimize Half of the Arc and Reflect Data to the Other Side**

The third assumption concerns the basic structure of the optimization problem as it relates to the sensor fields. The scenario in this research takes advantage of the problem's symmetry around the line  $x = 20$ . This is due to the assumed structure of the sensor array system and the assumed distributions for the threat vessels' arrivals, velocities, and bearings. Consequently, the dynamic program can be structured in order to optimize the placement of 2 UUVs along one-half of the interception arc, between the coordinates (10, 30) and (20, 20). After finding the optimal UUV coordinates for these two UUVs, the algorithm simply reflects the optimal coordinates around the line  $x = 20$  to find the optimal UUV coordinates for the second half of the interception arc.

### 7.3.1.4 Define the list of Candidate UUV Coordinates

In order to limit the span of the dynamic program, the model assumed a piecewise linear curve that contains the complete set of candidate points for one-half of the interception arc. This curve approximates the lower envelope of the sensor field for the left half of the sensor array, as shown by the line in Figure 7.2 below:

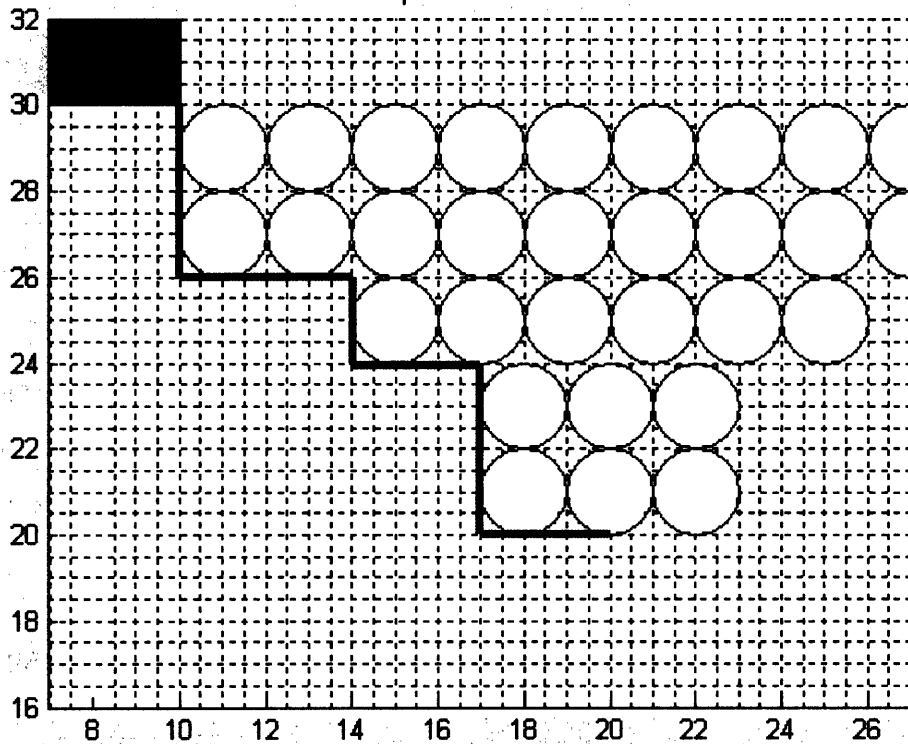
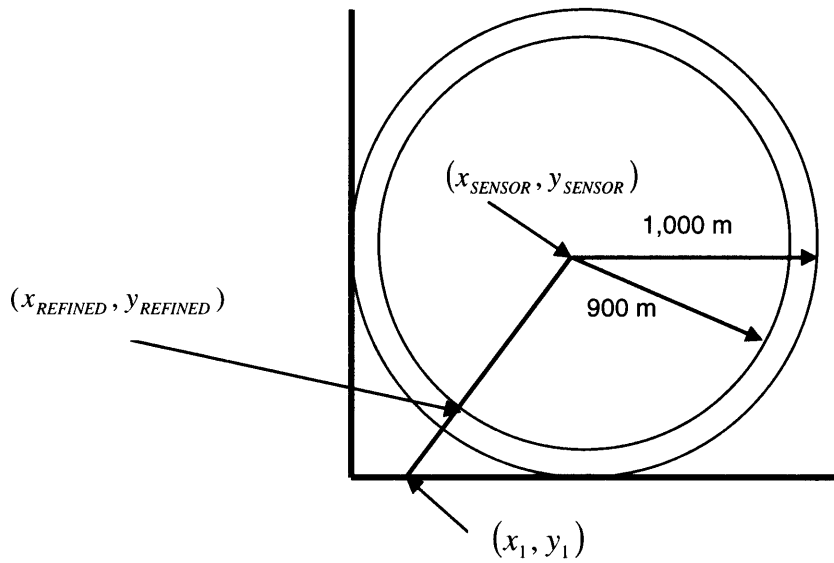


Figure 7.2 – Curve of Candidate Points for One-Half of the Sensor Array

By considering only the lower envelope of the sensor field, the algorithm assumes that any point above the lower envelope would be dominated by at least one corresponding point on the lower envelope. This is true because the lower envelope of points gives the UUVs the maximum expected initial distance from the threat arrival while minimizing the expected initial distance between the start point and the point  $(x_t, y_t)$  on the arc of optimal interception. Consequently, the points on the lower envelope of the sensor field allow for the least travel time to the point  $(x_t, y_t)$  while maximizing the UUV's ability to overcome the speed differential between threat vessel and UUV.

Upon careful inspection, all of the points on the curve in Figure 7.2 violate the second assumption by at least 100 meters. To overcome this problem, the algorithm uses a post-processing algorithm. After the dynamic program determines the optimal point along the piece-wise linear curve in Figure 7.2, the post processing algorithm determines the point in the sensor array that is closest to the optimal point on the curve while being no further than 900 meters from any of the ideal sensor centers. The mathematics of this process are described below.

- Start with a coordinate pair – the output of the dynamic program. The two points  $(x_1, y_1)$  and  $(x_2, y_2)$  are the optimal locations for the two UUVs on the left side of the sensor array. For each of the two points, follow the process below to determine the refined coordinates.
- Calculate the distance from  $(x_1, y_1)$  to the desired center grids of each of the 32 sensors.
- Determine which distance is shortest. Label the desired center coordinates of the closest sensor  $(x_{SENSOR}, y_{SENSOR})$ .
- The algorithm determines the point on a circle of radius 900 meters around  $(x_{SENSOR}, y_{SENSOR})$  that intersects the line connecting the points  $(x_1, y_1)$  and  $(x_{SENSOR}, y_{SENSOR})$  as shown in Figure 7.3 below. This point of intersection is the refined point  $(x_{REFINED}, y_{REFINED})$ .



**Figure 7.3 – Refinement Process for Dynamic Programming Output**

- Repeat this process for the second coordinate of the optimal coordinate pair.

If there is more than one pair of candidate points that each cover the interception arc with optimality, the model conducts the post-processing algorithm for each pair of “equally optimal points.” After post processing is complete, the algorithm recomputes the arc coverage for each coordinate pair using the refined grid locations. In some instances, a coordinate pair that had previously covered the entire arc will not cover the entire arc after considering the post-processing data. If there are still multiple optimal solutions, any of them are acceptable, optimal solutions. By design, the algorithm in this research selects the optimal solution from this refined set of optima whose second UUV is closest to the point (20,20). This will tend to bias the weight of any overlapped coverage towards the center of the sensor system, consistent with the assumptions of threat vessel arrivals and movements.



### **7.3.1.5 Defining the Discretization of Candidate Points**

The purpose of defining the curve of candidate points in Figure 7.2 was to limit the span of the dynamic program to a tractable set of points. However, the piece-wise linear curve defined in Figure 7.2 is still made up of an infinite number of points. To limit the set of points to a tractable number, the user defines the desired separation between candidate points along the curve. Based on the separation between points, the algorithm fills an array with all of the discrete candidate points along the curve in Figure 7.2. The points in the candidate array form the basis from which the dynamic program considers all possible combinations of point pairs.

### **7.3.1.6 Weighting of the Interception Arc**

The objective of the dynamic program is to cover as much of the interception arc as possible through the initial location of the UUVs, given the lowest possible value of  $\tau$ . However, the model does **not** assume that each equal-length segment of the arc has the same importance. In contrast, the algorithm assumes that the center of the arc has the highest importance while the extreme ends of the arc have the lowest importance. This is consistent with the expected values of the distributions that define the threat vehicles' arrivals and movements. Consequently, the weighting ensures that the dynamic program covers the middle of the interception arc by assigning a higher value to the center of the arc than the ends of the arc. The mathematical formulation of this weighting will be discussed in Section 7.3.2.4 when considering the formulation of the dynamic programming cost function.

### **7.3.1.7 No Added Value for Overlapped Coverage**

The weighting described above assigns each section of the interception arc a value that increases when approaching the center point (20, 20) and decreases when

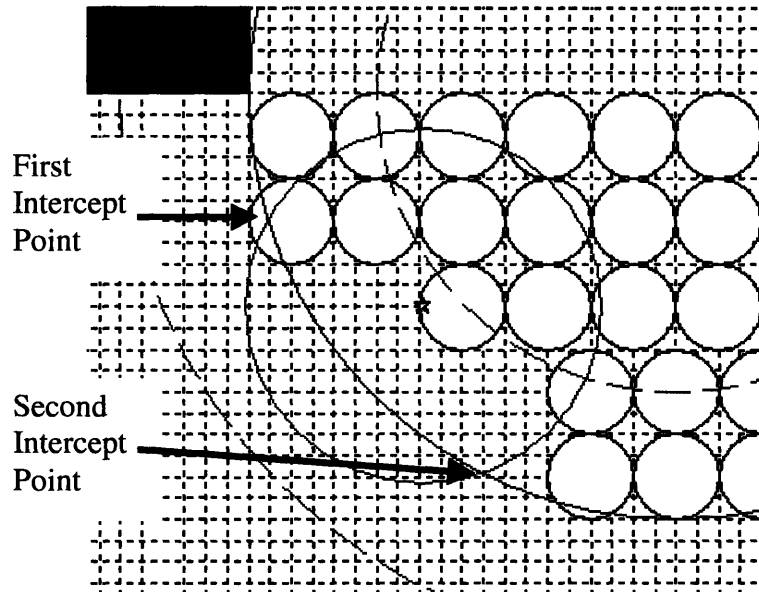
approaching the edge of the naval chokepoint at (10, 30). The dynamic program is designed to maximize the accumulation of this value along the arc by determining which sections of the arc are covered by UUVs at each of the possible candidate points. For many of the possible pairs of candidate points, the interception arc coverage regions for the two UUVs overlap. Because of the discretization in this scenario, very minor overlap is usually optimal because it ensures complete coverage of a high-valued region. However, the model credits arc coverage of the overlapped region only once. This tends to minimize the region of overlap.

### **7.3.2 Design of the Dynamic Program**

Given the assumptions above, the dynamic program is designed to maximize the weighted coverage of the interception arc. This section describes the detailed structure of the dynamic program, specifying the structure in terms of the state, control, system equations, cost function, and dynamic programming algorithm.

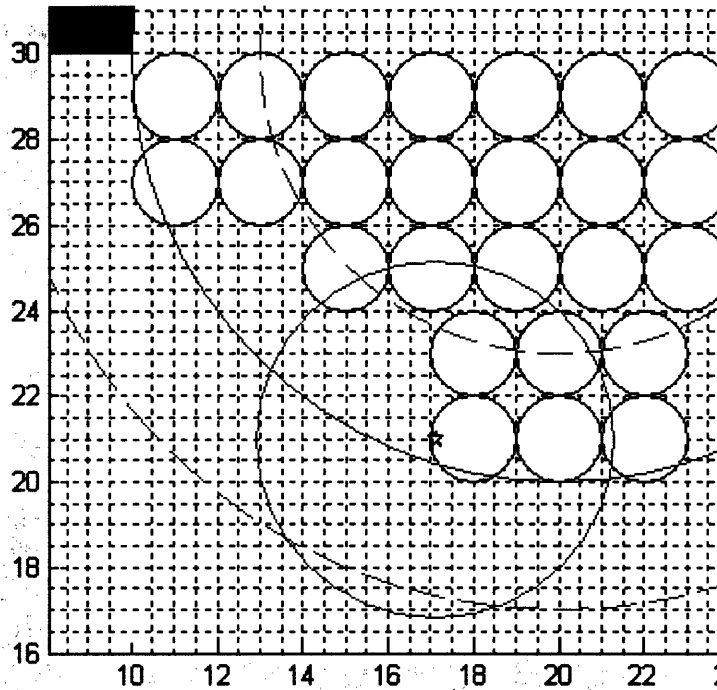
#### **7.3.2.1 State Variables**

The state of the dynamic program must include information about how many UUVs have been placed and the arc coverage corresponding to those UUVs. The UUV coverage radius was already calculated in Section 7.3.1.1 given the assumed values for UUV speed and the lowest possible value of  $\tau$ . Given this calculated coverage radius, each candidate point on the piece-wise linear curve in Figure 7.2 corresponds to two unique points on the interception arc where the coverage radius of the UUV intersects the intersection arc. The arc between these two intercept points corresponds to the unique arc coverage region for any candidate point along the outer envelope of the sensor system. This is shown in Figure 7.4 below for a UUV placed at the coordinates (14.1, 25).



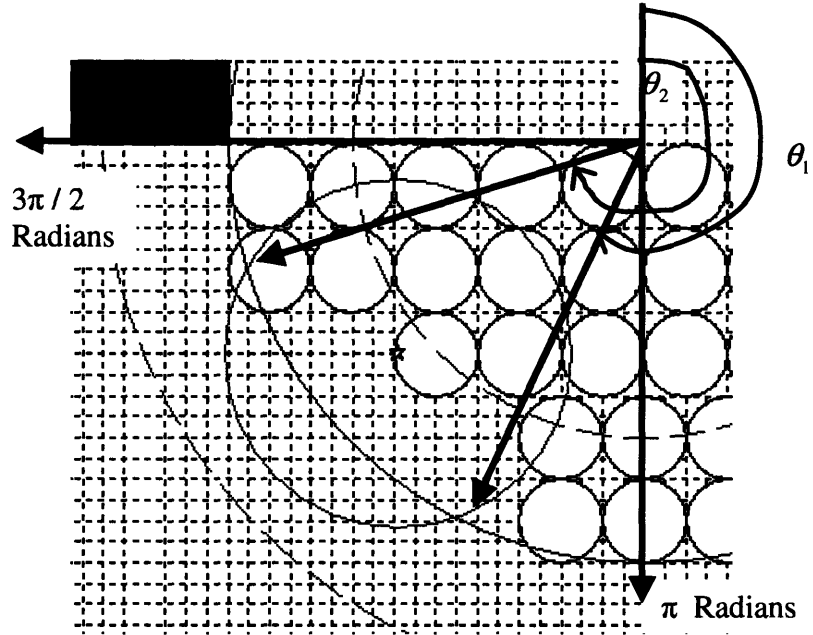
**Figure 7.4 – Coverage Region on the Interception Arc from One UUV Location**

If the coverage area on the intersection arc goes beyond the endpoint at (10, 30) or the center point at (20, 20), the algorithm truncates the coverage area corresponding to the UUV's coordinates. In this way, the UUV candidate point can only accumulate value for the portion of the interception arc corresponding to the left half of the map, between the points (10, 30) and (20, 20). For example, Figure 7.5 below shows a scenario in which the coverage radius of the UUV located at (17.1, 21) goes beyond the center point (20, 20). Specifically, the  $x$  coordinate of the right-most coverage point is greater than 20. In this example, the algorithm would credit the UUV with coverage only up to the point (20, 20) since the model is considering only the left half of the sensor field. In the same manner, the algorithm would truncate any coverage that goes beyond the left edge of the naval chokepoint (any coverage value whose  $y$  coordinate is greater than 30.)



**Figure 7.5 – UUV Coverage Radius Exceeds the Bounds of the Problem**

Having calculated the unique interception points on the optimal interception arc for any coordinate in the candidate array, the algorithm calculates the radian measure that corresponds to each of the two interception points, using the center of the naval chokepoint as the basis for the radian measure. This is depicted in Figure 7.6 below.



**Figure 7.6 – Translating Interception Points to Radian Measures**

In Figure 7.6, it is clear that a radian measure  $\theta$  for any interception point must follow

$$\theta \in \left\{ \pi \leq \theta \leq \frac{3\pi}{2} \right\} \quad (7.8)$$

Using the process described above, the algorithm maps two distinct radian measurements with each candidate point for UUV location, corresponding to the two points of interception between which that UUV could “cover” the interception arc.

Using this framework, the state variables for the dynamic program are defined as follows:

- $x_k$  = number of UUVs already placed at the beginning of time  $k$ ,  $x_k \in \{0,1,2\}$ .

Furthermore,  $x_0 = 0$  and  $x_N = 2$ .

- $\theta_1, \theta_2$  = radian measures (if applicable) corresponding to the first UUV placement.
- $\gamma_1, \gamma_2$  = radian measures (if applicable) corresponding to the second UUV placement.

### 7.3.2.2 Control Variable

The control variable in this dynamic program is whether or not to place the next UUV in the next candidate position considered from the array of candidate points. As such, the control is binary where  $u_k = 1$  if the model places the next UUV in the next candidate point, and  $u_k = 0$  if the model does not place the next UUV in the next candidate point.

### 7.3.2.3 System Equation

Recall that  $x_k$  corresponds to the number of UUVs placed at the beginning of time  $k$ , where  $x_0 = 0$ . If the algorithm places a UUV into position during time  $k$ , then  $u_k = 1$ . This suggests the system equation below.

$$x_{k+1} = x_k + u_k \quad (7.9)$$

Note that there is not a disturbance term ( $w_k$ ) in this formulation. If  $x_k \neq 0$ , then the values of  $\theta_1$ ,  $\theta_2$ ,  $\gamma_1$ ,  $\gamma_2$  are calculated as described in Section 7.3.2.1. If  $x_k = 1$ , then  $\gamma_1$  and  $\gamma_2$  are both set equal to the base measurement of  $\frac{3\pi}{2}$  because the second UUV is not yet placed.

### 7.3.2.4 Cost Function

In this dynamic program, the cost function uses the perspective of accumulating value rather than incurring cost. Section 7.3.1.6 described the weighting of the interception arc so that each portion of the arc corresponded to a measurable value score.

In this way, the cost function is the expression that describes the accumulation of this value score along the length of the interception arc.

Fundamental to the design of the cost function is the methodology used for weighting the interception arc. Earlier, the algorithm calculated the values of  $\theta_1$  and  $\theta_2$  that correspond to the radian measures of the section of the interception arc covered by one UUV. Now, the objective is to express the radian measure of any angle  $x$  on this curve as a value between 0 and 1. In particular, the radian measure  $x = \frac{3\pi}{2}$  should correspond to a value of 0 while the radian measure  $x = \pi$  should correspond to a value of 1. This is accomplished through the expression below, where  $\alpha$  is a constant that defines the formulation's weighting on the optimization process.

$$f(x) = \left[ \frac{\frac{3\pi}{2} - x}{\frac{\pi}{2}} \right] * \alpha \quad (7.10)$$

This formulation assigns  $f(x)$  a score value between 0 and  $\alpha$  for each radian measure  $x$ . By integrating Equation 7.10 between the values  $\theta_1$  and  $\theta_2$ , the algorithm determines the total score for the portion of the interception arc covered by the corresponding UUV, as shown below.

$$\int_{\theta_1}^{\theta_2} f(x) dx \quad (7.11)$$

$$F(x) = \left( \left( 3x - \frac{x^2}{\pi} \right) * \alpha \right)_{\theta_1}^{\theta_2} \quad (7.12)$$

$$\text{Score of Covered Arc} = \left( \left( 3 * \theta_2 - \frac{\theta_2^2}{\pi} \right) * \alpha \right) - \left( \left( 3 * \theta_1 - \frac{\theta_1^2}{\pi} \right) * \alpha \right) \quad (7.13)$$

This score measures the value of the interception arc segment covered by each UUV. As such, Equation 7.13 is the form of the “cost function” for the potential UUV placement.

One additional clarification is important. If  $x_k = 2$ , then the algorithm has placed both UUVs and there is the potential for an overlap of coverage. By design, if an overlap exists,  $\theta_1 > \gamma_2$ . In this case, the model must ensure that it only counts the value of the overlapped region once. This is accomplished by considering the radian measures corresponding to the endpoints of the union of the two coverage regions. This is described in the equation below.

$$\text{Total Score} = \left( \left( 3 * \theta_2 - \frac{\theta_2^2}{\pi} \right) * \alpha \right) - \left( \left( 3 * \gamma_1 - \frac{\gamma_1^2}{\pi} \right) * \alpha \right) \quad (7.14)$$

If  $\theta_1 \leq \gamma_2$ , then there is **not** an overlap in coverage between the two UUVs. In this case, the total score is the sum of the scores of the two coverage regions as shown below.

$$\text{Score} = \alpha * \left( \left( \left( 3 * \theta_2 - \frac{\theta_2^2}{\pi} \right) - \left( 3 * \theta_1 - \frac{\theta_1^2}{\pi} \right) \right) + \left( \left( 3 * \gamma_2 - \frac{\gamma_2^2}{\pi} \right) - \left( 3 * \gamma_1 - \frac{\gamma_1^2}{\pi} \right) \right) \right) \quad (7.15)$$

### 7.3.2.5 Implementing the Dynamic Programming Algorithm

The dynamic programming algorithm in the simulation is described below:

1. Start with the first point in the candidate array. By design, this is always the point (10, 30). Place the first UUV at this point.
2. Calculate the value score for the region of the interception arc covered by the first UUV. This can be thought of as the “cost” of placing the first UUV at its location from Step 1 above.
3. At this point, the task of the dynamic program is to calculate  $J^*$  given the placement of the first UUV. Intuitively, the algorithm is maximizing the “value to go” for the system given the initial UUV placement in Step 1. This is accomplished through the process described below.



- Consider the next point in the array of candidate points, and place the second UUV at this point.
  - Determine whether or not there is an overlap in coverage on the interception arc given the placement of the two UUVs.
  - Calculate the total value for coverage of the interception arc using either Equation 7.14 or Equation 7.15.
  - Repeat the bulleted algorithm above for every subsequent point in the candidate array using the same initial UUV location from Step 1 of the outer algorithm. Through this process, the model calculates the total value score for every possible combination of coordinates for the second UUV, given the initial point of the first UUV from Step 1 of the outer algorithm.
  - Determine which pair of points has the highest value score given the initial placement of the first UUV in the outer algorithm. If more than one coordinate for the second UUV are equally optimal, choose the coordinate closest to the point (20,20). This biases “equally optimal solutions” towards the middle of the sensor array consistent with the expected values of the threat vessel movement distributions.
4. Repeat the outer algorithm for each initial UUV placement coordinate in the array of candidate points. Store the maximum total value score that corresponds to each initial UUV placement coordinate in a new array of maximum scores.

Once the maximum score array is filled with the highest value scores for each initial candidate point, the algorithm compares these values to determine the best overall coverage of the interception arc. At this point, two scenarios are possible.

In the first scenario, there is one coordinate point pairing in the maximum score array that clearly dominates all other possibilities. This happens if none of the point pairs can fully cover the interception arc or if exactly one of the point pairs fully covers the interception arc. When this happens, the algorithm conducts the post-processing

refinement on the optimal point pair to ensure that the coordinates are within a sensor radius, and it returns the optimal grids to the user. In practice, this only happened when the program attempted to optimize the system starting with fewer than four UUVs.

In the second scenario, there are multiple combinations of point pairs that fully cover the interception arc. In this case, the algorithm conducts the post processing algorithm on each combination of optimal point pairs and compares the coverage of the refined coordinates. Depending on the original separation between candidate points, there are some scenarios in which only one coordinate point pair remains optimal after refinement. If this happens, that point pair is the obvious choice. In other cases, there may still be multiple point pairs that cover the complete interception arc after refinement. If this happens, the algorithm chooses the point pair whose second UUV coordinate is closest to the point (20, 20) as discussed above. In practice, this second scenario applied when optimizing the system with four available UUVs. Depending on the desired separation between candidate points, the refined optimal point pairs fell into both categories described in this second scenario.

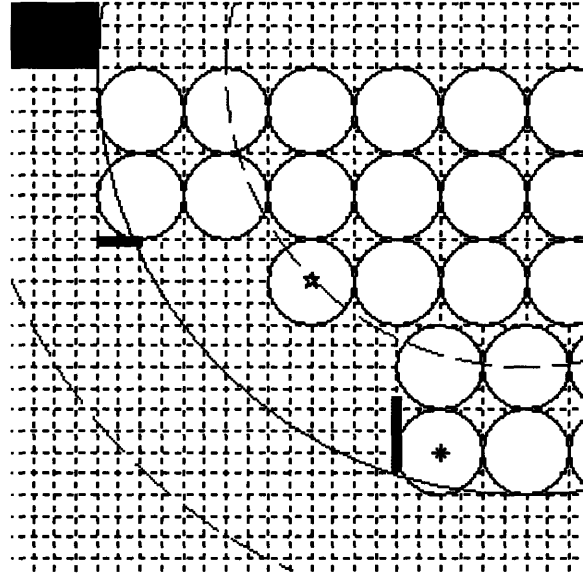
### **7.3.3 Results of the Dynamic Program**

This section will consider the optimal regions for initial UUV placements, the “curse of dimensionality” that exists when solving this dynamic program, a discretization paradox that occurs when solving the program with different separation values between candidate points, and the overall optimal initial coordinates for the case of four UUVs.

#### **7.3.3.1 Optimal Regions for Initial UUV Placements**

The results of the dynamic program varied slightly, depending on the user-controlled separation between candidate points along the lower envelope of the sensor system. In general, the optimal point pairs always came from the two regions designated

by the lines in Figure 7.7 below. These calculated optimal areas should be compared with the initially assumed UUV locations as indicated by the star and the asterisk.



**Figure 7.7 – Calculated Optimal Areas & Initially Assumed Starting Points**

The initial assumption for the lower of the two UUVs (the asterisk) is consistent with the optimal area as determined from the dynamic programming output. However, the initial assumption for the upper-left UUV (the star) is not adjacent to the optimal area as determined by the dynamic program. Instead, the initially assumed starting point failed to cover the portion of the interception arc near the ends of the naval chokepoint. This explains the relatively high number of failures in the original model that corresponded to tail events in the threat arrival distributions.

### **7.3.3.2 The Curse of Dimensionality**

Before continuing with the results, it is interesting to digress for a moment to discuss the so-called “curse of dimensionality” and how it applies to this problem. One author characterized this “curse” as the exponential rise in the time and space required to compute an approximate solution to a dynamic program as the dimension increases [27].

This arises in the dynamic program as the separation value between candidate points decreases, causing a corresponding increase in the dimension of the state space. Allowing a relatively high separation value of 500 meters produced only 40 candidate points on the left half of the sensor system. However, by reducing the separation value to 10 meters, there are 2,000 candidate points. The dynamic program was tested with a variety of separation values, ranging from 500 meters to 1 meter. During this experimentation process, any separation value less than 10 meters resulted in computer failure because of a lack of virtual memory.

### **7.3.3.3 The Discretization Paradox**

After running the dynamic program with different separation values, the coverage capabilities of the optimal results were compared. As the separation value between candidate points decreased, the program tended to produce better results. However, this was not always true. Even though every point considered with a higher separation value is also considered with a lower separation value, the optimal point paired with the initial candidate point changes if the separation value changes. Consequently, after post-processing is complete, the lower separation value does not necessarily mean that there will be a better overall objective function value. As the separation value is lowered in some cases, the corresponding optimal point pairs in the optimal range shown in Figure 7.7 tended towards areas that had larger refinement (post-processing) shifts. In some cases, the coverage of this refined data was not as good as the refined optimal point pair from iterations with higher separation values. This difference was extremely small, but it enabled the user to distinguish between the optimal results corresponding to different separation values.

### **7.3.3.4 Optimal Initial Coordinates for Scenarios with Four UUVs**

The overall optimal results are in Table 7.1 below, along with the originally assumed values and the distance between the assumed values and the calculated optima.

As seen below, the total distance between the four calculated optimal points and the four originally assumed coordinates is more than 10 kilometers. This is a substantial improvement when the objective is to cover an interception arc whose total length is approximately 31.4 kilometers. As discussed above, most of this error was in the placement of UUV 1 and UUV 2 – the two UUVs that cover the ends of the naval chokepoint.

	<b>Optimal</b>	<b>Assumed</b>	<b>Total Difference (km)</b>
<b>UUV 1</b>			<b>4.148</b>
x coordinate	11	15	
y coordinate	26.1	25	
<b>UUV 2</b>			<b>4.148</b>
x coordinate	29	25	
y coordinate	26.1	25	
<b>UUV 3</b>			<b>0.9</b>
x coordinate	17.1	18	
y coordinate	21	21	
<b>UUV 4</b>			<b>0.9</b>
x coordinate	22.9	22	
y coordinate	21	21	
<b>Total Difference (km)</b>			<b>10.096</b>

**Table 7.1 – Optimal Initial UUV Locations vs. Assumed Values**

Figure 7.8 below shows the optimal UUV locations with respect to the sensor array. The circles around the UUVs correspond to the UUV coverage radii given the highest possible threat vehicle speed. The dynamic programming solution covers the full length of the interception arc with four UUVs, while minimizing overlaps in coverage. When a large overlap does occur, it is in the center of the sensor system consistent with the expected values of the threat vehicles' stochastic distributions.

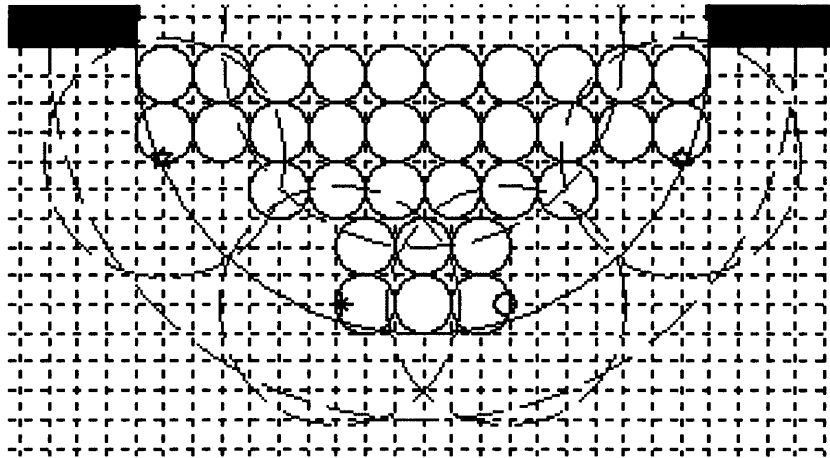


Figure 7.8 – Optimal UUV Locations with Four UUVs

### 7.3.4 Optimizing Coverage Positions

Having used the dynamic program to optimize the starting points for the four UUVs, it naturally follows that the dynamic programming process could be used to optimize the coverage positions in the case that there are 3, 2, or 1 UUV(s) remaining after the initial assignment. Each of these cases are handled separately in the sections below, but all three use the same basic assumptions and dynamic programming structure described earlier in this chapter.

#### 7.3.4.1 Three UUVs Remaining

When there were four UUVs, the basic form of the problem was to optimize the placement of two UUVs on the left half of the sensor array and then reflect this optimal solution to the right half of the sensor array. A similar symmetry will be employed for the case of three UUVs remaining.

The model begins by placing a UUV at the point  $(20, 20.1)$ . The point  $(20, 20)$  is the center of the interception arc, the center of the sensor array, and it corresponds to the center of the naval chokepoint. Because the point  $(20, 20)$  lies on the circumference of

coverage for an optimally placed sensor, there is the minimum possible correction to “post-process” this grid to the acceptable coordinate: (20, 20.1). Given that there are an odd number of UUVs available, the point (20, 20.1) must be one of the optimal UUV coordinates. This point is argued more completely in the paragraph below through the use of a counter-example.

Imagine that there is only one UUV available and consider the optimal placement of this one UUV. The value score corresponding to the weighted interception arc makes the center of the arc the most “valuable.” Furthermore, this weighting is symmetric about the center of the interception arc at precisely the point (20, 20). Imagine that the one UUV was placed at any other point marginally to the left or right of (20, 20) on the lower envelope of the sensor system. In this case, the model would fail to cover some of the highly-weighted length of the interception arc on one side of the point (20, 20) for each corresponding length of lower-weighted arc it would gain on the opposite side of the point (20, 20). The only possible exception would be if the proposed optimal point was not on the interception arc. In this case, the UUV would have to use some length of the UUV coverage radius just to move itself to the interception arc. This could result in less total coverage of the interception arc if another candidate point was located directly on the interception arc. However, this scenario cannot apply in this problem because the point (20, 20) is on the interception arc. This allows the UUV to use its full coverage radius to move laterally on the interception arc. The obvious conclusion is that the point (20, 20) must be the optimal point if there is only one UUV to place, and this point is refined to the acceptable coordinates (20, 20.1).

In the case of three UUVs, the algorithm places the first UUV at the point (20, 20.1) as described above. The model then exploits the symmetry of the problem in a similar manner as it did in the case of four UUVs. Previously, the algorithm optimally placed two UUVs on the left half of the interception arc. Now, the algorithm begins by calculating the coverage value on the left half of the interception arc corresponding to the UUV at (20, 20.1). The new task becomes optimally placing one UUV on the lower envelope of the left half of the sensor system, while considering the region already covered by the centrally located UUV. After optimally placing this second UUV (in addition to the centrally located UUV), the algorithm reflects the coordinate around the

line  $x = 20$  to find the corresponding optimal UUV placement on the right side of the sensor system.

The model uses the same basic cost function and dynamic programming algorithm as before, with the following exceptions:

- Since the first UUV is known to be at (20, 20.1), the algorithm calculates the coverage value of this point only once.
- The algorithm only has to test each candidate point once since it is placing only one UUV in addition to the UUV at (20, 20.1).

Unlike the optimization problem with four UUVs, the full length of the interception arc cannot be covered by three UUVs. However, the weighted value score ensures that the central areas of the interception arc are covered with a higher priority than the tails.

As the separation between candidate points decreases, the results improve. The discretization paradox described in the case of four UUVs no longer applies because there is no interaction between the varying placements of two UUVs. Instead, the dynamic programming algorithm only places one UUV, and the finer discretization allows for more detailed UUV placements that minimize overlapped coverage with the UUV at (20, 20.1).

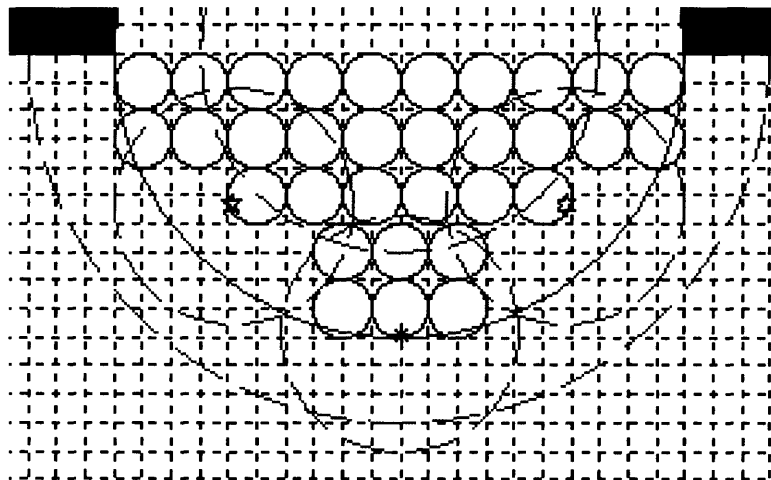
The overall optimal results for the problem with three UUVs are in Table 7.2 below, along with the originally assumed values and the distance between the assumed values and the calculated optima. As seen below, the initially assumed values are much closer to the optimal than was the case for the problem with four UUVs. In fact, each of the assumed values was less than one kilometer from the calculated optimal coordinates. Even so, the optimization process improved the placement coordinates for the three UUVs by a total of 2.7 kilometers.



	Optimal	Assumed	Total Difference (km)
<b>UUV 1</b>			<b>0.9</b>
x coordinate	14.1702	15	
y coordinate	24.6515	25	
<b>UUV 2</b>			<b>0.9</b>
x coordinate	25.8298	25	
y coordinate	24.6515	25	
<b>UUV 3</b>			<b>0.9</b>
x coordinate	20	20	
y coordinate	20.1	21	
<b>Total Difference (km)</b>			<b>2.7</b>

**Table 7.2 – Optimal Placement of Three UUVs & Originally Assumed Values**

Figure 7.9 below shows the optimal UUV locations for the case of three UUVs with respect to the sensor array. As before, the circles around the UUVs correspond to the UUV coverage radii given the highest possible threat vehicle speed. In this case, the dynamic programming solution fails to covers the full length of the interception arc with three UUVs. However, the weighting of the interception arc successfully ensured that the middle of the arc was covered with a higher priority than the ends of the arc. It is also noticeable that the optimization process solved the placement problem with almost no visible overlap in coverage.



**Figure 7.9 – Optimal UUV Locations with Three UUVs**

### 7.3.4.2 Two UUVs Remaining

The case of two UUVs is immediately parallel to the case of four UUVs described above. The only exception is that the dynamic programming algorithm is used to optimally place one UUV on the left half of the sensor array instead of placing two UUVs on the left half of the sensor array. After the dynamic program optimally places one UUV on the left half of the sensor array, the model employs the symmetry of the sensor system and reflect the results to the right side of the sensor array.

Like the optimization problem with three UUVs, the full length of the interception arc cannot be covered by two UUVs. Again, the weighted value score ensures that the central areas of the interception arc are covered with a higher priority than the tails. As with three UUVs, the problem with two UUVs did not include the discretization paradox because the algorithm was only placing one UUV on each half of the sensor system and there was no interaction between two varying points. Consequently, the results improved as the value of separation between candidate points decreased, and the algorithm minimized the overlapped area of coverage around the central point (20, 20).

The overall optimal results for the problem with two UUVs are in Table 7.3 below, along with the originally assumed values and the distance between the assumed values and the calculated optima. In this case, the initial assumptions for the case of two UUVs were extremely close to the calculated optima. The optimization process improved the placement coordinates for the two UUVs by a total of 198 meters.

	<b>Optimal</b>	<b>Assumed</b>	<b>Total Difference (km)</b>
<b>UUV 1</b>			<b>0.099</b>
x coordinate	17.1054	17.1	
y coordinate	22.9016	23	
<b>UUV 2</b>			<b>0.099</b>
x coordinate	22.8946	22.9	
y coordinate	22.9016	23	
<b>Total Difference (km)</b>			<b>0.198</b>

**Table 7.3 – Optimal Placement of Two UUVs & Originally Assumed Values**

Figure 7.10 below shows the optimal UUV locations for the case of two UUVs with respect to the sensor array. Again, the circles around the UUVs correspond to the UUV coverage radii given the highest possible threat vehicle speed. As with three UUVs, the dynamic programming solution fails to covers the full length of the interception arc with two UUVs. Again, the weighting of the interception arc successfully ensures that the middle of the arc is covered with a higher priority than the ends of the arc, and the optimization process solved the placement problem with almost no visible overlap in coverage.

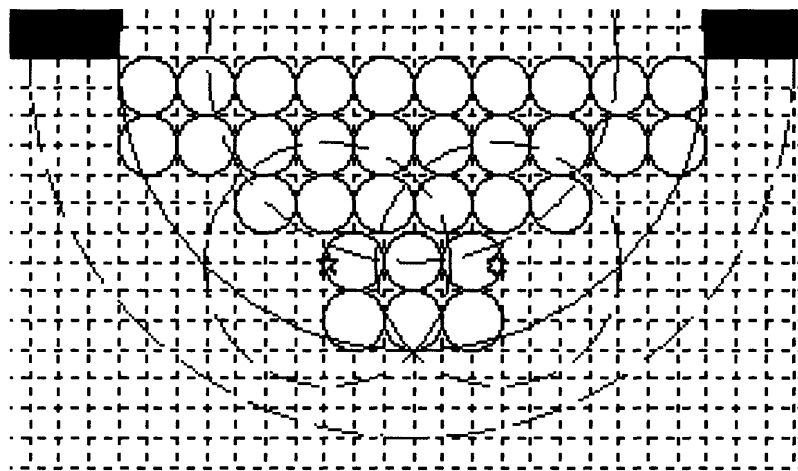


Figure 7.10 – Optimal UUV Locations with Two UUVs

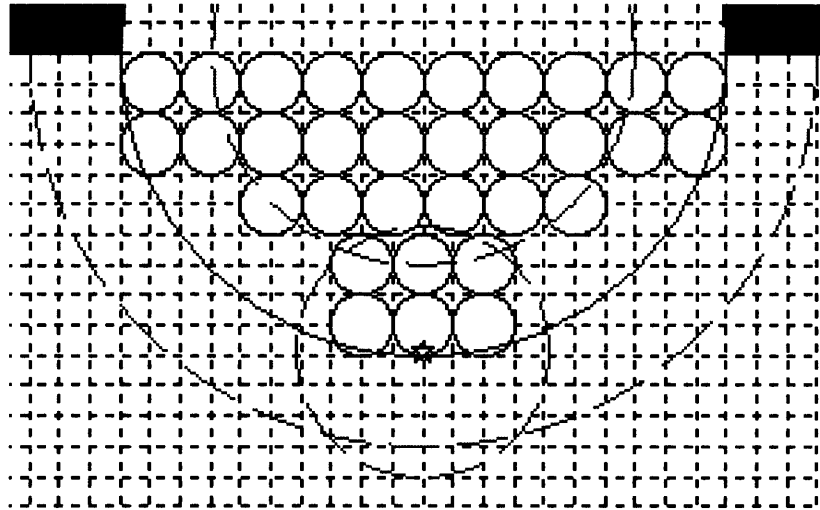
### 7.3.4.3 One UUV Remaining

The logic behind the case of one UUV remaining has already been explained in Section 7.3.4.1. If only one UUV remains, the optimal placement is at the coordinates (20, 20.1). Table 7.4 below shows that the difference between the optimal point and the assumed value was 900 meters.

	Optimal	Assumed	Total Difference (km)
<b>UUV 1</b>			<b>0.9</b>
x coordinate	20	20	
y coordinate	20.1	21	
<b>Total Difference (km)</b>			<b>0.9</b>

Table 7.4 – Optimal UUV Placement and Assumed Values for One UUV

Figure 7.11 below shows the optimal UUV location for one UUV with respect to the sensor array. As expected, the weighting of the interception arc successfully ensured that the center of the arc was covered with a higher priority than the ends of the arc.



**Figure 7.11 – Optimal UUV Location with One UUV**

## Chapter 8 Results, Analysis, and Alternate Strategies

The purpose of this chapter is to provide the results for the base-case simulation and for each of the simulation modifications described in Chapters 5-7. As expected, the reassignment algorithm in Chapter 5 and the optimization of parameters in Chapters 6 and 7 improved the algorithm's performance. This chapter also proposes alternative modeling methodologies for the problem considered in this thesis. These alternatives propose using the modeling tools of dynamic programming as the basis for the entire interception algorithm.

### 8.1 Numerical Results

The base-case simulation allowed for no reassignments and used assumed values for the radius of the interception arc, the initial UUV locations, and the UUV coverage locations. The methodology of Chapter 5 developed a reassignment algorithm by calculating confidence regions around  $(x_r, y_r)$ . The radius of the interception arc was optimized in Chapter 6, concluding that the optimal radius was 10 kilometers instead of the assumed value of 11 kilometers. The modeling and analysis in Chapter 7 used dynamic programming to calculate the optimal initial UUV locations and UUV coverage coordinates.

The simulation executed 10,000 stochastic runs for each of these scenarios to determine their effectiveness. Using the  $\pm 500$  meter threshold as the definition of success, the results are in Table 8.1 below.

Scenario	Success	Failure	Reassignment Used	If Reassignment Used	
				Success	Failure
<b>BASE CASE</b> No Reassignment Allowed Assumed Arc Radius (11 km) Assumed UUV Locations	74.09%	25.91%	0.00%	N/A	N/A
<b>CHAPTER 5</b> Reassignment Allowed Assumed Arc Radius (11 km) Assumed UUV Locations	86.28%	13.72%	14.80%	82.36%	17.64%
<b>CHAPTER 6</b> Reassignment Allowed Optimal Arc Radius (10 km) Assumed UUV Locations	89.01%	10.99%	16.07%	85.51%	14.49%
<b>CHAPTER 7</b> Reassignment Allowed Optimal Arc Radius (10 km) Optimal UUV Locations	92.29%	7.71%	21.13%	84.15%	15.85%

**Table 8.1 – Simulation Results**

## **8.2 Analysis of Results**

Each improvement or optimization dramatically increased the simulation’s success rate. In particular, the inclusion of a reassignment algorithm decreased the probability of failure by 47% compared to the base-case scenario. Optimizing the radius of the interception arc decreased the probability of failure by an additional 19.9% as compared to the scenario with only the reassignment algorithm. By optimizing the UUV locations, we decrease the probability of failure by another 29.8% as compared to the data with the optimal radius but the assumed locations for the UUVs. Combining the inclusion of the reassignment algorithm, the optimized arc radius, and the optimized UUV locations, the final version of the complete algorithm decreased the probability of failure by a remarkable 70.2% as compared to the base-case scenario.

It is also interesting to notice that as the system parameters are optimized, the algorithm tends to require the reassignment algorithm more often. When the radius of the interception arc is optimized, the algorithm requires the model to react more quickly to threats because the radius has decreased from 11 kilometers to 10 kilometers. The effect

of optimizing the UUV locations is that there is a thinly spread coverage over the entire interception arc instead of clumped coverage in the center. By spreading out the UUVs, the gap between the two UUVs on each half of the sensor system is enlarged. If a threat vehicle moves towards this gap, any change in its disposition will be likely to motivate the reassignment algorithm. In essence, by optimizing the resources, the algorithm has strained them to their limits. Any change in disposition by the threat vessel is now more likely to motivate a reassignment. Consequently, as the system parameters are optimized, the applied use of confidence regions becomes even more important to the success of the system through the reassignment algorithm.

Even though the model tends to require the reassignment algorithm more often as it optimizes the system parameters, the percentage of the reassignment runs that result in success remains fairly consistent at about 84%. Most of the cases when a reassignment still results in failure occur because the threat vehicle changes its disposition (speed or bearing) multiple times. When this happens, the algorithm may attempt to reassign the chaser UUV each time that the threat vessel changes its disposition. This can result in small gaps in coverage as the unassigned UUVs move towards coverage positions, but are then reassigned as chaser vehicles. In most unsuccessful iterations of this type, the chasing UUV still gets very close to the threat vehicle after multiple reassignments (within 600-700 meters) but fails to achieve the 500 meter threshold that defines a successful interception.

The algorithm already succeeds in intercepting the threat vehicle more than 92% of the time. However, two additional factors suggest that the “real-world” success rate of the algorithm may be even higher. These are described in the paragraphs below.

Most of the failures in the final version of the model occurred when the stochastics that govern threat vehicle movement allowed for iterations in which the threat vehicle dramatically changed speed or bearing multiple times. The distribution assumptions used in this research allow iterations of this type with very low probability to stress the effectiveness of the model, but this behavior would be even more unlikely in a real-world scenario.

Secondly, the use of  $\pm 500$  meters as the threshold for a successful interception is probably overly conservative. This assumption was based on the known range of archaic sonar systems that the UUV could potentially use. Given the rapid rate of advancement in underwater acoustic systems, this range would likely increase by several hundred meters. This would allow the UUV to achieve “successful interception” in most of the cases of multiple reassignments when the chasing UUV ultimately missed the threat vehicle by 500-800 meters.

However, the success rate in a real-world scenario will also be affected by numerous additional factors that the algorithm does not currently model such as terrain features, obstacles, and intelligent adversaries. Chapter 9 will consider some of these factors as areas for future research.

### ***8.3 Alternate Modeling Methodology – Imperfect State Information and Certainty Equivalent Control***

Chapter 7 introduced the basic concepts of dynamic programming and demonstrated how dynamic programming could be used to determine the optimal initial coordinates for the UUVs in this simulation. In addition to finding optimal parameter values, the principles of dynamic programming could also be used as a modeling framework for the entire simulation. This section will briefly address this possibility using the principles of imperfect state information and certainty equivalent control. These methods provide alternate modeling strategies that could produce very similar output to the results described in Section 8.1. The purpose of this section is to offer a brief perspective into the power of these two dynamic programming methodologies that could be useful for problems of this type. Throughout Section 8.3, Bertsekas’ textbook *Dynamic Programming and Optimal Control, Volume I* [4] and the corresponding lecture notes from his course in Dynamic Programming at the Massachusetts Institute of Technology [5] serve as the primary references.



### 8.3.1 Imperfect State Information

Previously, the perspective of this thesis was that the sensor / UUV system had “perfect information” of the threat vessel’s disposition as long as the threat vessel was within the sensing radius of one of the sensors during one of the discrete updates that occurred at 60 second intervals throughout the simulation. This modeling methodology approximated the continuous movement of the threat vessel and UUVs by the discrete updates that occurred each minute.

Using the imperfect state information model from dynamic programming, the perspective changes somewhat. Instead of approximating all the disposition stochastics at discrete updates, the new perspective is that movements and changes in disposition occur in continuous time. The model attempts to gain its information of threat vehicle activity through a series of snapshot observations that occur at discrete, one minute intervals. Furthermore, the observations may include some observation noise ( $v_k$ ) characterized by an assumed probability distribution. As before, if the threat vehicle is within the sensing radius of one of the sensors at the time of the snapshot, then the model uses this information to calculate speed and bearing information, calculate  $\tau$  and/or  $(x_\tau, y_\tau)$ , assign a chaser UUV, consider reassignment of UUVs, and calculate control information for the UUVs.

The simulation state space for this problem includes the  $x$  and  $y$  coordinates for each of the four UUVs, the threat vehicle’s  $x$  and  $y$  coordinates from the last visible observation,  $\tau$ , the last calculated values for threat vehicle speed and bearing, and the desired interception coordinate  $(x_\tau, y_\tau)$ . Since the state space is identified by the notation  $x_k$ , the  $x$  coordinates of the UUV and threat vehicle positions will be identified by the letter  $b$ . In this manner, the coordinate  $(b_1, y_1)$  will correspond to the current  $x$  and  $y$  coordinates of UUV number 1, with parallel notation for UUVs 2, 3, and 4 and the coordinates of the threat vessel. The variables  $x_\tau$  and  $y_\tau$  will continue to be used as the  $x$  and  $y$  coordinates for the point of desired interception in order to maintain consistency throughout the thesis. The definition of the state space is below, where  $k$  indexes time.

$$x_k = (b_1, y_1, b_2, y_2, b_3, y_3, b_4, y_4, b_{THREAT}, y_{THREAT}, \tau, speed, bearing, x_\tau, y_\tau) \quad (8.1)$$

The initial state  $x_0$  is known for the  $b$  and  $y$  coordinates of the UUVs, based on the optimization output in Chapter 7. The  $b$  and  $y$  coordinates, speed, and bearing of the threat vessel in the initial state  $x_0$  are realizations of the random variables described by the probability distributions in Chapter 3.

The control variables in this problem include which UUV is assigned as the chaser UUV, and the  $b$  and  $y$  coordinates that each of the four UUVs should move towards. As before, the  $x$  coordinates of the destination points for each UUV are shown with the letter  $b$  to maintain consistency with the notation of the state space.

$$u_k = (chaser, b_1, y_1, b_2, y_2, b_3, y_3, b_4, y_4) \quad (8.2)$$

The destination coordinates for the chaser UUV will be the point  $(x_\tau, y_\tau)$ . The other UUVs will have destination coordinates according to the optimal coverage positions calculated in Chapter 7. In this problem, the control variable *chaser* must take a value from the set  $chaser \in \{0,1,2,3,4\}$ , where  $chaser = 0$  corresponds to the event that no UUV has been assigned yet ( $k < 2$ ). The remaining control variables must take coordinate values from the acceptable ranges as defined by the area of operations in Chapter 2.

By the design of the algorithm, only one UUV is assigned as the chaser at any given time  $k$ , and the assignment of the chaser UUV is based on the closest UUV to the point  $(x_\tau, y_\tau)$ . This methodology would consider reassignment of UUVs at every discrete update by recalculating the control variable *chaser* during each time step. A more sophisticated reassignment algorithm like the one described in Chapter 5 would be possible after augmenting the state space with the confidence interval data around  $(x_\tau, y_\tau)$ . However, this would increase the state space by 72 terms (the  $x$  and  $y$  coordinates of both endpoints for each of the two confidence intervals around  $(x_\tau, y_\tau)$ , for each of the 9,  $\psi\%$  scoring regions considered.) This added dimensionality in the state space would dramatically reduce the model's ability to calculate optimal solutions because of the "curse of dimensionality" as described in Section 7.3.3.2. Consequently,

the proposed formulation in this section would recalculate the optimal chaser UUV at each time step based on the closest UUV to the updated coordinates  $(x_\tau, y_\tau)$ .

This formulation also allows for a disturbance  $w_k$  with an assumed probability distribution that may depend explicitly on  $x_k$  and  $u_k$ , but not on the previous disturbances  $w_0, \dots, w_{k-1}, v_0, \dots, v_{k-1}$  [4]. In this formulation,  $w_k$  would incorporate both the navigational uncertainty in UUV locations and the threat vessel disposition uncertainty.

This formulation assumes that the locations of the 4 UUVs are always known by the sensor / UUV system  $\pm w_k$ . The model gains information about the location of the threat vehicle through a series of observations  $(z_k)$ . The observations  $(z_k)$  correspond to information recorded by the sensor system during each attempted measurement update (every 1 minute). The importance of the snapshot observations  $(z_k)$  is to attempt to locate the threat vessel at time  $k$ , and to use this location to calculate threat vessel speed, bearing, and  $(x_\tau, y_\tau)$  using functions of the type described in Chapter 4. After rounding  $\tau$  to the nearest whole number, the observations  $(z_k)$  take the form described below [4].

$$z_0 = h_0(x_0, v_0), \quad z_k = h_k(x_k, u_{k-1}, v_k), \quad k = 1, 2, \dots, \tau - 1, \tau, \tau + 1, \dots, \tau + 10 \quad (8.3)$$

The model defines the finite horizon  $N$  as the quantity  $\tau + 10$  because the value of  $\tau$  is calculated after the second visible time step and the stochastics governing threat vehicle movement may cause actual interception to occur at a time slightly later than  $\tau$  time steps. Based on experience with the model, if successful interception occurs at all, it will occur after no more than  $\tau + 10$  time steps.

The imperfect state information formulation includes an information vector  $(I_k)$  that contains all the information that is available at time  $k$  [4].

$$I_0 = z_0, \quad I_k = (z_0, z_1, \dots, z_k, u_0, u_1, \dots, u_{k-1}), \quad k = 1, 2, \dots, \tau + 9 \quad (8.4)$$

Because of the high dimension of the information vector in this formulation, there would ideally be some sufficient statistic of smaller dimension than  $I_k$ , that summarized all the

important content of  $I_k$  for control calculations. If applied in a simulation, the information vector would be a storage matrix whose data would help influence future control decisions. Specifically, each function  $u_k$  maps  $I_k$  into the acceptable values from the control space  $U_k$  such that the expression below holds [4].

$$u_k(I_k) \in U_k, \quad \forall I_k, \quad k = 0, 1, \dots, \tau + 9 \quad (8.5)$$

Policies comprised of functions consistent with Equation 8.5 are considered admissible. Given this understanding of admissibility, the system equation for the state variables is below [4],

$$x_{k+1} = f_k(x_k, u_k(I_k), w_k), \quad k = 0, 1, \dots, \tau + 9 \quad (8.6)$$

where the functions  $f_k(x_k, u_k(I_k), w_k)$  are consistent with the movement algorithm calculations in Chapter 4. Specifically, the functions that update the UUV positions in  $x_{k+1}$  are consistent with the movement calculations in Section 4.10, and the functions that update the threat vessel disposition and  $(x_\tau, y_\tau)$  are consistent with the calculations described in Sections 4.4 and 4.5. The  $b$  and  $y$  coordinates of the threat vessel in  $x_{k+1}$  come directly from the snapshot observation  $z_k \pm v_k$ .

In a similar manner, the expression for the observations ( $z_k$ ) in Equation 8.3 take the new form below that considers the information vector [4].

$$z_0 = h_0(x_0, v_0), \quad z_k = h_k(x_k, u_{k-1}(I_{k-1}), v_k), \quad k = 1, 2, \dots, \tau - 1, \tau, \tau + 1, \dots, \tau + 9 \quad (8.7)$$

The cost function in this formulation is designed to reward a policy that arrives at  $(x_\tau, y_\tau)$  as quickly as possible. Previously, Section 4.10.1 considered the question of whether or not it was optimal for the assigned UUV to travel towards  $(x_\tau, y_\tau)$  at the maximum speed. The conclusion was in the affirmative, because any deviations in threat vessel disposition would be centered around the previous point  $(x_\tau, y_\tau)$ , and the UUV could respond best to these potential changes in disposition if it had already reached the previous point  $(x_\tau, y_\tau)$ . The argument of this section is parallel to the one in

Section 4.10.1. Consequently, it is desirable for the assigned UUV to arrive at  $(x_\tau, y_\tau)$  as quickly as possible. The cost function is designed to penalize the system by 1 point for each minute (each time step  $k$ ) that it takes for the assigned UUV to arrive at  $(x_\tau, y_\tau)$ , and the objective is to minimize the cost function. Since time is linear, this provides an additive cost function structure, consistent with fundamental dynamic programming principles.

It may seem intuitive to reward a successful interception by some large negative value. However, this would be an inappropriate metric because of the situation that may occur when the threat vessel changes its disposition after leaving the sensor array. In this case, the dynamic program may perform optimally, but the UUV could still fail to intercept the threat vessel based on threat vessel stochastics. Consequently, the cost function will be limited to penalizing each minute that the assigned UUV takes to arrive at  $(x_\tau, y_\tau)$  by 1 point. After the assigned UUV has reached  $(x_\tau, y_\tau)$ , the cost per stage (per time step  $k$ ) is 0 unless the point  $(x_\tau, y_\tau)$  changes because of the threat vehicle stochastics. If  $(x_\tau, y_\tau)$  changes, then the model again imposes a penalty of 1 point per time step  $k$  until the assigned UUV has reached the new point  $(x_\tau, y_\tau)$ . This cost system is described by the functions  $g_k(x_k, u_k(I_k), w_k)$ , where

$$g_k(x_k, u_k(I_k), w_k) = \begin{cases} 0, & \text{if } E_{w_k, v_k} [x_{k+1}(b_{CHASER\_UUV}, y_{CHASER\_UUV})] = (x_\tau, y_\tau) \\ 1, & \text{otherwise} \end{cases} \quad (8.8)$$

The objective of this formulation is to find an admissible policy  $(\pi)$  that minimizes the cost function below [4].

$$J_\pi = E_{x_0, w_k, v_k} \left\{ \sum_{k=0}^{\tau+10} g_k(x_k, u_k(I_k), w_k) \right\}, \quad k = 0, 1, \dots, \tau + 10 \quad (8.9)$$

Notice that there is no distinction in Equation 8.9 between the terminal costs  $g_{\tau+10}$  and the regular stage costs  $g_k$  because their form is identical.

There is a natural extension of this methodology discussed in [4] and [5] that reformulates the imperfect state information problem as a perfect state information

problem. Rather than address this reformulation directly, this thesis will consider a similar methodology in the section below that combines the formulations in Section 8.3.1 with the principle of certainty equivalence.

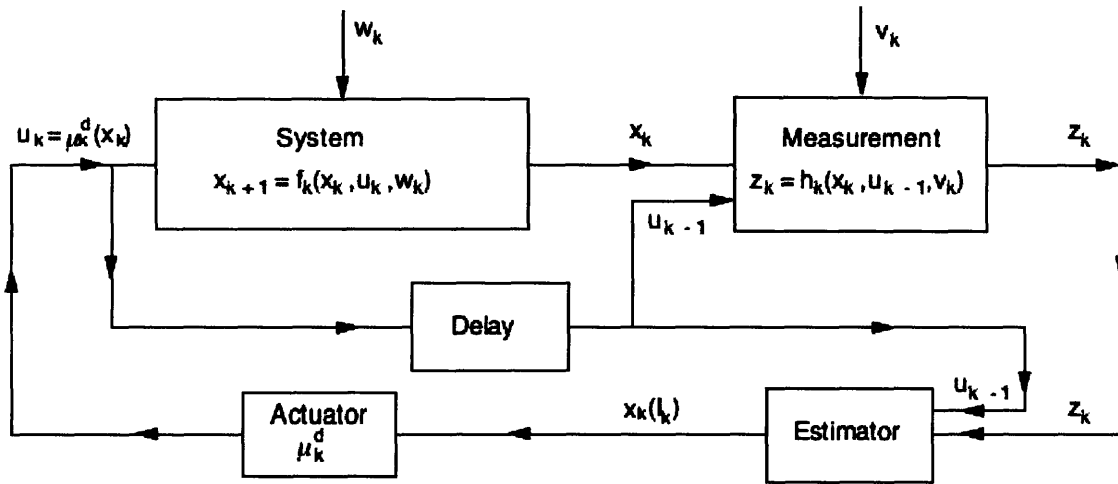
### 8.3.2 Certainty Equivalent Control

If the distributions of the underlying random processes are known, one technique is to model the dynamic program using the expected values instead of the random variables. Barto and Sutton call this the certainty equivalent estimate because it is equivalent to assuming that the estimates of the underlying processes are known with certainty [2]. Bertsekas further explains that certainty equivalence holds in dynamic programming if the optimal policy is unaffected when the disturbances are replaced by their means [4]. This technique seems particularly relevant for the problem considered in this thesis because all of the assumed stochastic distributions are symmetric about their means with well-defined expected values.

In this formulation, it is also reasonable to assume that the distributions of the noise parameters ( $w_k$  and  $v_k$ ) are symmetric with zero mean. This relies on the symmetric shapes of the confidence intervals around  $(x_\tau, y_\tau)$  described in Chapter 5, and assumes that both the measurement noise from the observations ( $z_k$ ) and the navigational noise from the reported UUV locations will be symmetric with zero mean.

The certainty equivalent controller assumes the existence of an estimator that uses the information vector ( $I_k$ ) to produce a typical value of the state  $\bar{x}_k(I_k)$ , and that there is a typical value of the disturbance ( $w_k$ ) for every pair  $(x_k, u_k)$  [4]. In this problem, it is valid to use the expected values of the distributions for the calculations of  $\bar{x}_k(I_k)$ , and the expected value of  $w_k$  is assumed to be zero for all pairs  $(x_k, u_k)$ . Given the new structure of this problem, an optimal controller of the form shown in Figure 8.1 below may be used to determine the control input  $\tilde{u}_k(I_k)$  applied by the certainty equivalent controller at time  $k$  [5]. The process is parallel to the imperfect state information problem

described earlier, and a detailed explanation of Figure 8.1 with respect to the problem in this thesis follows the diagram.



**Figure 8.1 – Certainty Equivalent Controller Implemented in Feedback Form (from [5])**

In this research, the UUV / Sensor system learns of the threat vessel through measurements that occur every 60 seconds. This corresponds to the *Measurement* box in Figure 8.1. The measurements are subject to noise ( $v_k$ ), but the model assumes that the noise parameter has zero mean. At each time step  $k$ , the *Estimator* considers the most recent measurement, estimates the threat vessel's updated speed and bearing, and recalculates  $(x_r, y_r)$ . If the threat vessel is not located within a sensor radius during the particular time step  $k$ , then there is no new measurement and the *Estimator* acts on the old information. In Figure 8.1, this possibility is shown through the *Delay* box. The calculations that yield the Estimator's output are consistent with the calculations described in Sections 4.4 and 4.5. The *Actuator* determines the new control variables based on the *Estimator's* output. This process determines the assigned UUV and calculates the coordinates that each UUV should move towards during that time step. These calculations are consistent with the UUV movement calculations in Section 4.10. Finally, the system is updated through the information of UUV locations  $\pm w_k$  and the calculated updates of threat vessel disposition that occurred in the *Estimator* box. The system update process is further simplified since  $w_k$  is assumed to have zero mean. The

process is repeated every 60 seconds when the system attempts to locate the threat vehicle through another sensor measurement.

Given the cost function structure described in the previous section on imperfect state information, the objective of the Certainty Equivalent Control formulation becomes minimizing Equation 8.10 below,

$$\sum_{k=0}^{\tau+10} g_k(x_k, u_k(x_k)), \quad k = 0, 1, \dots, \tau + 10 \quad (8.10)$$

subject to the system equations

$$x_{k+1} = f_k(x_k, u_k(x_k)), \quad u_k(x_k) \in U_k(x_k), \quad k \geq 0 \quad (8.11)$$

where the functions  $f_k(\cdot)$  control system behavior in the manner described in Chapter 4 and the terms  $(w_k)$  are omitted because they are assumed to have zero mean [4]. This process determines a control policy comprised of feedback controllers  $(u_k^d)$  that are optimal for the corresponding deterministic problem at each time  $k = 0, 1, \dots, \tau + 10$  [5].

The deterministic equivalent problem from this section may be solved using a heuristic policy that minimizes the sum of approximations to the current stage cost and the optimal cost to go at each stage  $k$  [5].

The imperfect state information and certainty equivalent control methodologies provide two alternatives that model the entire simulation within the context of the dynamic programming framework. Additional dynamic programming techniques such as look-ahead policies in combination with cost-to-go estimators could also provide effective solutions strategies for this problem, and they should be considered for future research.



## **Chapter 9 Conclusion**

The purpose of this chapter is to provide a summary of the research in this thesis and to offer suggestions for future research that could enhance the performance of the interception algorithm.

### **9.1 Research Summary**

The objective of this research was to develop an interception algorithm for a system of UUVs and sensors with the task of finding and intercepting a threat vessel in an aquatic environment. This interception task is important as a fundamental sub-task to numerous stated objectives in the U.S. Navy's recently released UUV Master Plan [30].

The algorithm developed in Chapters 2 – 7 of this thesis provides a solution methodology for the autonomous vehicle interception problem with a successful interception rate exceeding 92%. This algorithm uses confidence interval information to motivate the reassignment algorithm and applies optimization principles and dynamic programming to optimize system parameters. The corresponding simulation offers a mechanism to test the sensitivity of the successful interception rate as the user changes assumptions or parameter values. In summary, this research directly supports the specific mission requirements of the U.S. Navy's UUV Master Plan by modeling and analyzing fundamental characteristics of the UUV interception problem.

This research proposed two theoretical contributions in support of interception algorithms by autonomous systems. First, the thesis introduced the concept of using a desired interception region that may differ from the fastest analytical interception point. This was manifested as the desired interception arc in this research. The use of a desired interception region allows for a human-in-the-loop interface that may take advantage of terrain or environmental factors. Barring human influence, the optimization process in Chapter 6 determined the optimal arc radius that balanced the competing constraints of the UUV / threat vehicle velocity differential and the sensing limitation of the sensor array. The second important contribution in this thesis was the applied use of confidence

intervals around the expected point of desired interception. By using the confidence intervals as a scoring metric for the probability of successful interception, the proposed reassignment algorithm in Chapter 5 allows for a UUV reassignment only in cases when the probability of successful interception becomes unacceptably low. This tends to conserve the limited energy stores in the previously unassigned UUVs while reserving reassignment for situations in which it is required for successful interception of the threat vessel.

## **9.2 Areas for Future Research**

There are several areas for future research that could improve this algorithm, and these areas should be considered in depth as potential modeling advancements before implementing the programming methodologies of this thesis on a working UUV. Three areas that are highlighted below include the addition of known terrain features, response requirements when confronted with unplanned obstacles, and the added complexity of an intelligent adversary.

### **9.2.1 Addition of Known Terrain Features**

The current model assumes an absence of terrain features apart from the land that defines the ends of the naval chokepoint. This was an intentional modeling choice in an effort to generalize the algorithm to many possible scenarios and geographical locations. This assumption may be entirely valid for many geographical areas. However, an aquatic region that includes a naval chokepoint of the type described in this thesis may also include other terrain features that could affect the algorithm. These could include small islands or a variety of sub-surface terrain features such as deep water shipping lanes, regions of very shallow water, or regions with particularly hazardous sub-surface features that should be avoided.

Terrain and bathymetric data of this type could be incorporated into the algorithm before it is used by UUVs for specific missions. This data would have three fundamental effects on the model. First, the assumed distributions for threat vehicle arrivals and behaviors would need to include the terrain features. For example, the presence of an island near the center of the chokepoint would make the current assumption of Gaussian arrivals centered on the middle of the chokepoint invalid. Instead, the model would need to be adaptive in such a way that the model could assume different distribution assumptions like Gaussian arrivals centered on the two “minor chokepoints” created by the island near the center of the larger chokepoint, with some probability distribution defining which of the two minor chokepoints was used. Similarly, the threat vehicle movement assumptions could be tailored to avoid dangerous regions or shallow water.

The second important effect of terrain features would involve the assignment and control of the UUVs. The current algorithm uses straight line distance to determine the assignment of the chaser UUV and the path planning for each of the UUVs. The inclusion of terrain features would require a more advanced path planner that calculated shortest path with the known obstacle data. Applications of network theory could potentially be useful in solving this more sophisticated problem.

The third important effect of terrain features would involve the optimization processes that determine the optimal radius for the interception arc and the optimal initial coordinates for the UUVs. Currently, these optimization strategies assume straight-line travel for the UUVs. Consequently, the algorithms would need to be adjusted to include the known obstacles. This could change the optimal values for the interception arc radius and/or the initial UUV coordinates.

### **9.2.2 Unplanned Obstacles**

Another important problem for the interception algorithm is the potential for unplanned obstacles. Regardless of how well the UUV control algorithm adjusts to the known terrain in a specific area of operations, there is always the potential for unplanned obstacles. These may include such things as mine fields, fishing nets, or large ships

temporarily blocking the “optimal” path. Given that these unplanned obstacles are either temporary problems (large ships / fishing nets) or well known by the threat vehicles in advance (mine fields), these examples suggest that the unplanned obstacles are primarily a concern with regard to the path planning algorithms for the UUVs.

Dynamic programming methodology may be a natural candidate for this problem because of its intrinsic design. The applied use of the imperfect state information algorithm from Section 8.3 may offer one successful methodology. Applications of rollout algorithms, limited look-ahead policies, or Q-learning may also provide useful modeling frameworks to address the dynamic path replanning problem [7].

### **9.2.3 Intelligent Adversary**

Throughout this thesis, the structure of the interception algorithm has assumed that the threat vehicles are unaware of the sensors and UUVs monitoring the naval chokepoint. This is probably an accurate assumption given the UUVs’ small size and quiet motors. However, it is reasonable to assume that the threat vehicles have the potential to learn of the chokepoint monitoring, either through direct sensing / discovery of the UUVs or through long-term analysis of the “friendly” response caused by threat vehicle activity in the vicinity of the chokepoint.

If the threat vessels have a reasonable likelihood of learning of the monitoring, it may be useful to employ some of the tools from game theory such as the min-max modeling framework. Additionally, an intelligent adversary in this scenario may have aspirations of capturing or destroying the UUVs as they accomplish their missions. This suggests that the UUV must be able to analyze the movements of the threat vehicle after successful interception to determine whether or not the threat vehicle has detected the UUV. If so, the UUV may require protocols that determine when it is appropriate to stay with the threat vessel and risk capture, and when it is best to break contact and hide.

## Appendix A

This spreadsheet (covering 3 pages) shows the relationship between the interception arc radius and the ability of the UUVs to cover the full length of the interception arc. Even when the threat vessel travels at maximum speed, 4 UUVs will always be able to cover the full length of the arc regardless of the arc radius.

Radius		Threat					
Radius (km)	(in meters)	E[vel] (knots)	E[vel] (m/s)	E[Tau]	max_vel (knots)	max_vel (m/s)	min_Tau
0	0	8	4.11552	0	12	6.173328	0
0.5	500	8	4.11552	2.0248555	12	6.173328	1.349893
1	1000	8	4.11552	4.049711	12	6.173328	2.699786
1.5	1500	8	4.11552	6.0745665	12	6.173328	4.04968
2	2000	8	4.11552	8.099422	12	6.173328	5.399573
2.5	2500	8	4.11552	10.124278	12	6.173328	6.749466
3	3000	8	4.11552	12.149133	12	6.173328	8.099359
3.5	3500	8	4.11552	14.173989	12	6.173328	9.449252
4	4000	8	4.11552	16.198844	12	6.173328	10.79915
4.5	4500	8	4.11552	18.2237	12	6.173328	12.14904
5	5000	8	4.11552	20.248555	12	6.173328	13.49893
5.5	5500	8	4.11552	22.273411	12	6.173328	14.84882
6	6000	8	4.11552	24.298266	12	6.173328	16.19872
6.5	6500	8	4.11552	26.323122	12	6.173328	17.54861
7	7000	8	4.11552	28.347977	12	6.173328	18.8985
7.5	7500	8	4.11552	30.372833	12	6.173328	20.2484
8	8000	8	4.11552	32.397688	12	6.173328	21.59829
8.5	8500	8	4.11552	34.422544	12	6.173328	22.94818
9	9000	8	4.11552	36.447399	12	6.173328	24.29808
9.5	9500	8	4.11552	38.472255	12	6.173328	25.64797
10	10000	8	4.11552	40.49711	12	6.173328	26.99786
10.5	10500	8	4.11552	42.521966	12	6.173328	28.34776
11	11000	8	4.11552	44.546821	12	6.173328	29.69765
11.5	11500	8	4.11552	46.571677	12	6.173328	31.04754
12	12000	8	4.11552	48.596532	12	6.173328	32.39744
12.5	12500	8	4.11552	50.621388	12	6.173328	33.74733
13	13000	8	4.11552	52.646243	12	6.173328	35.09722
13.5	13500	8	4.11552	54.671099	12	6.173328	36.44712
14	14000	8	4.11552	56.695954	12	6.173328	37.79701
14.5	14500	8	4.11552	58.72081	12	6.173328	39.1469
15	15000	8	4.11552	60.745665	12	6.173328	40.4968



**Arc Properties given Radius**

Length of Arc (m)	length / 4	length / 3	length / 2
0	0	0	0
1570.795	392.6988	523.5983	785.3975
3141.59	785.3975	1047.197	1570.795
4712.385	1178.096	1570.795	2356.193
6283.18	1570.795	2094.393	3141.59
7853.975	1963.494	2617.992	3926.988
9424.77	2356.193	3141.59	4712.385
10995.565	2748.891	3665.188	5497.783
12566.36	3141.59	4188.787	6283.18
14137.155	3534.289	4712.385	7068.578
15707.95	3926.988	5235.983	7853.975
17278.745	4319.686	5759.582	8639.373
18849.54	4712.385	6283.18	9424.77
20420.335	5105.084	6806.778	10210.17
21991.13	5497.783	7330.377	10995.57
23561.925	5890.481	7853.975	11780.96
25132.72	6283.18	8377.573	12566.36
26703.515	6675.879	8901.172	13351.76
28274.31	7068.578	9424.77	14137.16
29845.105	7461.276	9948.368	14922.55
31415.9	7853.975	10471.97	15707.95
32986.695	8246.674	10995.57	16493.35
34557.49	8639.373	11519.16	17278.75
36128.285	9032.071	12042.76	18064.14
37699.08	9424.77	12566.36	18849.54
39269.875	9817.469	13089.96	19634.94
40840.67	10210.17	13613.56	20420.34
42411.465	10602.87	14137.16	21205.73
43982.26	10995.57	14660.75	21991.13
45553.055	11388.26	15184.35	22776.53
47123.85	11780.96	15707.95	23561.93

## References

- [1] Alksne, R. Rapidly Deployable Systems (RDS) Underwater Acoustic Telemetry Trials Report. DSTO Aeronautical and Maritime Research Laboratory, Melbourne, Victoria, Australia. March 2000.
- [2] Barto, Andrew G. and Richard S. Sutton. *Reinforcement Learning*. A Bradford Book, The MIT Press, Cambridge, Massachusetts, London, England, 2000.
- [3] Bellman, Richard. Dynamic Programming. Princeton University Press, Princeton, New Jersey, 1957.
- [4] Bertsekas, Dimitri P. Dynamic Programming and Optimal Control, Volume I, 2<sup>nd</sup> Edition. Athena Scientific, Belmont Massachusetts, 2000.
- [5] Bertsekas, Dimitri P. Lecture Notes from *Dynamic Programming and Stochastic Control*, (Course 6.231). Massachusetts Institute of Technology, 2004.
- [6] Bertsekas, Dimitri P. and John N. Tsitsiklis. Introduction to Probability. Athena Scientific, Belmont, Massachusetts. 2002.
- [7] Bertsekas, Dimitri P. and John N. Tsitsiklis. Neuro-Dynamic Programming. Athena Scientific, Belmont Massachusetts, 1996.
- [8] Bertsimas, Dimitris and John N. Tsitsiklis. Introduction to Linear Optimization. Athena Scientific, Belmont, Massachusetts, 1997.
- [9] Boehring, Robert. "Pseudo-Random Numbers." 2004. Virginia Polytechnic Institute, Blacksburg, VA.  
<<http://filebox.vt.edu/users/rboehrin/Buffon/Pseudo-Random.htm>>
- [10] Callas, Jon. "Using and Creating Cryptographic-Quality Random Numbers." 3 June 1996. <<http://www.merrymeet.com/jon/usingrandom.html>>



- [11] Carter, Everett F., Jr. "Generating Gaussian Random Numbers." 2001. Taygeta Scientific Inc., Monterey CA. <<http://www.taygeta.com/random/gaussian.html>>.
- [12] Casella, George and Roger L. Berger. Statistical Inference, Second Edition. Duxbury Thomson Learning, 2002.
- [13] Coddington, Paul. "Random Number Generators." Northeast Parallel Architectures Center. April 22, 1995. Syracuse University. <<http://www.npac.syr.edu/projects/random>>.
- [14] Cox, D.R. and D.V. Hinkley. Theoretical Statistics. Chapman and Hall/CRC, Boca Raton, 1974.
- [15] Freund, John E. Mathematical Statistics, 5<sup>th</sup> Edition. Prentice Hall, Upper Saddle River, New Jersey, 1992.
- [16] Gomaa, Walid E. "A New Learning Technique for Planning in Cooperative Multi-Agent Systems." Masters Thesis, Alexandria University, Egypt, 2002.
- [17] Haahr, Mads. "Introduction to Randomness and Random Numbers." Random.org, June 1999 Department of Computer Science, Trinity College, Dublin, Ireland. <<http://www.random.org/essay.html>>.
- [18] Hellekalek, Peter. "pLab: Random Number Generators." pLab: Theory and Practice of Random Number Generation. 2004. Mathematics Department, University of Salzburg. <<http://random.mat.sbg.ac.at/generators/>>.
- [19] Kou, Samuel. Lecture Notes from *Statistical Inference*, (Course STAT 211). Harvard University, 2004.
- [20] LaPointe, Kenneth and Melissa St. Peter. "Swarming Mobile Searchers." Undersea Warfare Analysis Department, Newport Undersea Warfare Center. 2004 Joint Undersea Warfare Technology Spring Conference.
- [21] Larson, Richard C., and Amadeo R. Odoni. Urban Operations Research. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1981.

- [22] Martinoli, Alcherio. "Swarm Intelligence in Autonomous Collective Robotics: From Tools to the Analysis and Synthesis of Distributed Control Strategies." Ph.D. Thesis, Ecole Polytechnique Federale de Lausanne, 1999.
- [23] Montgomery, Douglas C. and George C. Runger. Applied Statistics and Probability for Engineers, 2<sup>nd</sup> Edition. John Wiley & Sons, Inc. New York, 1999.
- [24] Naeem, W., R. Sutton, and S.M. Ahmad. "Pure Pursuit Guidance and Model Predictive Control of an Autonomous Underwater Vehicle for Cable/Pipeline Tracking." Marine and Industrial Dynamic Analysis Research Group. The University of Plymouth, Plymouth UK.
- [25] Prins, Jack, et.al. "What are Confidence Intervals." NIST/SEMATECH e-Handbook of Statistical Methods. September 2, 2004. National Institute of Standards and Technology. <<http://www.itl.nist.gov/div898/handbook/>>.
- [26] Ross, Sheldon M. Simulation, 3<sup>rd</sup> Edition. Academic Press, San Diego, 2002.
- [27] Rust, John. "Using Randomization to Break the Curse of Dimensionality." Yale University, November 1996.
- [28] Walker, John. "HotBits: Genuine random numbers, generated by radioactive decay." 22 March 2005. Fourmilab Lab, Switzerland. <<http://www.fourmilab.ch/hotbits/>>.
- [29] MATLAB Reference Guide. Natick, MA: The Math Works, Inc. August 1992.
- [30] The Navy Unmanned Undersea Vehicle (UUV) Master Plan. November 9, 2004. <<http://www.chinfo.navy.mil/navpalib/technology/uuvmp.pdf>>.
- [31] "Statistics Toolbox." Mathworks.com Helpdesk. 2005. <<http://www.mathworks.com/access/helpdesk/help/toolbox/stats/>>.