October 1970

Man-Computer Problem Solving in Real-Time Naval Duels

John W. Carr III  
*University of Pennsylvania*

Noah S. Prywes  
*University of Pennsylvania*

Philip Bursky  
*University of Pennsylvania*

Stuart Linzer  
*University of Pennsylvania*

Bruce Lull  
*University of Pennsylvania*

Follow this and additional works at: [http://repository.upenn.edu/cis_reports](http://repository.upenn.edu/cis_reports)

Recommended Citation

Carr, John W. III; Prywes, Noah S.; Bursky, Philip; Linzer, Stuart; and Lull, Bruce, "Man-Computer Problem Solving in Real-Time Naval Duels" (1970). *Technical Reports (CIS)*. Paper 824.  
[http://repository.upenn.edu/cis_reports/824](http://repository.upenn.edu/cis_reports/824)


This paper is posted at ScholarlyCommons. [http://repository.upenn.edu/cis_reports/824](http://repository.upenn.edu/cis_reports/824)  
For more information, please contact repository@pobox.upenn.edu.
Man-Computer Problem Solving in Real-Time Naval Duels

Abstract
The development of a new Man-Computer Problem Solving Methodology to be widely and effectively applied by the Navy has been the objective of this Research Project. The basic hypothesis that has been examined is as follows. If an interactive system would be available by which a human problem solver could put together, easily and quickly, a simulation of the problem and quickly perform tests of various solutions, perform an evaluation and then further improve the solution, then large scale economies and improved effectiveness would result. The research reported here may be considered to having taken the empirical approach. An experimental environment was selected, namely a Naval War. An interactive problem solving computer system was designed for this environment. To obtain an indication of the effectiveness of the system required the solution of problems in human engineering, computational methods and strategy in the areas of tracking and navigation, sonar applications and processing, and weapon application. New real-time interactive systems were incorporated to simplify the evolution of new problem solving methodologies.

Comments

This technical report is available at ScholarlyCommons: http://repository.upenn.edu/cis_reports/824
Technical Report

MAN-COMPUTER PROBLEM SOLVING IN REAL-TIME NAVAL DUELS (U)

by

John W. Carr III
Noah S. Prywes, et. al.

October 1970

Prepared for the
Department of the Navy
Office of Naval Research
Arlington, Virginia 22217

under

ONR Contract No.: N00014-67-0216-0013
Research Task No.: NR276-007

This document is subject to special export controls
and each transmittal to foreign governments or foreign
nationals may be made with prior approval of the Office
of Naval Research (Code 462).

Moore School Report No.: 71-08
MAN-COMPUTER PROBLEM SOLVING IN REAL-TIME NAVAL DUELS (U)

Technical Report

John W. Carr III Philip Bursky
Noah S. Prywes Stuart Linzer
Bruce Lull

October 1970

Technical Report

Moore School Report No. 71-08

None

Office of Naval Research (Code 462)

Department of the Navy
Arlington, Virginia 22217

The development of a new Man-Computer Problem Solving Methodology to be widely and effectively applied by the Navy has been the objective of this Research Project. The basic hypothesis that has been examined is as follows. If an interactive system would be available by which a human problem solver could put together, easily and quickly, a simulation of the problem and quickly perform tests of various solutions, perform an evaluation and then further improve the solution, then large scale economies and improved effectiveness would result. The research reported here may be considered to having taken the empirical approach. An experimental environment was selected, namely a Naval War. An interactive problem solving computer system was designed for this environment. To obtain an indication of the effectiveness of the system required the solution of problems in human engineering, computational methods and strategy in the areas of tracking and navigation, sonar applications and processing, and weapon application. New real-time interactive systems were incorporated to simplify the evolution of new problem solving methodologies.
<table>
<thead>
<tr>
<th>KEY WORDS</th>
<th>LINK A</th>
<th>LINK B</th>
<th>LINK C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Man-computer problem solving methodology</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interactive Computer Facilities</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Environmental Simulation Programs</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Management System for Storage and Retrieval</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Command Facilities</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pattern Recognizer</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Naval Duel</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Multilist Executive Language</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Multilang Procedure Capability</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Navigation and Tracking</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Strategy Development</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Active Sonar Readings</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Passive Sonar Analysis</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Documented Game Playing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Decision Making</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Weapon Launching</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interception Calculations</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
MAN-COMPUTER PROBLEM SOLVING IN
REAL-TIME NAVAL DUELS (U)

Abstract

The development of a new Man-Computer Problem Solving Methodology to be widely and effectively applied by the Navy has been the objective of this Research Project. The basic hypothesis that has been examined is as follows. If an interactive system would be available by which a human problem solver could put together, easily and quickly, a simulation of the problem and quickly perform tests of various solutions, perform an evaluation and then further improve the solution, then large scale economies and improved effectiveness would result. The research reported here may be considered to having taken the empirical approach. An experimental environment was selected, namely a Naval War. An interactive problem solving computer system was designed for this environment. To obtain an indication of the effectiveness of the system required the solution of problems in human engineering, computational methods and strategy in the areas of tracking and navigation, sonar applications and processing, and weapon application. New real-time interactive systems were incorporated to simplify the evolution of new problem solving methodologies.
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0 INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Objectives of the Research Reported</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Special Credits</td>
<td>3</td>
</tr>
<tr>
<td>1.3 Recommendations</td>
<td>3</td>
</tr>
<tr>
<td>1.4 Components of a Problem Solving System</td>
<td>4</td>
</tr>
<tr>
<td>1.5 Summary and Conclusions</td>
<td>6</td>
</tr>
<tr>
<td>2.0 THE NAVAL PROBLEM SOLVING FACILITY</td>
<td>11</td>
</tr>
<tr>
<td>2.1 Introduction to the Naval Duel</td>
<td>11</td>
</tr>
<tr>
<td>2.2 The Multilist Executive Language</td>
<td>15</td>
</tr>
<tr>
<td>2.3 The Multilang Procedure Capability</td>
<td>16</td>
</tr>
<tr>
<td>3.0 NAVIGATION AND TRACKING</td>
<td>18</td>
</tr>
<tr>
<td>3.1 General</td>
<td>18</td>
</tr>
<tr>
<td>3.2 Human Engineering</td>
<td>18</td>
</tr>
<tr>
<td>3.3 Computational Methodology</td>
<td>28</td>
</tr>
<tr>
<td>3.4 Strategy Development</td>
<td>33</td>
</tr>
<tr>
<td>4.0 SONAR ANALYSIS AND PROCESSING</td>
<td>36</td>
</tr>
<tr>
<td>4.1 General</td>
<td>36</td>
</tr>
<tr>
<td>4.2 Active Sonar Estimates</td>
<td>38</td>
</tr>
<tr>
<td>4.3 Preprocessing Active Sonar Readings</td>
<td>42</td>
</tr>
<tr>
<td>4.4 Passive Sonar Analysis</td>
<td>44</td>
</tr>
<tr>
<td>5.0 WEAPON SYSTEMS</td>
<td>50</td>
</tr>
<tr>
<td>5.1 General</td>
<td>50</td>
</tr>
<tr>
<td>5.2 Launching a Weapon</td>
<td>51</td>
</tr>
<tr>
<td>5.2.1 General</td>
<td>51</td>
</tr>
<tr>
<td>5.2.2 Initial Model</td>
<td>51</td>
</tr>
<tr>
<td>5.2.3 Weapon Firing Using a Procedure</td>
<td>53</td>
</tr>
</tbody>
</table>
### TABLE OF CONTENTS (continued)

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.3 Testing and Evaluating Original Model</td>
<td>57</td>
</tr>
<tr>
<td>5.3.1 Decision Making</td>
<td>57</td>
</tr>
<tr>
<td>5.3.2 Weapon Launching</td>
<td>59</td>
</tr>
<tr>
<td>5.4 Confirmation of Weapon Firing</td>
<td>61</td>
</tr>
<tr>
<td>5.4.1 General</td>
<td>61</td>
</tr>
<tr>
<td>5.4.2 Original Model</td>
<td>61</td>
</tr>
<tr>
<td>5.4.3 Improvements in Weapon Confirmation</td>
<td>64</td>
</tr>
<tr>
<td>5.5 Weapon Status</td>
<td>66</td>
</tr>
<tr>
<td>5.6 Conclusions</td>
<td>67</td>
</tr>
<tr>
<td>6.0 A DOCUMENTED GAME</td>
<td>68</td>
</tr>
<tr>
<td>6.1 Game Initialization</td>
<td>68</td>
</tr>
<tr>
<td>6.2 The Game as Played by the Submarine Captain</td>
<td>74</td>
</tr>
<tr>
<td>6.3 The Game as Played by the Captain of Destroyer</td>
<td>89</td>
</tr>
<tr>
<td>6.4 Results of Game</td>
<td>109</td>
</tr>
<tr>
<td>APPENDIX A: Four Bearing Solution of Passive Sonar</td>
<td>112</td>
</tr>
<tr>
<td>APPENDIX B: Interception Calculations</td>
<td>124</td>
</tr>
<tr>
<td>APPENDIX C: Description of Game Functions</td>
<td>132</td>
</tr>
<tr>
<td>BIBLIOGRAPHY</td>
<td>155</td>
</tr>
</tbody>
</table>
1.0 INTRODUCTION

1.1 Objectives of the Research Reported

Research described in this report has been conducted under Contract NO0014-67-A-0216-0013 for the development of new man-computer problem solving methodology, which could be widely applied by the Navy.

The hypothesis that is examined in the research reported here is as follows. If an interactive system would be available by which a human problem solver could put together, easily and quickly, a simulation of the problem and quickly perform tests of various solutions, perform an evaluation and then further improve the solution, then large scale economies and improved effectiveness would result for two reasons:

a. the cost of implementing the simulation would be greatly reduced, but what is far more important,

b. the feedback cycle of

1) analysis-devide solution
2) test solution
3) evaluate-and-modify-solution

would be greatly shortened in time, and save major costs involved with real life testing.

Such systems are in existence, for instance, for chemical and electrical engineers engaged in design of chemical processing plants or electric circuits respectively. The problem posed to the researchers here was whether this approach could be applied widely in a large, diverse organization such as the Navy and in particular, whether the Navy could use this problem solving methodology for conducting operations, management, and engineering design.
It was recognized from the outset that a problem solving system for the Navy problem solver would have facilities beyond those available in a system for a chemical or electrical engineer. For instance, simulation of conflicts is required in aiding the work of the Navy weapon systems designer. Simulating a conflict implies several teams of antagonists participating interactively with the computer. Additionally, the complex problem of real time data acquisition and analysis must be handled by the problem solving system.

The research reported here may be considered to have taken the empirical approach. An environment was selected - that of a Naval war. An interactive problem solving computer system was designed for this environment. It is referred to in this report as a submarine vs. task force war game. The project staff then experimented with this system in the solution of problems to obtain an indication of the effectiveness of the system for problem solving. In particular, problems in:

a. human engineering  
b. computational methods  
c. strategy

have been solved. These problems have been in the three areas of

a. tracking and navigation - reported in Sect. 3,  
b. sonar applications and processing - reported in Sect. 4, and  
c. weapon application - reported in Sect. 5.

Section 6 of the report illustrates the operation of the problem solving system through the presentation of a scenario of a war game.
1.2 Special Credits

In the construction of this system, the project was highly indebted for many valuable suggestions to Cdr. Philip Charest, USN, then a graduate student in Computer and Information Sciences at the Moore School. Many of the strengths of the system are due to Cdr. Charest's ideas and suggestions. Without them the system would have been far less oriented to a realistic Naval situation. Nevertheless, design decisions were, of course, the complete responsibility of the Project Technical Staff.

In addition, other Naval officers assigned to the University Naval ROTC, and officers and civilians from ONR, Johnstown Naval Air Development Center, and elsewhere supplied pertinent comments during demonstrations.

1.3 Recommendations

In the course of the research, a Problem Solving Facility has been implemented which responds to the requirements of a naval systems designer or naval trainer. It includes a structure to simulate multi-antagonist conflicts. It is modular and has a command language and data management facilities that allow for easy and quick reorganization and modification of a simulation to suit specific problem solving situations. It is open-ended in that the user can readily add modules for the study of a particular interaction or features that he wishes to test. It allows simulation of roles of members of a team, or entire teams of participants.

While the results of our research have been only qualitatively evaluated, the rapid problem solving progress of our research staff using the problem solving system causes us to make the following recommendations to the Navy:

1. The man-computer problem solving methodology described in this report should have a strong impact on Naval problem solving effectiveness in commanding, conduct of operations, managing, and engineering.
2. The Navy should develop similar man-computer problem solving systems for several other broad Naval environments such as surveillance, various types of warfare, logistics, and intelligence.

3. The problem solving systems for the various environments should be disseminated throughout the Navy using widely available time sharing facilities. The ease and ready availability of these systems, training in their use, and their impact on effectiveness, should induce wide application of this methodology by the Navy.

1.4 Components of a Problem Solving System

When we refer in this report to a computer system used for problem solving, we mean a computer system interactively operating with human problem solvers, and having a program library available for the use of the human problem solver. In particular we imply the inclusion of the following components:

1. Interactive Computer Facilities - including keyboard, printer, and graphic terminals interacting on a real time basis with a major computer system [10]. In order to simulate properly the environment of conflicts, it is required that the problem solving system interact simultaneously with a number of human problem solvers who represent the various participants in the conflict. Additionally, many more people should be able to participate in an expansible and contractible system in which machine (program) modules may replace, or be replaced by, certain human participants on-line, interactively, and in real time. This permits both the augmentation of the number of users at any time and the machine simulation of any subsystem that might ordinarily be carried out by a human being.
2. Environmental Simulation Programs - As indicated above, these would simulate Naval environments such as surveillance, various types of warfare, logistics or intelligence, respectively [1,2,3,4]. Each one of these environments represents a broad spectrum of Naval activities. The environmental simulators must be highly modular and open-ended for ready and easy enhancements and modifications by users. A simulator must be completely independent of, but capable of interacting with, the data management system that handles all the data acquisition, storage and retrieval as described below.

3. A Data Management System for Storage and Retrieval - capable of recording and updating information on a real time basis during the conduct of the simulation [7,8,9]. For instance, the simulation of a sonar data acquisition system involves storage and updating of the acquired information on a real time basis. The Data Management System responds to storage and retrieval commands made in the language of the Data Management System. The commands are included in the simulating program modules.

4. Command facilities that call on the Environmental Simulation with arguments that specify the Data Management System's storage and retrieval functions [5,6]. For instance, in the simulation of a Naval duel, the human problem solver may direct the simulation of a weapon fired at targets that are computed from simultaneous data acquisition by sonar devices. The data acquisition is handled largely by the storage and retrieval component. The command language must have a hierarchy of commands to provide easy grouping of primitive commands under a higher level command, thus effecting additions to the system. It should also be open-ended to include additional commands that can be programmed in a variety of levels of programming languages.
5. A Pattern Recognizer - This component is still being studied. We find it is necessary interactively to identify various patterns which developed through the simulation or the man-machine interaction.

The combination of the above five components was found in our research to be necessary to give the Naval problem solver the ability of easily and quickly putting together a simulation of his problem. These requirements exceed considerably those inherent in a simulating system for a chemical or electrical engineer as mentioned above.

1.5 Summary and Conclusions

To investigate the new man-computer problem solving methodologies we had first to select an environment which was suitable for truly broad Naval application, in which we would perform our experiments in problem solving. The wide applicability of the methodologies that we have been exploring has always been a prime requisite of the research. We have selected a simulator (game) for submarine vs. task force engagement as being applicable to a large cross-section of Naval applications. For instance, it would be useful in the development of navigation, sonar and fire controls. It would be useful for development of sonar processing techniques. It would be useful for developing tactics and strategies for command of forces. Finally, it could be a very useful vehicle as a trainer.

Once an initial system had been developed we planned to engage in problem solving in this environment and then evaluate the effectiveness of this problem solving methodology. In fact, through the problem solving, the system itself became richer and more powerful as it incorporated in itself the progress made in problem solving.
We have assigned three of the staff of the project to the areas of:

1. navigation,

2. sonar application and processing,

3. weapon application.

They have been instructed to pay attention in particular to the human engineering of the interface between the operator and the weapon system, the computational methods required to process acquired data into meaningful information for the operator, and finally, to the developing of strategies.

A man-machine interactive model capable of simulating the maneuvers between two naval forces is operating within the Problem Solving Facility of the University of Pennsylvania. We have therefore been able to experiment with, develop and analyze, in this environmental simulator, problem solving requiring multi-dimensional communication between man and machines. Because of the modularity of the design, it has been easy to incorporate refinements and extensions. Our accomplishments in the three areas listed below illustrate the effectiveness of the Problem Solving Facility:

A. Human Engineering - the development of new techniques and languages for communication between man and machine in a real-time problem solving facility.

B. Computational Methodology - the incorporation of new computational aids within the system by adding new modules to the game.

C. Strategy Development - By conducting experiments with the game, new ideas for interactive problem solving have evolved.

The advances made in human engineering have emphasized the need for short and flexible input formats. In a problem solving environment where rapid decisions are made, it is essential that man-machine communication be made through a concise and simple language. The users have been given a
compact set of functions so that all requests for computer response may be expressed in a mnemonic form. New higher level commands may be developed by the user as he gains experience and finds the necessity for such additions.

Our commands have been made as English oriented as possible. Many parameters have been made optional, greatly shortening the commands. For example, such parameters as the number of sonar readings to be used in the next analysis and the type of weapons to be fired are assumed parameters which may be changed by the user. We have also provided the facility for the easy use of previously defined maneuvers which have become standard. To facilitate the use of our system, it is not even necessary to know the command language; English sentences may be used, and these are translated internally into the command language. Finally, results of computations are automatically stored and are readily available for future calculations or for output to the user in the form of displays or printout.

The utilization of a graphic display console has greatly extended our problem solving capabilities in many ways. We have developed a new display oriented language which almost totally eliminates the need for typing. Using a set of mnemonic instructions which is displayed on the screen, the user makes his requests to the computer by pointing at the screen with a light sensitive device and by pushing various buttons.

Instead of printed output, the user employs pictures as much as possible [3]. To analyze the present situation, he may obtain a display of his own sailing plan and his estimates of the opponents movements. He can then draw a new sailing plan using the old picture as a guide.

By displaying active sonar readings, it is possible to preprocess the readings at the console and then request further computation by the computer [4]. Thus, the commander interactively obtains an estimate of his
opponent's position, speed, and course.

A record of the solution may be saved in the form of pictures showing, for instance, the sailing histories of the ships, the estimates they made of the opponents' movements, and the weapons that were used. Thus, an analysis may be made at the display console at any future time.

In the area of computational methodology, we have developed a new passive sonar ranging technique which requires only a generalized maneuver. This technique combines two solutions over the same time interval in order to obtain a reliability factor for the result. We have modified other components of our simulation as we have learned their weaknesses. The navigation function now allows relative turns and the specification of maneuvers to be started in the future. The analysis of active sonar readings uses a fit which emphasizes no one point and which minimizes the error with respect to both distance and time. In the weapon model, we provide interception calculations for multi-speed weapons, such as ASROC.

The major decisions of how to maneuver and when to fire weapons are still completely made by the user. The computer does assist by analyzing sensor data but, once its analysis is done and results presented to the player, no more aid is given. By accumulating data from game to game and by analyzing standard tactics, a computer system would be able to answer such questions as:

1. Has there been a pattern to the opponent's track?
2. From the opponent's sailing history, what maneuvers is he most likely to execute now?
3. What would be my optimal evasion strategy in the present situation?
4. What attack and firing patterns have been successful in previous encounters?
Before the actual encounter, the user can enter various standard tactics into the computer system. The solution of each problem can also be added, as it is solved, to continually update the total amount of data available. Upon the request of the user, the computer system could then relate the present problem to previous similar situations and present, in graphical form, a resume of maneuvers used in the past together with their likelihood of success.

Experience with the game has shown that reasonable strategies may be developed even by inexperienced players. In this environment, a user is immediately confronted with the operational problems of a naval commander. Satisfactory maneuvers for passive sonar analysis must be developed. Attack patterns and evasion tactics can be tried, analyzed, modified and stored for future use. The total environment facilitates the development of new problem solving methodologies within the interactive man-machine framework.

The following five sections describe in detail our experience with the system and the new problem solving methodologies we have developed. We feel that the evolution of such methodologies has been greatly simplified through the use of our real-time interactive system.
2.0 THE NAVAL PROBLEM SOLVING FACILITY

2.1 Introduction to the Naval Duel

To investigate man-computer problem solving methodologies, an encounter between two opposing Naval vessels was simulated. The design of this environment can be divided into three main subgroups - navigation, sensing and estimating, and weapons control.

The ships must first have the ability to determine position and maneuver. The navigation subgroup consists of two parts. One is the simulation of the performance of a ship; the other is the programs which allow the player to make his requests. These parts are characteristic of all components of the game. There is always an internal simulation problem, and an external request capability.

Sensing and estimating includes the simulation of active and passive sonar devices and programs to help the player analyze this data. The weapons capability needs the simulation of each type of weapon and certain computational abilities, such as determining an interception course for a weapon, to aid the player in the selection of firing parameters.

The play of a game consists of a series of cycles between ship commander and computer. A cycle is a set of requests supplied by a player and the resulting printed or displayed output from the computer. The cycles of each player are independent. The results of previous cycles influence the content of the current one; the strength of this influence is a measure of the effectiveness of both the player and the problem solving facility. A sample cycle might include a maneuver based on information in a previous cycle, a request to make a new active sonar estimate, and a request to display this estimate with the current sailing plan of one's own ship. An attempt to fire a weapon might also be included. The computer executes each of the
above in succession and prints and displays the results.

In the original game the only interface between the man and machine was the teletype. Each player sat at a teletype which was connected to the main computer via a telephone line. Use of teletypes alone proved inflexible. The addition of a satellite display computer for one of the players has greatly influenced more recent work. At first the display was little more than a high speed printer. Then a function was developed which displays a picture containing items specified by the player. Usually the display has one's own ship's path and combinations of recent estimates made from active and passive sonar data. Weapon tracks can also be included. Alternately, active sonar points can be displayed to aid in visual pattern recognition and the player can select any portion of the readings displayed and make an estimate including just those readings. The display has assumed a more active role too. Most functions can be executed without any typing by using the light pen and pushbuttons alone. It is even possible to draw a new maneuver right on a picture of the latest estimate and path of the ship's sailing history.

Descriptions of the basic functions presently available for this environment are provided in Table 2-1. The reader will note the uniqueness of each function as well as the completeness of the list. Most of the building blocks necessary for a complete simulation are provided. A detailed discussion of these functions is given in Appendix C. One of the most important aspects of the research to date has been the ability of the system to grow with minimal strain and to grow not only in size and complexity but in power as well. This system is modular; the addition of new blocks is simple. More important perhaps is that changes in existing blocks are relatively easy to make if care has been taken in design. This building block
<table>
<thead>
<tr>
<th>Type of Function</th>
<th>Name</th>
<th>Functional Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Referee</strong></td>
<td>INTLZ</td>
<td>Initialize ships' characteristics</td>
</tr>
<tr>
<td></td>
<td>INTCL</td>
<td>Initialize and control clock</td>
</tr>
<tr>
<td></td>
<td>LAYER</td>
<td>Set thermal layer depth and sonar noise characteristics</td>
</tr>
<tr>
<td></td>
<td>START</td>
<td>Set initial position of a ship</td>
</tr>
<tr>
<td><strong>Player - Navigation</strong></td>
<td>SETSP</td>
<td>Control of speed, course and depth</td>
</tr>
<tr>
<td></td>
<td>POS</td>
<td>Print position</td>
</tr>
<tr>
<td></td>
<td>RANGE</td>
<td>Print range and bearing from own position to estimate</td>
</tr>
<tr>
<td></td>
<td>INTER</td>
<td>Calculate an interception course</td>
</tr>
<tr>
<td></td>
<td>OPINT</td>
<td>Calculate opponent's interception course</td>
</tr>
<tr>
<td></td>
<td>HISTORY</td>
<td>Display own path and specified estimates and weapon tracks</td>
</tr>
<tr>
<td><strong>Player - Sonar and Estimating</strong></td>
<td>USESN</td>
<td>Turn active and passive sonars on and off, control of frequency of readings</td>
</tr>
<tr>
<td></td>
<td>ESTPS</td>
<td>Make an estimate using passive sonar</td>
</tr>
<tr>
<td></td>
<td>ESTAC</td>
<td>Make an estimate using active sonar</td>
</tr>
<tr>
<td></td>
<td>DISPLAY</td>
<td>Display active sonar points</td>
</tr>
<tr>
<td></td>
<td>ELISA</td>
<td>Make an estimate based on display list interaction</td>
</tr>
<tr>
<td></td>
<td>ASCOM</td>
<td>Determine if a new estimate is needed</td>
</tr>
<tr>
<td><strong>Player - Weapons</strong></td>
<td>WEAPO</td>
<td>Fire a specified weapon</td>
</tr>
<tr>
<td></td>
<td>CNFRM</td>
<td>Print results of a weapon firing</td>
</tr>
<tr>
<td></td>
<td>WEPST</td>
<td>Print number of weapons remaining</td>
</tr>
<tr>
<td></td>
<td>WPARM</td>
<td>Sets parameters used in firing weapons</td>
</tr>
</tbody>
</table>

Table 2-1 - List of Game Functions
characteristic has proven to be of immense value. Several programs have been completely redesigned in the light of experience; in the case of the active sonar analyzer three different programs have been used at one time or another. Evolution is stimulated by flexibility and the ease of incorporating new and improved modules in the system.

To close this section a short summary of the flow of a game is presented.

The referee starts the clock and sets the initial positions and sailing plans of both ships. Then the players begin. Both would turn on passive sonar immediately. The surface vessel might also turn on active sonar. Next a request for a passive (or active) estimate would be made. If a response is noted then a maneuver may be warranted. Thus the first three cycles would be

1. turn on sonars
2. make an estimate
3. maneuver based on estimate.

The game continues with executions of cycles 2 and 3 until one of the players is in weapon range, whereupon he can fire if he so chooses. Other basic functions are executed as desired by the player. The encounter ends when one player is hit, runs out of weapons, evades, or gives up in frustration over his inability to damage the enemy. The goals of each player must be considered in evaluating the result if no hit was achieved.

The game itself is a specific environment. It has been found that it is a useful tool in determining characteristics which are desirable in any environment of even remotely similar needs.
2.2 The Multilist Executive Language

The naval war duel simulation is dependent on the Multilist information storage and retrieval system. This system is capable of manipulating data records and programs as well as storing and retrieving items satisfying certain constraints. This capability is used by the simulation to allow the user to retrieve previously defined or "built-in" functions when requested and to operate on all data records.

In order to communicate with the Multilist system, an executive language called Multilang [6] was designed. A request made of the Multilist system must be either a Multilang statement or a group of Multilang statements called a procedure.

A Multilang statement has the form:

```
LABEL OPERATION/OPERAND1/OPERAND2/.../OPERANDk
```

A statement may begin with a label - an octal integer of one to five digits - or it may have a blank first field. When present, the label serves to name the statement so that the program it designates can be called and executed by other programs.

The operation part of a Multilang statement consists of the name or a description of the program that the user wants to be executed.

All "Naval war duel simulation" commands described in this report are Multilang statements. When the war duel environment is in operation there is an interpreter in the computer which reads the Multilang statements being inputted, retrieves the program to be executed (see Appendix C for a list of these programs) and passes control to the designated entry point of the program. A method is provided by which the program in control can examine the operands in the statement and generate further instructions to the storage and retrieval system.
2.3 The Multilang Procedure Capability

The Multilist Executive Language (Multilang) has the capability of grouping several Multilang statements together into a procedure. A procedure provides for the execution of the programs sequentially unless a transfer of control is made to another section of the procedure. This structure allows the programmer to write a simple procedure in which each statement can be a complex program. This type of structure allows communication of data between programs without having to make disc accesses, allows sequential and non-sequential transfer of control between programs and by the use of formal parameters, has the capability of executing a sequence of programs with different operands on every call. The use of the procedure in the manner described above leads to the construction of higher level commands, where each command is a sequence of the basic commands described in this report. In this manner the command language is open ended and can grow to meet the needs of each individual user. In practice it has been noted that different users develop these higher level commands to aid in the solution of those subproblems which they consider to be most important. Even higher level languages may be created by forming procedures which have other procedures in their bodies. By the use of a procedure, the amount of information that need be inputted in order to execute several programs is minimal.

When a Multilang procedure is defined, the Multilang assembler converts the console statements into a compact internal format which is stored. A procedure is originally constructed with formal parameters and has actual parameters substituted at execution time.
The user requests a Multilang procedure in the same manner as he would request a Multilang statement. When the interpreter reads the request it brings in a copy of the procedure, substitutes actual operands for formal operands and given control to the first program in the procedure body.
3.0 NAVIGATION AND TRACKING

3.1 General

A Naval captain must have the ability to easily issue commands and obtain the latest information as it is received and analyzed by his subordinates. In our simulation of a Naval war duel, we have been constantly concerned with the problems of communication in a real-time environment. We have developed solutions for both display and printed media. This chapter summarizes the evolution of the sailing plan control program, called SETSP (SET Sailing Plan), which enables the user to modify the speed, course, or depth of his ship. Because of the essential function performed, SETSP was one of the first programs on our system; it is also one of the most frequently executed. This long and frequent use has emphasized the shortcomings of the program as it was originally implemented and has provided an impetus towards improvements.

Modifications have been made to simplify the interaction with the machine, to increase the power of the simulation and to enhance the user's ability to develop strategies. Our problem solving system has a modular design and so these varied improvements were easy to add to our model.

3.2 Human Engineering

Various improvements have been made to the command language in order to simplify the user's interaction with the machine. The navigation function, SETSP, exemplifies many of these changes which were made to reduce the number of circumstances under which the program could not function because of user errors and to minimize the time necessary for planning and typing.
One basic consideration in designing our system was to maximize the information available to a user without overloading him with useless details. This can be done quickly and in an understandable format using the display and this form of communication is used as much as possible. Sometimes however, a printed comment is the best response. The printout of SETSP was expanded from the simple line

```
THE SAILING PW HAS BEEN STARTED. TIME=XXXXXX
```

to a full listing of the sailing plan changes made by the request. This includes the speed, course (the word TURN is substituted in turns), depth and the time that each block starts. This insures that the player is fully aware of the results of the request; it also provides easy reference to the time that a particular maneuver began.

Besides using the display as a form of output, it can also be used as a reliable and quick method of entering commands. All standard requests can be listed before the user who then chooses one by pointing at it with the light pen. Such a format is shown in Figure 3-1 which shows the standard functions displayed on the bottom half of the screen. There is a second similar list which is employed to start or stop the execution of a series of programs, generate displays, save displays on magnetic tape, recall old displays, obtain output and a few other necessary system commands.

In the example shown in Figure 3-2, the destroyer captain has previously obtained a display of his own sailing history and three independent estimates of a detected submarine which have been projected into the future. By hitting the appropriate pushbutton and then pointing at the phrase

```
SET SAILING PLAN
```

as shown in Figure 3-1, a program is initiated which permits the destroyer
Figure 3-1
Using the Interactive Display Computer to Enter Commands into the System
Figure 3-2
A Display of the Destroyer and Three Estimates of the Submarine's Position
captain to draw or type (in this case, he chooses to draw) changes to his sailing plan. Using the present display as a reference, it is very convenient to draw a new sailing plan using the light pen. The DD captain decides to enter an attack maneuver. The speed of each leg of the maneuver is set by using a pushbutton and is shown in the upper left corner of the picture (Figure 3-3a). Below the "speed line" is the scale of this display and the time duration of the course presently being drawn. The speed, course and time duration of each leg can be altered until the appropriate pushbutton is hit which fixes that line on the screen. The total distance travelled and the course of each leg are shown next to the appropriate line. The final sailing plan is shown in Figure 3-4 and the result of its execution is shown below:

```
SETSP/
=20/=112/=35/
=17/=154/=37/
=09/=19?=16/
=09/=249/=9
SPEED   COURSE   START TIME
17.50   TURN     124639
20.00   112      124719
18.50   TURN     125139
17.00   154      125155
13.00   TURN     125839
9.000   192      125855
THE SAILING PLAN HAS BEEN STARTED.
```

A display depicting this attack maneuver and the estimate of the submarine upon which it was based is shown in Figure 3-5.

Similarly, the user may type his navigation commands. The original format of the SETSP call was as follows:

```
SETSP/(player's color)&OWN/=(speed)/=(course)/=(depth, if sub)
```
a) Initial point has been fixed  
b) One turn has been requested  
c) A second turn is drawn.

Figure 3-3
The Various Stages of Entering a Sailing Plan Using the Display Facilities
Figure 3-4
The Completed Sailing Plan as Entered by Captain of Destroyer Using the Light Pen
Figure 3-5
Display of Destroyer's Past and Future Track Together with an Estimate of the Submarine's Position
A quantity between two slashes is called a parameter; that part of a parameter within parentheses indicates that a substitution is to be made. The words RED and WHITE are used as the players' color or identifier. RED is usually used for the submarine and WHITE for the surface vessel; depth is of course not applicable for player WHITE. The parameter RED&OWN (or WHITE&OWN) specifies the name of the record containing the sailing plan to be modified; an improper or insufficient name would cause chaos if the programs had not been designed to check for this possibility. The original version used the first parameter directly to retrieve the record; later versions retrieve indirectly and automatically tack on '&OWN' to insure that an OWN record is retrieved. The older specification RED&OWN is still valid since (RED&OWN)&OWN is equivalent to (RED)&OWN.

A major advance was made when the ability to identify a user was added. The system was already aware of the identity of the user. Picking this up automatically eliminates the necessity of supplying a user identification. To maintain compatibility, the first parameter can be the identification; if so, it is checked to insure that it is correct. If it is, then the second parameter is expected to be a speed. If the first parameter is not a user identification, then it is expected to be a speed. If the identity supplied is incorrect (e.g. WHITE at player RED's station) then an error message is printed and execution is skipped. In later examples, the color will usually be included to identify the type of ship.

Another change that was made was to eliminate the requirement that an '=' precede all of the other parameters. The special character is required to tell the facility that a number is to be supplied to the program; the change was made so that the program would be able to recognize the input parameter as a number whether or not the equal sign is used. This addition was dictated
by the frequency with which players had to retype whole lines to add a forgotten '='.

These changes were made without affecting the basic logic of the program; they were made principally to simplify the input format. A typical request now looks like the following:

```
SETSP/(speed)/(course)/(depth, if sub)
```

This is effectively identical to the original format except that fewer characters need be typed; upwards compatibility is maintained so that any parameter which had been valid in the old version is still valid.

In the original version, all parameters except the identification (parameter 1) were limited to numbers, and all parameters had to be specified. It seemed desirable to expand the specifications to include some key words, particularly the word SAME. The ability to substitute this word for any speed, course or depth considerably expands the versatility of the program. In the past if a user desired to maintain a speed he had to know it; sometimes he lost valuable seconds looking back to check his current speed so he could maintain it. The addition of SAME means that he can just type SAME instead of the actual speed. Similarly the user can maintain his present course and/or speed if desired. To further simplify matters, a null parameter has been defined to be equivalent to SAME. A null parameter is either two consecutive slashes or if it is the last parameter, a blank after the previous parameter. This makes the following instruction pairs equivalent:

```
SETSP/RED//310/200     SETSP/RED/SAME/310/200
SETSP/RED/10/310        SETSP/RED/10/310/SAME
SETSP/RED/10            SETSP/RED/10/SAME/SAME
SETSP/RED//310          SETSP/RED/SAME/310/SAME
```
If two specifications are to be made then the duration must be in its normal position. Thus the following pair is equivalent:

\[ \text{SETSP/RED//310//2//260} \quad \text{SETSP/RED/SAME/310/SAME/2/SAME/260/SAME} \]

The end of a statement is more difficult to determine now. Previously, a missing duration signified the end; now a specification with all speed, course and depth parameters null is also an end condition. If the first specification of a request is null then no action is taken except the printing of an error message.

The addition of \text{SAME} prompted other vocabulary additions. Speeds of \text{SLOW}, \text{STANDARD}, \text{FULL} and \text{FLANK} are defined for each ship; the submarine has different values for surfaced and submerged conditions. The word \text{STOP} does more. It is a speed specification of 0 and course and depth specifications of \text{SAME}. All the speeds but \text{STOP} are followed by a course; \text{STOP} is followed by a duration since it is in itself a full specification.

3.3 Computational Methodology

In the above ways, we have considerably simplified the format of the command language. Various changes have been made to improve our model once we gained experience with the original model and discovered its deficiencies. The \text{SETSP} command was found to be lacking in several ways. A logic change was introduced when the program was given the ability to interrupt a previously scheduled turn to begin a new maneuver. This change can be better understood after an examination of the structure of the \text{OWN} record, which stores the sailing plan.

The \text{OWN} record is formed by a succession of blocks which are of two types. The first is used by the program whenever the ship is to be on a constant course and consists of the following information:
1. latitude at start of straight section (degrees)
2. longitude at start of this section (degrees)
3. time of start of this section (minutes)
4. speed throughout this section (knots)
5. course throughout this section (degrees)
6. depth of ship for this section (feet) (for submarines only)

The other format is used for all turns and consists of the following:
1. latitude at start of turn
2. longitude at start of turn
3. time of start of turn
4. the negative of the speed throughout the turn
5. the turning radius for this turn (degrees)
6. depth of ship for this section (for submarines only)

The type of block can always be determined by the program by examining the sign of the speed word in the block. If it is negative, the ship is turning; otherwise the ship is maintaining a straight course. If the vessel is not a submarine then the length of both blocks is reduced to five as depth is not applicable.

The only restriction in the positioning of the individual blocks in the record is that every turn block must be followed by a straight block. This is necessary because the direction of the turn is determined by the program from the courses before and after the turn.

When a player initiates a sailing plan modification at a time "t" then a search is made by the program for the last block beginning before "t". If it is a straight block then either a turn or a straight block can be added after it. If it is a turn then only a straight block can follow.
Most maneuvers begin with a turn; the only maneuvers which don't are those where the new course coincides with the present course. Because of this, the original SETSP allowed the initiation of a new maneuver only during intervals covered by straight blocks. If a turn block was encountered, then the message

WAIT A MINUTE, YOU ARE IN A TURN.

was printed. Modern versions of SETSP function as the original unless a turn is encountered. When this occurs a short straight section is inserted before the new maneuver is even examined. The course used is the course calculated at "t" considering the portion of the turn completed; the speed and depth are set to the values in the turn. The length of the inserted section was arbitrarily set to .1 minutes. This simulates a time lag of 6 seconds when a turn is already scheduled.

A turn is terminated in the above manner only when a new request is made during a time when a previous request specified a turn. All turns in a request are scheduled to be completed when the processing of the request is finished. This sometimes requires adjustments in the timing of the maneuver as indicated below.

If more than two course specifications are to be included in a single request then they are separated by the duration of the first leg in minutes. For example, the request

SETSP/RED/5/30/100/1/5/150/100

has two specifications included. The first is to turn from the present course to 30° at 5 knots and 100 feet depth. One minute after the start of that turn, a second turn is to begin from 30° to 150° at 5 knots and 100 feet. It is possible that the first turn may require more than one minute to complete at this speed for this ship. The time necessary for a turn is a function of the degree of the turn, the turning radius and the speed of the vessel.
The formula can easily be derived. The result is:

\[ T = \frac{20 \pi \varphi R}{S} \]

where \( T \) is the time for the turn in minutes
\( \varphi \) is the number of degrees turned
\( R \) is the radius of the turn in degrees
and \( S \) is the speed in knots.

If more time is required to complete a turn than is available from a specified duration then several possible steps are possible. One is to consider the entire specification invalid. This was done by the original SETSP; it printed the message:

SAILING PLAN IGNORED, INCREASE DURATION WHEN CHANGING COURSE.

This seemed unsuitable so it was decided to seek a remedy. There are two other fairly obvious ways of proceeding. The first is to complete as much of the turn as possible in the allotted time; the second is to complete the specified turn by increasing the duration. Both involve making a change in the request. It was felt that the best choice was to complete the turn and set the duration to the time required for the turn plus an arbitrary \( .1 \) minutes. The extra time was used for a short straight section to avoid having two consecutive turn blocks.

Course specifications have been changed even more. Normally the course is a number representing the course in degrees. If the course parameter is either RIGHT or LEFT then a relative course change is indicated. The next parameter is the number of degrees to be turned. Thus, a relative course change requires two words; the first gives direction and the second magnitude. This flexibility is particularly useful when an evasion maneuver is needed. No longer is it necessary to calculate the new course from the old one in the player's head; the program does it faster and more accurately.
Depth changes can also be specified as relative. UP and DOWN are the direction words; they are followed by the increment in feet. If a depth specification would cause the ship to have a negative depth (flying) then the depth is set to 0. Similarly if the depth would exceed the maximum allowable for this vessel then the maximum is automatically substituted. The word SURFACE has also been defined.

The final major change incorporated allows a user to start a sailing plan in the future, if desired. The original version of SETSP always started the new sailing plan at the present time. It was decided that it would be useful to allow the change to start at some time in the future. This was accomplished by allowing the insertion of a time specification before any other parameters except the optional color identification. The player has two alternative ways to set the time. If the first parameter after the identification is the word TIME then the player is using one of these options. If the next parameter is the word PLUS then the third parameter is the number of minutes added to the present time to get the starting time of the new section of the sailing plan. If the second parameter was not PLUS then it is the time when the change is to be effective in four digit format (hrhrminmin). If the specified time is in the past then the present time is automatically substituted and processing continues as if TIME has not been specified. Sample SETSP requests using the time options are:

\[
\text{SETSP/RED/TIME/1345/... the specified change is to start at 13:45} \\
\text{SETSP/RED/TIME/PLUS/4/... the specified change is to start in 4 minutes}
\]
3.4 Strategy Development

The addition of the vocabulary and computational changes described above has had a remarkable effect considering how restricted it appears at first. The ease of specifying zig-zag patterns has been found to be particularly effective; the evasion patterns now show signs of ingenuity whereas before they tended to be uninspired and mechanical.

The use of the procedure definition abilities of the problem solving facility has greatly increased since these changes were made. It is relatively easy to define and save (for a user's exclusive use) a named procedure with parameters. This definition can later be used as often as desired by specifying the name and parameters in a manner which is identical to that used with the original basic commands. One type of procedure which has frequently been used is a sailing plan with four turns, the first two to the left and the last two to the right. The first and third turns are the same amount as are the second and fourth. Two speeds can also be specified. The first sets the speed in the first two legs while the other sets the last two. A sample definition of such a procedure is as follows:

XTRANS PROC EVADE

begin the definition of the
procedure named EVADE

FPARAM A,B,SPD1,SPD2
define parameter list

definition of procedure including
substitutional arguments

END

indicate end of procedure

XENTER/RED

save this in file RED

In this definition A and B are relative course parameters while SPD1 and SPD2 specify speeds. If RED should want to execute this procedure at some
later time then he could do so by typing EVADE followed by the parameter values to be substituted. An example is:

\texttt{EVADE/30/20/8/FULL}

The sailing plan changes would be made as indicated in the original definition with 30 substituted for A, 20 for B, 8 for SPD1 and FULL for SPD2. If desired, any of the parameters could have been null; this has the same effect as a null in the original definition. For example, \texttt{EVADE/30/30} has the effect of two 30° left turns followed by two equal right turns; the speed is unset so it is equivalent to \texttt{SAME}. If a single left turn followed by a single right turn is desired, then the second parameter is deleted and both speeds should be supplied. This makes the statement equivalent to:

\texttt{SETSP/RED/SPD1/LEFT/A///3/SPD1///4/SPD2/RIGHT/A///3/SPD}

If only the first parameter had been supplied then the request degenerates to a single left turn. This is because the second leg is completely null. (the sequence /LEFT// is equivalent to /SAME/)

The example given above for an evasion procedure is relatively simple in form, but it is quite versatile. More complex definitions can easily be made as desired and old ones deleted too. By storing the procedure in RED, full protection is gained since WHITE has no access to that file. WHITE of course has his own private WHITE file for his use.

It is hoped that the discussion of the evolution of this major function has illustrated the value of a flexible problem solving facility. It is further hoped that the reader has gained some insight into the types of evolution that can occur on a problem solving system. The function discussed has undergone many changes. The two main classifications of these changes are:
1) the correction of deficiencies discovered through use, and
2) the addition of new powers.

The first stage of SETSP evolution was principally to reduce the number of circumstances under which the program was unable to function. The second phase added some completely new powers to make the program more flexible.
4.0 SONAR ANALYSIS AND PROCESSING

4.1 General

The experience in development of the techniques of sonar analysis, on-line, at the display console give a concrete example of the power of an on-line Problem Solving Facility. In this section is described the evaluation of two key portions of the overall system, as they were developed by an individual researcher aided by on-line facilities. At all times the system was in operation; its performance level (and capability) changed markedly over the course of this interactive experiment.

In any real naval encounter sensor data plays a prime role. To simulate such an environment, sources of such data must be supplied. Two types of sensor data are presently provided:

1) Active sonar gives data on the range and bearing to the other ship at specified times.

2) Passive sonar provides only bearing information.

All sonar data is obtained from system programs which calculate the true range and bearing, insert noise, and then pass the readings to other programs. The amount of noise and some characteristics such as layer depth are controllable by a user in the non-competitive referee state.

The analysis of sonar data is a major task of the user. To facilitate solutions of sonar data, two basic sonar analysis programs were prepared. The first is used for active sonar. It takes a set of time, range, bearing triads representing active sonar points and generates an estimate having constant course and speed. The method of fitting the points to the line is discussed in detail in Section 4.2. The basic analyzer is imbedded in several functions. The function ESTAC (ESTimate using ACTIVE sonar) uses a group of n readings ending at the present time or a selected time in the past. ELISA (Estimate
with LIST Active) is used in conjunction with the display to allow the user to select a group of readings with the light pen and then make an estimate using just the selected readings. Both of these functions are also available in versions which allow higher level active sonar analyzers to call them and pass parameters internally. The higher level analyzers are discussed in detail in Section 4.3.

The second basic analyzer is used with passive sonar and is called ESTPS (ESTimate using Passive Sonar). It requires a fixed number of points (4) to create a straight line fit. It does however select points to a limited degree to make the solution possible. This process is discussed in detail in Section 4.4.

Both estimating programs print an estimate of the relative position, course and speed of the opponent. In addition an ESTO (ESTimate of Opponent) record is created which saves the information in the data management system for use by other programs if desired by the user.

The area of active sonar analysis is being further developed so that the user will have more flexibility in selecting how a solution is determined. He will be able to choose different levels of computational complexity as desired depending on the time which is available and the confidence he has in the computer generated solutions. He will still be able to select points using the interactive display, and the facility of having the computer computationally select points according to criteria controlled by the user is being added. Two new programs are under development which add much to this problem area.

The program ASSEL (Active Sonar SELECT) will provide the user with the option of having the computer computationally select the best points to be used by an internal version of the active estimating program ELISA. The
program ASCOM (Active Sonar COMPare) can be used to determine when an existing estimate should be replaced. ASCOM includes the facility that the user can have either function ESTAC or ASSEL executed automatically when a new estimate is needed; this is controlled by ASCOM parameters.

The four basic programs to be available to the user are ESTPS for passive sonar and ESTAC-ELISA, ASCOM and ASSEL for active sonar. These will provide an extremely flexible and powerful computational group.

4.2 Active Sonar Estimates

The design of the sonar analysis programs has been influenced greatly by experience gained on-line using the original active sonar processing program ESTOP (ESTimate OPPonent). ESTOP and ESTAC are identical in function. There are several differences in implementation which make the newer ESTAC much more effective.

Computationally both programs fit a line to a group of active sonar points. ESTOP used a technique which emphasized the first and last points in the interval. Speed was determined from the distance between those two extreme points and their times. Position at the time of the final reading and course over the interval were obtained by applying a Least-Square-Fit technique to minimize the sum of all the squares of the perpendicular distance from each point to the line. This technique was sufficiently valid when ESTOP was in use because no noise was present in the readings. ESTAC uses a line fitting technique which stresses no single point. This is achieved by a least-square technique which considers the time of all of the points rather than just the two extreme times. The line resulting from ESTAC minimizes the sum of the squared distance from the point to the position on the line corresponding to the time of that sonar reading. Comparison testing has shown the newer method to be far superior in the presence of noise.
Another significant difference between the two programs is the manner in which the sonar readings are obtained. The older ESTOP used sonar records directly; each record was a sequential set of sonar readings which had been requested at a time prior to the earliest reading in the record. ESTOP used all of the readings in a specified record which were available at the time of the ESTOP request.

The procedure followed when an estimate of the opponent was desired was first to request that \( n \) sonar readings be taken by executing USESN (USE SteNsur) at a time before the sonar readings were needed. USESN created a sonar record with the name specified by the user, which could then be used by ESTOP, with the actual reading computation done by a system program. A sample USESN call was as follows:

\[
\text{USESN/RED&SON^4&STEP/=\Delta t/=n}
\]

where RED was the user's identifying color, SON^4 was the name of the sonar record to be created, STEP specified that \( n \) readings were to be taken at interval \( \Delta t \) with the first reading immediately. Since \( \Delta t \) was commonly between .25 and 1 minute, at least .25 minute had to elapse before the minimum number of readings (2) required to make an estimate would be available in the new record. The user therefore had to execute ESTOP on an older sonar record and request a new record each time he executed any functions, to insure that an up-to-date sonar packet would always be available.

After a sonar record was created then ESTOP could be executed. ESTOP performance required that the user supply the sonar record name and the name of the ESTO record to be created. A sample ESTOP call was

\[
\text{ESTOP/SON^4/DD^4}
\]
where SON was again the sonar record name and DD was the ESTO name to be used. ESTOP provided a printout which included the estimated course and speed of the opponent, and the time that the estimate was made. ESTOP stored internally an ESTO record which could be used by other programs. It did not print out the range and bearing to the opponent; it was necessary to execute RANGE to get this information. A sample range call is:

```
RANGE/RED/OWN/DD
```

which means: calculate the present range and bearing from RED's true position to estimate DD.

These three functions had to be executed every time a new estimate was desired. It was felt that this essential task was needlessly complex for the user, so several improvements were installed in a new active sonar analyzer and the name was changed from ESTOP to ESTAC to emphasize that it is an active sonar analyzer only.

The first change was to add the ability to turn the sonar on and leave it on until it is turned off. ESTAC gets its sonar readings indirectly from another program (for internal use only) which has the ability of searching for a group of sonar readings ending at the time specified by ESTAC. ESTAC supplies the time of the last reading desired and the number of readings, and the system provides a list of the data specified. This change eliminated the necessity of frequent USESN calls; now the sonar is usually just turned on and left on.

A minor change eliminated the necessity of executing RANGE each time; the ESTAC printing was simply expanded to give the range and bearing too.

These two changes reduced the number of programs specified by the user to the bare minimum - one. Further modifications reduced the data requirements.
One advance was achieved by giving the computer the ability to automatically give a name to the ESTO record if desired. With ESTOP the user had to supply a unique name to each ESTO record since only the last record of a particular name is available. The user now has the option of supplying the name to ESTAC or having the computer assign it for him. The ESTO name whether supplied or assigned is printed out for use with other worker programs. This change transferred the last bookkeeping task from the user to the computer.

The computer also has assumed the task of identification; the user no longer has to specify identification on each call to ESTAC.

The format for an ESTAC call is as follows:

```
ESTAC/(number of readings)/(time of last reading)/(ESTO name)
```

If the ESTO name is not supplied then the computer generates and prints its own. If the time of last reading is not specified or if the time is in the future, then the computer uses the present time. If the number of readings is not specified then the computer assumes 4. Note that all parameters are optional! This is a far cry from the requirements of ESTOP. To get the same information as obtained by executing

```
ESTAC
```

the player previously would have had to execute the following functions:

```
USESIN/RED&SON4&STEP/=25/=4
ESTOP/SON4/DD4
RANGE/RED&OWN/DD4
```

The sonar record in the above example is SON4; DD4 is the name assigned to the ESTO record. A count of the characters shows that what used to require 58 characters for a typical request now needs only 5; also only one program is specifically called instead of three thus reducing the time needed for execution.
These changes have considerably improved the effectiveness of the game. Normally only 5 characters are typed for each active estimate; more time is available for tactical decisions. Since no parameters are necessary there is little chance for error; as yet no one has misspelled ESTAC, and that is the only possible error. The active sonar problem has been reduced to a minimum for this level of processing.

The active sonar analyzer ELISA (Estimate LIST Active) is available for use with the display. It is identical to ESTAC except that it uses a list of the times of sonar readings which has been selected by the player, on the display for analysis, rather than a sequential group ending at a specified time. ELISA requests are normally generated by the display system automatically.

4.3 Preprocessing Active Sonar Readings

Extensive use on-line of the ESTAC-ELISA functions showed that further work in the active sonar analysis process was desirable. It was found that the user was making new estimates continually to insure availability of recent data; frequently no new estimate was necessary. The program ASCOM was implemented to give the user a method of determining when a new estimate is needed, and to provide for automatic execution of ESTAC if desired by the user. This function uses an estimate previously created, and successively compares each sonar point with the estimated position at that time to determine when the error between the readings and the estimate exceeds any of certain criteria.

The criteria used include the distance of individual points from the extrapolated position on the estimate, the average distance, and the RMS distance. Errors in distance components are available such as error parallel to the estimate (speed error) and error normal to the estimate (course error).
A total of seven rejection criteria is presently available; these can be turned on and off by the user and all values are adjustable as often as desired. Of particular value to this function is the fact that all parameters can be set at the beginning of the encounter to the values desired by the user; these values are used until specified otherwise. Any subsequent use of ASCOM can include changes in parameters, but these are for single runs only unless the word SET is included in the parameter list in which case the present parameters become the new standard set for this user. In addition the amount of output is under player control. He can choose any one of four output levels ranging from data on each point which runs at least one line per point to a level which prints just the reason that the estimate is no longer valid. This allows the experienced (and trusting) player to save printing time. Of course the output level is similar to the rejection criteria in that a standard level is set and that it can be adjusted for each run of ASCOM. The current values of all rejection criteria are printed if a player includes the word LIST in the parameter list. A description of all parameters and their current values is printed when the word DESCRIBE is present.

It is felt that the program ASCOM is quite a bit more effective because of the above characteristics, many of which are available only in ASCOM at present. To make ASCOM more useful, we have added the ability to specify the action to be taken when an estimate is rejected. The possibilities include no action, use ESTAC, and execute the program ASSEL.

The other area in which further development was felt to be desirable is a way of having the computer select a list of points to be used for an estimate computationally to augment the power of visual selection available with the display and ELISA. The program ASSEL is under development to...
do this; it will provide the internal version of ELISA with a list of times to be used for an estimate.

ASSEL will use readings over a specified time span. A line will be drawn between the average of a small group of readings at the start of the span and that of a group at the end. Then the program will calculate the position estimated by the line at the time of each reading. Rejection of a single point occurs if any rejection criterion is exceeded. If too many points are deleted, then a new try can be made using later readings. All parameters of rejection are to be handled as those in ASCOM so as to minimize typing by the user and to shorten the computation time by reducing the number of parameters handled.

4.4 Passive Sonar Analysis

Initially, active sonar was the only sensor available to a user and ESTOP was the only function to make an estimate of the opponent's path. It was felt that a passive sonar facility would be a useful addition; the development of ESTPS was therefore initiated.

The analysis of passive sonar information is considerably more difficult than for active sonar data; active sonar yields the range and bearing to the opponent at each point while passive sonar provides only the bearing. The solution is also more sensitive as there are circumstances under which passive sonar alone cannot possibly make an estimate, whereas no such conditions exist with the active sonar estimating procedures used by ESTOP and ESTAC. As an example, consider two ships which are on an exact interception course; there is no change of bearing under this circumstance, but there is a change in range unless the two ships are moving with identical speeds and courses. There is no way of telling how far away the ships are using only bearing information, but ESTAC could easily solve the corresponding active
sonar problem. Under many circumstances a passive sonar analysis can be achieved; the rest of this section is a discussion of such a method of solution and some problems inherent in it.

An initial assumption was necessary before any progress could be made; it was that the opponent does not change either his speed or course during the interval over which the estimate is made. The quantities desired are the opponent's speed, course and range. It was found that it was possible to get three linear equations with just these quantities as unknowns if three sonar readings were given. Unfortunately, these equations were not independent; the determinant was zero. After-thought showed that this should not be surprising as linear equations have the property that there can be either zero, one or an infinite number of solutions to a system of equations. If both ships are moving with constant velocity then there are at least two solutions, hence an infinite number must exist. A first version of ESTPS was implemented which required that the player supply an estimate of one of the three unknowns and then the other two would be calculated. This version was satisfactory as long as the supplied value was accurate, but was of little value when it wasn't.

A major reason that the above approach is unusable is that the paths of both ships were solutions of the equations. The question was raised as to the effect of requiring that the sensor ship be maneuvering during the interval, thus reducing the number of solutions known to exist to one. At about the same time, the maneuvering board was introduced to the project personnel. This tool provided new insight into the problem and pointed the way towards a reasonable method of solution. It was found that a unique solution was indeed possible when the sensor ship was maneuvering. A derivation of the solution is provided in Appendix A.
A second version of ESTPS was prepared which used this method of solution. It was considerably more useful than the first but was still unsatisfactory.

One of the problems was the difficulty in insuring that a maneuver was included. The program used four consecutive readings which were normally about .25 minute apart; the total time span was only .75 minutes. It is not natural that a ship changes course or speed appreciably every 45 seconds or less. When an ESTPS was executed on a set of readings which did not include a maneuver, then the program provided only the present bearing and bearing rate since no solution was possible. This problem was largely overcome by having ESTPS always request the last 12 readings and then checking them until a change in sailing plan was found. The position of the sensor ship is known at each time. Let the most recent time be \( t_0 \) and the oldest \( t_{12} \). (Note: the reverse numbering is for convenience in Appendix A.) The course and speed between \( t_0 \) and \( t_1 \) are calculated using the position at those times. This line is then extrapolated to the earlier times \( t_2 \) to \( t_{11} \). As soon as the distance between the true position and the extrapolated position at the same time is non-zero then a maneuver is included in the time spanned; let the time of that reading be called \( t_1 \). For practical purposes it has been required arbitrarily that this distance must exceed .06 miles to be considered a sufficient maneuver. The most recent sonar reading used by ESTPS is \( t_0 \) and the earliest is \( t_1 \). Any readings earlier in time than \( t_1 \) are ignored; if all 12 readings are examined without a maneuver being encountered then a message is printed which gives the present bearing and bearing rate. The selection technique uses the smallest possible time span including a maneuver. In practice this additional ability has more than doubled the effectiveness of ESTPS.
The other major problem noted while using the second ESTPS was caused by the initial assumption; if the opponent is not moving with constant velocity during the interval, then the answer must be wrong. The technique used to alleviate this difficulty was to make two estimates over the same interval and compare the results. To reduce computation time, only the most critical computations are run twice. This was accomplished by using one extra sonar reading to make a total of 5. The first pass calculates the range, relative speed and relative course using sonar readings at \( t_0, t_1, t_2, \) and \( t_i \); the second is identical except that the values at \( t_0, t_1, t_3 \) and \( t_i \) are used. After both passes are complete then the average value of the two estimates of each of the unknowns is used in the remaining calculations, thus somewhat smoothing out any errors. To provide the player with a measure of the agreement between the two passes, an unreliability factor is computed and printed out. This factor is the percent deviation of each of the unknowns. It is computed from the following formula, where the ' - ' indicates the average of both passes.

\[
\text{unreliability factor} = \frac{100}{3} \left[ \frac{1}{R} \sum_{i=1}^{2} | R_i - \bar{R} | + \frac{1}{\phi_R} \sum_{i=1}^{2} | \phi_{R_i} - \bar{\phi}_R | + \frac{1}{V_R} \sum_{i=1}^{2} | V_{R_i} - \bar{V}_R | \right]
\]

where \( R \) is range
\( \phi_R \) is relative course
\( V_R \) is relative speed.

When the two passes are not in agreement then either there is noise in the data or the opponent is turning. In a noiseless environment a high unreliability factor means the opponent is turning. A low unreliability factor provides only the information that the two passes agree closely; it is possible that
they give a similar wrong estimate. Tests with a noiseless sonar model showed that the unreliability factor was successful; a factor below 1% just about guaranteed a good estimate and above about 2% did guarantee an invalid one. The introduction of noise changed this. A factor above 20% usually meant that the opponent was turning, but little correlation was found in the range 5% to 20%. It is expected that in the future the ability to compare estimates will be added in order to determine which of these are probably valid. It seems that this problem is best approached using the display so that a player could visually select a consistent set of estimates. This ability seems necessary in a noisy environment.

If the noise sensitivity is too high and sufficient computation time is available, then additional solutions using different combinations of readings could be made and combined to give an even better estimate.

Under some circumstances one or both passes through ESTPS will be suppressed because of an obvious inconsistency. For example it is impossible for the relative speed of the two ships to exceed 60 knots; if $V_{R_i}$ exceeds 60 then that pass is suppressed internally by ESTPS. Similarly, the maximum range of passive sonar is about 30 miles; a pass is suppressed if the estimated range exceeds 35 miles. In both cases this was done to avoid a possible division by zero; the relative magnitudes of the numerator and denominator are checked before the division to insure that the result is reasonable. Usually, although not always, only one pass is affected in this way; the other continues normally except that it is used directly for the later calculations and no unreliability factor is available.

The present version of ESTPS includes all of the above features and also the advances inherent in ESTAC which are applicable. In the absence of noise it performs well; its performance degrades when incorrect bearings are introduced by noise simulation.
Some maneuvers yield better results with this passive sonar analyzer; this affects the tactics used. It is desirable to avoid a course which yields a direct or near interception. More generally, estimates are best when the bearing rate of change is high. Thus to intercept, one reasonable tactic is to make occasional durations of course on the order of 60 or more degrees and try to make estimates during the time of this rather violent maneuver thus using periods of high bearing rate of change.
5.0 WEAPON SYSTEMS

5.1 General

A major area that must be considered in the construction of a valid and realistic environment for a Navy war duel simulation is that of weapons systems. Having a system that can fire a weapon and then track it along its path provides the motivation for the development of strategies in the areas of navigation, sonar acquisition and estimating as well as weapons control. In an environment in which the problem is to locate and sink an enemy ship, navigation strategies are concerned with maneuvering to get good sonar data, preventing being sunk by an opponent's weapons and maintaining a good firing position. The validity of strategies developed by users in all of the above areas can be fully evaluated only by the employment of a weapons model.

In the use of on-line problem solving for evolution of a weapons system, actual experimental trial and error aided greatly in the testing of competitive strategies for the individual commander. Both human engineering and performance weaknesses were discovered on-line and changed. Mistakes in analysis were naturally discovered during on-line competitive duels and then corrected. Man machine decision making was continually added to the system. A player oriented "decision tree" was also introduced.

The model that has been implemented in the war duel simulation consists of four user functions. These are:

WPARM (Weapon PARAMETERS) - to preset or change any of various decisions which have to be made in order to fire a weapon.

WEAPO (WEAPON) - to launch the weapon,

CNFRM (CONFIRM) - to trace the path of the weapon and determine if during its allotted running time the opposing player was hit, and
WEPST (WEaPon STatus) - to inform the player what weapons he has remaining in his arsenal.

The weapons model has had several improvements made in its structure as information and experience were gained by continuous use of the problem solving system. The rest of this chapter will explain the original design of the model, point out the weaknesses that were discovered during its evaluation and the modifications that were made to strengthen it.

5.2 Launching a Weapon

5.2.1 General

When firing a weapon, the most difficult problem that must be solved is to accurately determine where the opponent is and where he is heading. This information depends upon the constant acquisition of sonar information and the continuous processing of the sonar points into estimates of the enemy's course, speed, bearing and range. Since the war duel simulation is in a real-time environment, the opponent can change his course often and make an estimate invalid in a very short period of time.

5.2.2 Initial Model

The initial weapons launching function was:

WEAPO/(name of weapon)/(ESTO item description)/(number of weapons).

Using this function, the destroyer could elect to fire asrocs, torpedoes, or hedgehogs while the submarine was limited to torpedoes.

Since the weapons systems simulation was a relatively new functional module to be added into the war duel environment, the original design drew upon experience gained during the evaluation of other programs.

The initial weapons model contained improvements over the older versions of existing programs in several areas.
The first areas of improvement were concerned with human engineering factors such as

1) simplifying the amount of input the user must give the computer,
2) blocking together several input statements into a procedure so that many user functions might be executed after a simple command was inputted.

These improvements tended to decrease the likelihood of the user making a typing error, forgetting a necessary parameter in a program which required numerous parameters, and speeding up the process of inputting information. (This is usually the slowest link in an on-line, interactive problem solving system.) These improvements also freed the user to make strategic decisions in the time that would normally have been spent typing.

In the original versions of most programs, the user's identification (red or white) was a required parameter. The addition of a mechanism which automatically retrieved the identification from an area in storage (which was created when the user signed into the system) made the input of this extra information unnecessary. This mechanism added an additional feature to the model since it provided an internal method of protecting one user's weapons from interference by an opponent.

Another method by which the input necessary to launch a weapon was considerably shortened was through the use of default conditions. Each of the three parameters in the WEAP0 call had a default value which was used when the parameter was not specified explicitly. If the first parameter (name of weapon) was left empty, it was assumed that the destroyer commander wanted to fire an asroc, and the submarine commander, a torpedo. If the ESTO item description was not supplied, then the firing was based on the last active estimate that was made by the user. Lastly, it was assumed
that only one weapon was to be fired if a number of weapons was not specified.

The most important feature that was present in the original model was the use of a procedure to allow maximum input requests with minimal typing. The major problem in firing a weapon is to get the best possible estimate. If we chose to fire several weapons - all based on a single estimate - we might very well find that each weapon fired was further and further away from the actual enemy position. For this reason it was necessary to develop a means by which the user could ask the computer to make new estimates for him just prior to a weapon's firing and to provide a mechanism which could continue this automatically when several weapons were to be fired in succession.

In order to make these estimates one must call on the programs described in Chapter 4. The most apparent way to accomplish this would be to create an entry point to the estimating programs which could handle calls from the weapon firing program and then transfer control back. This method would have required changes to ESTAC and ESTPS; it was considered advisable to avoid this. The result of this effort was the decision to use the Multilang procedure capabilities.

5.2.3 Weapons Firing Using a Procedure

The player function WEAPO is a Multilang procedure. The major component of this procedure is the program WEAP which performs the necessary interception and bookkeeping computations. As an example of the construction of a procedure, consider part of WEAPO:
XTRANS PROC WEAP0
FPARAM COLOR, WPNUM, ESTO, WPNUM
SET/L(50)/WPNUM/PLUS/=1
BRNCH/ESTO
:
30 SET/L(50)/L(50)/MINUS/=I/L(45)
ESTAC/COLOR
WEAP/WPNUM/LASTA/COLOR
GOTO/L(30)
:
45 NULL
ENDSTA
50 =2
END

In the first statement we are telling the Multilang system that we are defining a procedure which is to be called WEAP0. Then in the second statement we declare that there are to be four formal parameters: the user's color (COLOR), the name of the weapon to be fired (WPNUM), which of the several ESTO descriptions (to be discussed later) to be used when firing (ESTO), and the number of weapons to be fired (WPNUM). For each of the ESTO descriptions that can be chosen by the player, there exists a sequence of programs similar to the four beginning with statement 30. Suppose, for example, that the user wants to fire three torpedoes and take a new active estimate before each is fired. To do this he would type

WEAP0/TORPEDO/NARNA/=3.

In this statement the word NARNA is the ESTO item description and stands for "fire the first torpedo on a New Active estimate and fire the Remainder also on New Active estimates".

When the Multilang system sees the procedure call WEAP0, it retrieves a copy of the definition from secondary storage and substitutes all the actual parameters in the call for every occurrence of these parameters within the procedure body. It then transfers control to the first executable
statement (which is a program). In this case the first program that is
executed is SET which initializes the procedure (SETS the contents of label
50 to the number of weapons to be fired plus 1 = 4). Since the system is
designed to be as easy as possible to use, the lack of a WPNUM parameter
results in one weapon being fired and the contents of label 50 containing 2.
The next program to be reached is BRNCH, which branches to label 30 after
reading the second parameter (NARNA). At label 30, we again encounter the
program SET which acts as loop control (subtracts 1 from label 50). We
execute ESTAC which gives us a new estimate and then fire a weapon using WEAP,
where the ESTO parameter of WEAP is LASTA. This means that the ESTO item
just created will be used in the interception calculations. These two programs
are executed sequentially n times, where n is the number of weapons to be
fired - in this case 3. Each time WEAP is executed, control is transferred
back to label 30 by the GOTO program and the counter (program SET) is decre-
mented by one. After 3 repetitions label 50 is zero, and control is trans-
ferred to label 45 which is effectively the exit from the procedure.

By employing the procedure method, the player has the option of
electing one of nine ESTO item descriptions when firing a weapon. These are:

LASTA: fire all n weapons on the last active estimate made
LASTP: fire all n weapons on the last passive estimate made
"a name": fire all n weapons on the named estimate
LARNA: fire the first weapon on the last active estimate and fire each
of the remaining n-1 weapons on new active estimates
LPRNP: same as LARNA but with passive estimates
NARNA: make a new active estimate before each of n firings
NPRNP: same as above with passive estimates
NEWA: fire all n weapons on one new estimate
NEWP: same as above for passive
Since each of the ESTO item descriptions represents a sequence of programs, it is very easy to add new descriptions. All that would be needed is a minor change in BRNCH and a list of the new programs.

The major part of the WEAPO procedure is the program WEAP. In the original implementation, the program was designed to make some decisions on its own to relieve the player from unimportant decisions that he otherwise would have had to handle himself.

When WEAP was called it checked to see if the weapons specified were available, computed an interception course based on the specified estimate of the opponent and fired a weapon. The extensive mathematics necessary to compute the earliest possible interception position is described in detail in Appendix B. In the course of execution, several situations may have arisen which were handled automatically. One assumption that was continually made is: when a player tried to fire a weapon, and one of the parameters specified in the request was incorrect, the computer was to correct it automatically, if possible, and execute the corrected program instead of printing an error message and halting. For instance, if the user asked to have a weapon fired based on the last passive estimate, and there was no last passive estimate, the program would state that no passive estimates had been made and proceed with the last active estimate (and vice versa). A situation that occurred much more frequently was that of selecting the correct weapon to be launched. In many instances, the destroyer was about three miles away from the sub when he tried to fire a torpedo; and the program computed that the distance that the torpedo would have to travel to intercept the submarine was greater than its maximum range. In this case the player was informed and an attempt was made to launch an asroc. For each situation, the program had the capability of choosing the correct weapon to be fired.
5.3 Testing and Evaluating the Original Model

5.3.1 Decision Making

During the period when this model was tested and evaluated, it became apparent that the model could be made into a more adequate problem solving tool if further strengths were included in its design.

The first area in which additions were deemed to be necessary was the area of decision making on the machine level. The original model had assumed that it was to fire a weapon if at all possible, and would choose to use an active estimate if no passive could be found. It would choose the correct weapon to fire if the distance to the enemy ship was incorrectly specified as well as making other necessary decisions. These decisions were done automatically and in some instances the action carried out by the computer was not to the liking of the user. To make the man-machine decision process a more interactive and more powerful one, a method was added to the problem solving system which would allow the user to set the decisions to be made automatically by the computer and to change the "decision tree" on-line at any time and as often as desired.

The implementation of this new area was not restricted to just the simple decisions stated above. By writing a program which could set and reset parameters, a new powerful structure was added to the system which includes:

1) branching to one of several new actions when an impasse has been reached in the program. For instance, if there is no last active estimate one can now instruct the computer to a) use the last passive estimate, b) make a new active estimate, c) stop execution and state the reason why.

2) default conditions for all parameters can be specified and changed on-line. Whereas, previously, all default parameters were preset, they can now be changed as often as desired. This is of particular value in the new
weapons launching model which will be described in the next section.

3) many new complex decisions that are dependent on a user introduced value may be added to the automatic decision process carried out by the computer. These types of decisions include performing a specified action if:

   a) an estimate is too old (user specifies the maximum allowable age), or
   b) an estimate is too unreliable, or
   c) if the weapon will run too long before a scheduled interception, etc.

To aid the user in specifying the parameter values that he wants to have, the program WARM (Weapon Parameters) was devised. WARM is called by:

WARM/(keyword)/(parameter name)/(value)/(parameter name)/(value)...

In this program call, the keyword is optional and may be:

ALL - set all non-numerical decisions "on". Between decisions stemming from the same impasse, an automatic precedence is built in.

NONE - all non-numerical parameters are turned off. Numeric parameters are set to original default values.

EXPLAIN - an explanation is printed of how to specify the values of all the parameters available.

LIST - a list is printed of all parameters and their current values.

The parameters are separated into groups of EST (estimating) parameters and WEP (weapon firing) parameters. To enter a value to be stored the input would be as above with no KEYWORD parameter.
When the program reads these values it stores them in a record in secondary storage which is always retrieved and examined when a weapon is to be launched. In this manner it is possible to change the values of parameters as often as desired. It would be feasible to change them before every weapon launch if so desired.

5.3.2 Weapon Launching

Although WEAP, the weapon launching function, had drawn on the experience gained through the use and testing of all the preceding programs in the naval war duel simulation, it was found that the design of a more complex model would provide a better vehicle for the determination of strategies involved with sinking an opponent.

In the original model, if one desired to fire more than one weapon at an opponent, one had to specify the number of weapons to be fired and the area in the procedure to which to branch. If, for example, one wanted to fire 3 weapons, the program WEAP had to be executed three complete times. Since the program employs involved and complex computational methodologies, this was more time consuming than desired. In the use of the procedure, the user could employ old estimates or make new estimates, and the computer did all the interception calculations and fired the weapons. In the above example, when firing three weapons, the user had no control over the angle between the weapons and the bearing that each weapon was fired on.

For the above reasons and because the user may be able to get a more accurate picture of how to fire a weapon by studying several estimates of the enemy instead of just using one, it was decided to revise the weapons model to include firing spreads of weapons and firing on a bearing without using an estimate. By adding this complexity to the model, a method has been provided by which an additional "human decision" can be inserted, if desired.
by the user, in the string of machine problem solving decisions. Since user decision making is involved in these cases, the complex calculations determining the projected position of the enemy ship and the course of the weapons can all be avoided, as their values are supplied. This cuts down the computation time of the program and allows execution of the desired strategies much more quickly.

The new WEAPO function is called by:

\[
\text{WEAPO/(name of weapon)/ (ESTO item description) or/(number of weapons in spread)/ (bearing to fire on) }
\]
\[
\text{(the number of degrees between weapons)/ (vertical angle of launch for ASROC)/(NOFIRE option)/ (number of spreads)}
\]

WEAPO still has the same procedure structure as was described in Section 5.2. The program WEAP, which does the necessary computations, has been revised. The program now has many new capabilities which include all previous capabilities plus: 1) the ability to fire a spread of 2 or 3 weapons using a specified center course or having the computer find the center course (by using the type of estimate described by an ESTO description); 2) firing more than one spread of weapons with the option of making a new estimate between spreads (in the manner that was mentioned in the procedure discussion); 3) electing to have all the calculations done and then have the information on what would happen if one had fired the weapon without actually firing (NOFIRE option above).

The preceding addition marks a major improvement in the area of man-machine interaction in the solution of a problem. By allowing as much or as little involvement in the step-by-step solution, a maximal system is evolved. It is also noteworthy that additional human interaction is achieved in this
part of the process, since such default conditions as the assumed spread between weapons can be set and reset by use of the parameters program (WPARM).

When a weapon is fired, a record describing the position, time and course of firing is created and stored for future use. If the weapon was fired using an estimate projecting the enemy ship's future position, the calculated position and time of interception are also stored in this record. This record has a similar format to the sailing history record, and therefore, the path of the weapon can be displayed by using the display program. An example of this type of record is shown in Table 5-1.

5.4 Confirmation of Weapon Firing

5.4.1 General

After launching a weapon, the primary interest of the user is to determine the success of that weapon. He therefore needs a method by which he can determine whether the weapon launched hit, missed or damaged the enemy ship. To determine this the program CNFRM (CONFIRM) was written.

5.4.2 Original Model

In the original model CNFRM was called by:

CNFRM/(name of the record created when weapon fired).

The first action taken by CNFRM was to retrieve the record which was set up when the weapon was fired (Table 5-1). If the present time was earlier than the expected interception time, then the player was informed that it was too early to confirm a hit. If the inquiry occurred after the calculated interception time, the computer determined where the enemy ship actually was at the interception time (by looking at the opponent's sailing history) and computed the distance (R) between it and the position at which the weapon was fired (see Fig. 5-1).
Table 5-1  Record Created by WEAP (Example)

<table>
<thead>
<tr>
<th>Key</th>
<th>Element</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>player's identification color</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>this is a WEAPO record</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>weapon name and number</td>
</tr>
<tr>
<td>3</td>
<td>200</td>
<td>your latitude at time of firing</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>your longitude at time of firing</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>time of firing (in minutes)</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>speed of weapon in air</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>course of weapon</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>latitude of weapon hitting water</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>longitude of weapon hitting water</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>time of weapon hitting water</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>speed in water</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>latitude of interception</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>longitude of interception</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>time of interception</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>opponent est. lat. at time of firing</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>opponent est. long. at time of firing</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>opponent est. speed at time of firing</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>opponent est. course at time of firing</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>opponent est. depth at time of firing</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>your course</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>your depth</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>name and number of weapon</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>weapon sunk opponent (0-miss, 1-sunk, 2-damage)</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>record was confirmed (0-unconfirmed, 1-confirmed)</td>
</tr>
</tbody>
</table>
Course of ship

Position of weapon

\( \alpha \): course of ship
\( \beta \): bearing from ship to weapon
\( \theta \): \( \beta - \alpha \)
\( R \): distance from ship to weapon
\( R \sin \theta \): distance along width of rectangle
\( R \cos \theta \): distance along length of rectangle

Figure 5-1
Determining if Weapon Hit Enemy Ship
To evaluate the results of the weapon firing two rectangles were drawn. The inner rectangle simulated the size and shape of the ship, while the outer rectangle had dimensions twice those of the ship. If the position of the weapon was outside of both rectangles then the weapon missed. If the weapon was within the outer rectangle the enemy ship was damaged. If it was also within the inner rectangle the ship was sunk.

Once the confirmation had been completed the WEAPO record was marked as having been confirmed and the results were placed within the record as well as being supplied to the player. In this manner if either the referee or the player checked the results of the weapon firing again, the calculations would not be repeated; the program would see that everything was previously computed and print out the results.

5.4.3 Improvements in Weapon Confirmation

The preceding model was a very simple one. The program knew where the weapon would be at the calculated interception time and also knew where the enemy was at that time and just determined the distance between the two. This model did not take into account any variations in the opponent's actual speed and course which would have resulted in an interception at some time different from the calculated time.

Other than the above mentioned shortcomings in the original model we were now faced with an entirely new situation due to the changes made in the weapons launching simulation. If a weapon was fired without the use of an estimate, the assumed position of interception and time of interception would not be calculated and the original model could not even be applied.

To account for all of the preceding, a new and complex method of confirmation of simulated homing weapons was devised and implemented. The better features from the old model were retained. These included accounting for previously confirmed weapons, telling the user when he was inquiring
too early and using the rectangle method to tell whether the weapon was close enough to hit, miss, or damage the opponent.

In the new model, after the weapon record is retrieved and the preliminaries are carried out, the program begins tracing the weapon from its point of arming. At selected time intervals (which become smaller as the weapon approaches the opposing ship) the position of the weapon and the enemy and the distance and bearing between them are calculated. The weapon is assumed to have its own small active sonar unit. If no sonar contact is made, the time is incremented and the calculations are repeated. If the incremented time is greater than the present time, the user is told that he has not gotten a hit as of the present and he should finish confirming the weapon a little later. If the incremented time is greater than the maximum run of the weapon the user is informed that his weapon has missed.

When the weapon is within sonar range of the ship, the rectangle method is employed to determine if the weapon has hit as yet. If it hasn't, the weapon course may be changed, depending on the sonar information, to bring the weapon in on a closer interception. The time is then incremented by a small amount and the calculations are repeated until a hit, miss, or damage results.

If a weapon hits or damages the enemy ship, the user is told of his success and the opponent is informed that he has been hit. If he was sunk, he becomes immobile in the water. If he was damaged, his maneuverability is reduced by a predetermined factor. In this manner each succeeding damage would result in the opponent having less maneuverability than before.

Each time a series of calculations to determine the relative position of the enemy and the weapon is done, the results are placed in the weapon record. In this manner, the exact course of the weapon can be displayed and analyzed, detailing all the turns that were made by the weapon as it bore
5.5 Weapon Status

At any point during the course of a game, a player may want to know how many weapons, of a particular type, still remain available to be fired. This information is of great importance in the planning of strategy; the player has to know if he has the strength to stay and fight or if he should save his last weapons to protect himself during a retreat.

It is also possible that the referee, monitoring the progress of both players, would like to know the weapon status at a particular time. With this information he can execute CNFRM on any of the fired weapons to see if they were successful. He could also retrieve the weapons records to get an insight into why and where they were fired. This allows the referee great flexibility in keeping an accurate move by move picture of the two players.

The weapon status of each player is kept in a small record along with the weapon parameters. It is automatically updated every time the program WEAP is executed successfully.

The weapon status program is called by

WEPST/(OWN item description)/(list of weapons).

If the "list of weapons" parameter is not specified, the status of all weapons available to the player is outputted.

The destroyer's list of weapons consists of TORPEDO, ASROC, HEDGEHOG, any combinations of the preceding three connected by 'AND' or the word ALL. For example, the call:

WEPST/WHITE/TORPEDO'AND' ASROC

would result in the printing of the number of asrocs and torpedoes presently available to the destroyer. The format of the output in this case would be:
YOU HAVE "number" ASROCS LEFT
YOU HAVE "number" TORPEDOES LEFT

The word "ALL" is equivalent to specifying all three names.

Except for the storage of the weapon parameters in the status record, the program WEPST has remained unchanged since its original implementation.

5.6 Conclusions

From the preceding discussions, it is evident that the weapons model has gone through a feedback cycle which has greatly improved the capabilities of the simulation. In the environment supplied by the use of a computer, strategies were developed, systems were tested, results were evaluated, and the necessary modifications were implemented. This process is much less complex and less costly than would have been involved in real life testing.
6.0 A DOCUMENTED GAME

6.1 Game Initialization

As an illustration of the ideas presented in this report, this chapter details the playing of a particular game. Each player sits at a console and via the computer terminal, enters queries and commands to the central computer. The computer simulates each ship's movements and carries out the instructions of the commanders. A detailed description of the basic functions are given in Appendix C. The two players are RED - a submarine - and WHITE - a destroyer. The goal of both players in this game was to attack any detected enemy ships. The figures in this chapter are actual computer generated displays. The game is started by the referee who, in this case, had previously stored the initialization functions in the procedure GAME1.

\texttt{\textasciitilde TRANS PROC GAME1}
\texttt{\textasciitilde INTCL/=2030}
\texttt{\textasciitilde INTLZ/DD}
\texttt{\textasciitilde START/WHT&OWN/=25.2/=30}
\texttt{\textasciitilde START/RED&OWN/=25.3/=29.8}
\texttt{\textasciitilde ND}
\texttt{\textasciitilde ENTER/GAME}

The result of executing GAME1 was

\texttt{GAME1}
\texttt{203000 IS NOW THE PRESENT TIME}
\texttt{INTLZ WAS ENTERED}
\texttt{INITIAL POSITION HAS BEEN STORED FOR TIME = 203020}
\texttt{INITIAL POSITION HAS BEEN STORED FOR TIME = 203023}

The game is initialized. Initial positions are set: DD: $25.2^\circ N, 30^\circ W$
\texttt{SS: 25.3^\circ N, 39.8^\circ W}$

Thus, the ships are placed within passive sonar range of each other. Each player now sets his first sailing plan and requests sonar readings to be taken.
DD:

XTRANS PROC BEGIN
SETSP/15/1\$
USESN/WHITE&PASSIVE/CONTN/=6.4
USESN/WHITE&ACTIVE/CONTN
END
XENTER/WHITE
BEGIN
SPEED COURSE START TIME
15.6/6 1\$ 203429
THE SAILING PLAN HAS BEEN STARTED.
SONAR READINGS WILL BE TAKEN
SONAR READINGS WILL BE TAKEN

SS:

XTRANS PROC PATROL
SETSP/7/345/100/5/S/LEFT/45/S/5/S/RIGHT/45/S/5/
S/LEFT/45/S
USESN/RED&PASSIVE/CONTN/=6.4
END
XENTER/RED

PATROL
SPEED COURSE DEPTH START TIME
7.6/6 345 100 203436
7.6/6 TURN 100 203536
7.6/6 300 100 203617
7.6/6 TURN 100 204036
7.6/6 345 100 204117
7.6/6 TURN 100 204536
7.6/6 300 100 204617
THE SAILING PLAN HAS BEEN STARTED.
SONAR READINGS WILL BE TAKEN.
Figure 6-1
Initial Sailing Plans
Each side now, upon detecting an enemy ship, maneuvers to attack.

DD:

**ESTPS**  
TOO FEW READINGS FOR AN ESTIMATE  203152  
PRESENT BEARING 62 BEARING RATE 0 DEG/MIN

<table>
<thead>
<tr>
<th>SPEED</th>
<th>COURSE</th>
<th>START TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.5°</td>
<td>TURN</td>
<td>35</td>
</tr>
</tbody>
</table>

SETSP/WHITE&OWN/SLOW/=35  
SPEED  COURSE  START TIME  
12.5°  TURN  203418  
10.5°  35  203599

THE SAILING PLAN HAS BEEN STARTED.

SS:

**ESTPS**  
TOO FEW READINGS FOR AN ESTIMATE  203127  
PRESENT BEARING 241 BEARING RATE 0 DEG/MIN

<table>
<thead>
<tr>
<th>SPEED</th>
<th>COURSE</th>
<th>DEPTH</th>
<th>START TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.5°</td>
<td>TURN</td>
<td>10°</td>
<td>203232</td>
</tr>
<tr>
<td>241</td>
<td>TURN</td>
<td>10°</td>
<td>203497</td>
</tr>
</tbody>
</table>

THE SAILING PLAN HAS BEEN STARTED.
Figure 6-2

The Sailing Plans are Changed
time is automatically substituted and processing continues as if TIME has not been specified. Sample SETSP requests using the time options are:

\[
\text{SETSP/RED/TIME/1345/... the specified change is to start at 1345} \\
\text{SETSP/RED/TIME/PLUS/4/... the specified change is to start in 4 minutes}
\]

SETSP version III incorporates all of the features detailed in this chapter. It is considerably more useful than version I. It is less prone to errors of the types made with I; it did add some new errors but the diagnostic messages are unusually helpful and complete.

It is hoped that the discussion of the evolution of this major function has illustrated the value of a flexible problem solving facility. It is further hoped that the reader has gained some insight into the types of evolution that can occur on a problem solving system. The function discussed has undergone many changes. The two main classifications of these changes are first the correction of deficiencies discovered through use and second the addition of new powers. The first stage of SETSP evolution was principally to reduce the number of circumstances under which the program was unable to function. The second phase added some completely new powers to make the program more flexible.
6.2 The Game as Played By the Submarine Captain

We will now give many of the functions executed by the submarine captain so that his decisions can be followed.

ESTPS
RANGE 13.76 BEARING 242 TIME 203316
COURSE 340 SPEED 29 RELIABILITY +451.5
NO ESTO RECORD CREATED

SS receives an estimate of opponent's position using passive sonar. The estimate is very unreliable - the larger the number, the worse the estimate - and only the bearing information is usable. Therefore no new record is created.

SETSP/STAND/LEFT/50//4/SLOW/RIGHT/70
SPEED COURSE DEPTH START TIME
8.500 TURN 100 203718
10.00 191 100 203759
7.500 TURN 100 204118
5.000 TURN 261 100 204220

THE SAILING PLAN HAS BEEN STARTED.

SS maneuvers in order to continue using passive sonar.
Figure 6-3

Submarine's Sailing Plan for First Ten Minutes of Game
ESTPS/2035
PRESENT BEARING 243 BEARING RATE -7193 DEG/MIN

ESTPS/2036
RANGE 02.74 BEARING 244 TIME 203604
COURSE 094 SPEED 05 RELIABILITY +80.99
NO ESTO RECORD CREATED

ESTPS
RANGE 10.39 BEARING 246 TIME 203852
COURSE 040 SPEED 12 RELIABILITY +21.61
NO ESTO RECORD CREATED

ESTPS/2041
RANGE 06.24 BEARING 249 TIME 204052
COURSE 211 SPEED 28
NO ESTO RECORD CREATED

ESTPS/2043
RANGE 10.90 BEARING 250 TIME 204252
COURSE 284 SPEED 12 RELIABILITY +20.54
NO ESTO RECORD CREATED

SS has difficulty obtaining reliable estimates.

SETSP/TIM'E//2/RIGHT/10
SPEED COURSE DEPTH START TIME
5.000 TURN 100 205140
5.000 271 100 205152
THE SAILING PLAN HAS BEEN STARTED.

SS requests a course change to be started in two minutes.

ESTPS/2045
RANGE 00.43 BEARING 252 TIME 204452
COURSE 255 SPEED 07 RELIABILITY +16.33
NO ESTO RECORD CREATED

ESTPS
INSUFFICIENT MANEUVER FOR ESTIMATE, TIME 205052
PRESENT BEARING 259 BEARING RATE 1.232 DEG/MIN

ESTPS/2052
RANGE 03.99 BEARING 260 TIME 205204
COURSE 271 SPEED 23 RELIABILITY +14.30
NO ESTO RECORD CREATED

ESTPS
RANGE 02.24 BEARING 261 TIME 205316
COURSE 306 SPEED 04 RELIABILITY +18.52
NO ESTO RECORD CREATED

SS continues attempts to locate enemy ship.
USES/N/RED&ACTIVE/CONTIN
SONAR READINGS WILL BE TAKEN

<table>
<thead>
<tr>
<th>SPEED</th>
<th>COURSE</th>
<th>DEPTH</th>
<th>START TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.00</td>
<td>TURN</td>
<td>100</td>
<td>205652</td>
</tr>
<tr>
<td>15.00</td>
<td>341</td>
<td>100</td>
<td>205745</td>
</tr>
<tr>
<td>10.00</td>
<td>TURN</td>
<td>100</td>
<td>210052</td>
</tr>
<tr>
<td>5.000</td>
<td>281</td>
<td>100</td>
<td>210138</td>
</tr>
</tbody>
</table>

THE SAILING PLAN HAS BEEN STARTED.

SS starts using active sonar.
SS starts another maneuver and at the same time continues on a general intercept course.

ESTPS/2057
INSUFFICIENT MANEUVER FOR ESTIMATE, TIME 205652
PRESENT BEARING 265 BEARING RATE 9643 DEG/MIN

ESTPS/2058,
RANGE 07.62 BEARING 265 TIME 205804
COURSE 072 SPEED 45 RELIABILITY 3.917
ESTO RECORD RP001 WAS CREATED

ESTPS
RANGE 09.31 BEARING 263 TIME 210140
COURSE 345 SPEED 46 RELIABILITY 1242
NO ESTO RECORD CREATED

ESTAC/12
ONLY ONE READING AVAILABLE AT TIME 210210
RANGE 07.05 BEARING 263

INTER/RP001/=5
NO POSSIBLE INTERCEPTION AT THIS SPEED.

INTER/RP001/=10
IF YOU TRAVEL AT 0010 KNOTS,
INTERCEPTION IN 0005 MINUTES AT 210728 CRSE=-0031

RANGE/RED&OWN/RP001/=2059/=2/=5
RANGE = +0006.880000 BEARING = -0095 AT TIME = 205900
RANGE = +0005.290000 BEARING = -0097 AT TIME = 210100
RANGE = +0003.620000 BEARING = -0095 AT TIME = 210300
RANGE = +0002.010000 BEARING = -0087 AT TIME = 210500
RANGE = +0000.660000 BEARING = -0037 AT TIME = 0

SS obtains his first reliable estimate using passive sonar and determines an intercept course.
SS obtains his first estimate using active sonar.

<table>
<thead>
<tr>
<th>SETSP/TIMET//4/RIGHT/20</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPEED</td>
</tr>
<tr>
<td>5.000</td>
</tr>
<tr>
<td>5.000</td>
</tr>
</tbody>
</table>

The sailing plan has been started.

SS sets a sailing plan to be started in four minutes.

Procedure NEAR stored previously by SS.

SS changes the interval at which passive readings will be used for making estimates. The time interval can be reduced at small ranges.
SS obtains contradictory estimates from passive and active sonar.

The sailing plan has been started.

SS gets estimates and maneuvers to confuse his opponent and to get into a good position for firing torpedoes.
A previously stored procedure to get a new estimate and to fire a torpedo using this estimate.

FIREA

RANGE 02.71 BEARING 260 TIME 212125
COURSE 142 SPEED 11
ESTO RECORD RA009 WAS CREATED

TORPEDO 1 HAS BEEN FIRED AT TIME= 212153, ON COURSE= 238
CALCULATED TO INTERCEPT OPPONENT AT TIME= 212637
AT LATITUDE= 25.29 DEG. N LONGITUDE= 29.91 DEG. W
WEAP RECORD RED 'AND' TORP1 HAS BEEN GENERATED
FIRING BASED ON ESTO RECORD RA009

FIREA

RANGE 02.20 BEARING 252 TIME 212310
COURSE 099 SPEED 10
ESTO RECORD RA010 WAS CREATED

TORPEDO 2 HAS BEEN FIRED AT TIME= 212333, ON COURSE= 237
CALCULATED TO INTERCEPT OPPONENT AT TIME= 212652
AT LATITUDE= 25.30 DEG. N LONGITUDE= 29.90 DEG. W
WEAP RECORD RED 'AND' TORP2 HAS BEEN GENERATED
FIRING BASED ON ESTO RECORD RA010

FIREA

RANGE 02.06 BEARING 221 TIME 212555
COURSE 114 SPEED 08
ESTO RECORD RA011 WAS CREATED

TORPEDO 3 HAS BEEN FIRED AT TIME= 212623, ON COURSE= 201
CALCULATED TO INTERCEPT OPPONENT AT TIME= 213022
AT LATITUDE= 25.30 DEG. N LONGITUDE= 29.90 DEG. W
WEAP RECORD RED 'AND' TORP3 HAS BEEN GENERATED
FIRING BASED ON ESTO RECORD RA011

FIREA

RANGE 01.94 BEARING 208 TIME 212810
COURSE 157 SPEED 13
ESTO RECORD RA012 WAS CREATED

OPPONENT OUT OF WEAPON RANGE-WEAPON NOT FIRED TO CALCULATED INTERCEPT PO@

SS fires three torpedoes.
The three torpedoes miss as shown in the following figures.

**FIRE A**
RANGE 01.88 BEARING 199 TIME 213025
COURSE 151 SPEED 12
ESTO RECORD RA013 WAS CREATED

TORPEDO 4 HAS BEEN FIRED AT TIME= 213058, ON COURSE= 181
CALCULATED TO INTERCEPT OPPONENT AT TIME= 213629
AT LATITUDE= 25.27 DEG. N LONGITUDE= 29.89 DEG. W
WEAP RECORD RED 'AND' TORP4 HAS BEEN GENERATED
FIRING BASED ON ESTO RECORD RA013

**FIRE A**
RANGE 00.78 BEARING 215 TIME 213425
COURSE 355 SPEED 10
ESTO RECORD RA014 WAS CREATED

TOO CLOSE TO FIRE TORPEDO-WEAPON NOT FIRED
TO CALCULATED INTERCEPT POSITION-
RANGE= 5107, BEARING= 226.7, AT TIME= 213459
TO OPPONENT'S POSITION-
RANGE= 6335, BEARING= 214.2

SS fires again

**CNFRM/RED&TORP4**
CONFIRMATION OF TORP4
WEAPON MISSED OPPONENT

but misses.
Figure 6-4

SS Fires TORP1, Based on the Estimate RA009 and TORP2, Based on RA010
Figure 6-5

The Three Torpedo Firings with the Estimate RA011 Upon Which the Course for TORP3 was Based
SETSP/SLOW
SPEED COURSE DEPTH START TIME
5.000 258 100 214019
THE SAILING PLAN HAS BEEN STARTED.

SETSP/29
SPEED COURSE DEPTH START TIME
5.000 TURN 100 214231
5.000 29 100 214320
THE SAILING PLAN HAS BEEN STARTED.

FIREA
RANGE 00.95 BEARING 026 TIME 214240
COURSE 345 SPEED 09
ESTO RECORD RA017 WAS CREATED

TORPEDO 5 HAS BEEN FIRED AT TIME= 214314, ON COURSE= 12.
CALCULATED TO INTERCEPT OPPONENT AT TIME= 214556
AT LATITUDE= 25.33 DEG. N LONGITUDE= 29.91 DEG. W
WEAP RECORD RED 'AND' TORP5 HAS BEEN GENERATED
FIRING BASED ON ESTO RECORD RA017

SETSP/RIGHT/30
SPEED COURSE DEPTH START TIME
5.000 TURN 100 214645
5.000 59 100 214656
THE SAILING PLAN HAS BEEN STARTED.

FIREA
RANGE 01.28 BEARING 037 TIME 214710
COURSE 046 SPEED 10
ESTO RECORD RA018 WAS CREATED

TORPEDO 6 HAS BEEN FIRED AT TIME= 214738, ON COURSE= 39.
CALCULATED TO INTERCEPT OPPONENT AT TIME= 215129
AT LATITUDE= 25.34 DEG. N LONGITUDE= 29.89 DEG. W
WEAP RECORD RED 'AND' TORP6 HAS BEEN GENERATED
FIRING BASED ON ESTO RECORD RA018

SS maneuvers and fires two more torpedoes.

CNFRM/RED&TORPS
CONFIRMATION OF TORPS
WEAPON MISSED OPPONENT

CNFRM/RED&TORP6
CONFIRMATION OF TORP6
WEAPON MISSED OPPONENT

Both torpedoes miss their target.
FIREA
RANGE 01.43 BEARING 086 TIME 215040
COURSE 138 SPEED 24
ESTO RECORD RA019 WAS CREATED

OPPONENT OUT OF WEAPON RANGE - WEAPON NOT FIRED
TO CALCULATED INTERCEPT POSITION -
RANGE = 6.076, BEARING = 127.7, AT TIME = 215111
TO OPPONENT'S POSITION -
RANGE = 1.528, BEARING = 92.93

FIREA
RANGE 02.09 BEARING 163 TIME 215555
COURSE 231 SPEED 28
ESTO RECORD RA023 WAS CREATED

OPPONENT OUT OF WEAPON RANGE - WEAPON NOT FIRED
TO CALCULATED INTERCEPT POSITION -
RANGE = 20.16, BEARING = 225.7, AT TIME = 215621
TO OPPONENT'S POSITION -
RANGE = 2.186, BEARING = 169.4

FIREA
RANGE 03.32 BEARING 170 TIME 220455
COURSE 229 SPEED 10
ESTO RECORD RA026 WAS CREATED

OPPONENT OUT OF WEAPON RANGE - WEAPON NOT FIRED
TO CALCULATED INTERCEPT POSITION -
RANGE = 4.285, BEARING = 187.33, AT TIME = 220529
TO OPPONENT'S POSITION -
RANGE = 3.3793, BEARING = 171.0

The two ships are diverging and no torpedoes can be fired.

The actual positions of the destroyer are shown below together with six estimates made by the submarine.
Figure 6-6

The Estimates RA018, RA019, RA020
Figure 6-7

The Estimates RA023, RA024, RA025
The game was stopped at 22:10.

Figure 6-8

The Sailing History of the Submarine for the Entire Game
6.3 Game As Played by Captain of Destroyer

The following are some of the commands and queries made by the destroyer captain.

ESTPS/2Φ35
PRESENT BEARING Φ64 BEARING RATE .6934 DEG/MIN

ESTPS/2Φ36
TIME REQUESTED WAS IN THE FUTURE. PRESENT TIME USED
RANGE Φ9.8Φ BEARING Φ64 TIME 2Φ3531
COURSE 244 SPEED 47 RELIABILITY + Φ
ESTO RECORD WΦΦ1 WAS CREATED

ESTPS/2Φ36
RANGE 14.57 BEARING Φ64 TIME 2Φ3555
COURSE Φ67 SPEED 34
NO ESTO RECORD CREATED

ESTPS/2Φ38
RANGE Φ1.39 BEARING Φ65 TIME 2Φ3755
COURSE 25Φ SPEED 42 RELIABILITY + Φ
ESTO RECORD WΦΦ2 WAS CREATED

ESTPS/2Φ39
INSUFFICIENT MANEUVER FOR ESTIMATE, TIME 2Φ39Φ7
PRESENT BEARING Φ66 BEARING RATE 1.212 DEG/MIN

DD makes estimates using passive sonar.
THE SAILING PLAN HAS BEEN STARTED.

ESTPS/2φ46
RANGE φ3.61 BEARING φ73 TIME 2φ4555
COURSE φ49 SPEED 19 RELIABILITY + 15.5φ
NO ESTO RECORD CREATED

ESTPS/2φ47
INSUFFICIENT BEARING CHANGE FOR ESTIMATE. 2φ47φφ
PRESENT BEARING φ74 BEARING RATE φ DEG/MIN

ESTPS/2φ49
TIME REQUESTED WAS IN THE FUTURE. PRESENT TIME USED
RANGE φ4.94 BEARING φ76 TIME 2φ4755
COURSE 317 SPEED φ6 RELIABILITY + φ
ESTO RECORD WPφφφ WAS CREATED

DD maneuvers and continues to use passive sonar.

XTRANS PROC NEAR
USES/WHITE 'AND' PASSIVE/STOP
USES/WHITE 'AND' PASSIVE/CONTIN
END
XENTER/WHITE

NEAR procedure - which was stored before the game began.

NEAR
SONAR STOPPED FINAL READING AT 2φ5154
SONAR READINGS WILL BE TAKEN

ESTPS/2φ5φ
INSUFFICIENT MANEUVER FOR ESTIMATE, TIME 2φ4954
PRESENT BEARING φ78 BEARING RATE 1.2φ5 DEG/MIN

ESTPS/2φ51
RANGE φ2.51 BEARING φ79 TIME 2φ5054
COURSE 328 SPEED φ9 RELIABILITY +4.277
ESTO RECORD WPφφφφ WAS CREATED

DD starts using passive sonar at smaller time intervals and gets another estimate.
This procedure was stored before the game began.

THE SAILING PLAN HAS BEEN STARTED.

DD maneuvers to use passive sonar.

DD fires a weapon based on the last estimate which used passive sonar (LASTP).

Torpedo misses target.
ESTPS/2159
RANGE 8.35 BEARING 34 BEARING 33 BEARING 245
COURSE 284 SPEED RANGE 1.51 BEARING 33 TIME 2452
COURSE 61 SPEED 43 RELIABILITY +27.73
NO ESTO RECORD CREATED

ESTPS/2191
RANGE 2.68 BEARING 33 TIME 2195
COURSE 33 SPEED 37
NO ESTO RECORD CREATED

ESTAC/12
NO READINGS AVAILABLE FOR THIS ESTIMATE.

ESTPS/2153
RANGE 5.99 BEARING 34 TIME 2125
COURSE 319 SPEED 31 RELIABILITY +6.345
ESTO RECORD W55 WAS CREATED

WEAPON/LASTP

ANTISUB ROCKET 1 HAS BEEN FIRED AT TIME= 21557, ON COURSE= 85.
CALCULATED TO INTERCEPT OPPONENT AT TIME= 218829
AT LATITUDE= 25.30 DEG. N LONGITUDE= 29.87 DEG. W
WEAP RECORD WHITE 'AND' ASRCL HAS BEEN GENERATED
FIRING BASED ON ESTO RECORD W55

While no active sonar readings are available, another estimate using passive sonar is obtained and an antisubmarine rocket is fired based on this last estimate.

CNFRM/WHITE&ASRCL
CONFIRMATION OF ASRCL
WEAPON MISSED OPPONENT

The rocket misses the sub.
Figure 6-9

The Firing of ASRC1 and the Estimate WP005 Upon Which Its Trajectory is Based
A previously stored procedure to obtain a display of the last fifteen active sonar readings.

Figure 6-10

DD Obtains a Few Active Sonar Readings but They are Poor
INTER/WP95/ = 19

IF YOU TRAVEL AT +901. KNOTS, INTERCEPTION IN +926. MINUTES AT 21342$ CRSE=+978.

INTER/WP95/ = 29

IF YOU TRAVEL AT +932. KNOTS, INTERCEPTION IN +914. MINUTES AT 21222$ CRSE=+984.

DD obtains intercept courses at two speeds.

<table>
<thead>
<tr>
<th>SLOW/39/29</th>
<th>COURSE</th>
<th>START TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>11.99</td>
<td>63</td>
<td>211143</td>
</tr>
<tr>
<td>10.99</td>
<td>TURN</td>
<td>211149</td>
</tr>
<tr>
<td>10.99</td>
<td>93</td>
<td>21130$</td>
</tr>
<tr>
<td>10.99</td>
<td>TURN</td>
<td>211649</td>
</tr>
<tr>
<td>10.99</td>
<td>73</td>
<td>211738</td>
</tr>
<tr>
<td>10.99</td>
<td>TURN</td>
<td>212149</td>
</tr>
<tr>
<td>10.99</td>
<td>103</td>
<td>21232$</td>
</tr>
<tr>
<td>10.99</td>
<td>TURN</td>
<td>212649</td>
</tr>
<tr>
<td>10.99</td>
<td>83</td>
<td>212738</td>
</tr>
</tbody>
</table>

THE SAILING PLAN HAS BEEN STARTED.

<table>
<thead>
<tr>
<th>SLOW/39/29</th>
<th>COURSE</th>
<th>START TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.99</td>
<td>TURN</td>
<td>211437</td>
</tr>
<tr>
<td>10.99</td>
<td>123</td>
<td>21155$</td>
</tr>
<tr>
<td>10.99</td>
<td>TURN</td>
<td>211937</td>
</tr>
<tr>
<td>10.99</td>
<td>103</td>
<td>212026</td>
</tr>
<tr>
<td>10.99</td>
<td>TURN</td>
<td>212437</td>
</tr>
<tr>
<td>10.99</td>
<td>133</td>
<td>21255$</td>
</tr>
<tr>
<td>10.99</td>
<td>TURN</td>
<td>212937</td>
</tr>
<tr>
<td>10.99</td>
<td>113</td>
<td>213026</td>
</tr>
</tbody>
</table>

THE SAILING PLAN HAS BEEN STARTED.

DD maneuvers.
The Active Sonar Readings are Improving
DD Obtains More Estimates.

Figure 6-12

Active Sonar Readings Look Reliable
Some readings are eliminated using the light pen and by hitting the appropriate pushbutton, the rest are used to make an estimate.

SLOW/5φ/3φ
SPEED COURSE START TIME
10.0φ TURN 213141
10.0φ 2 213200
10.0φ TURN 213641
10.0φ 332 213652
10.0φ TURN 214441
10.0φ 22 214290
10.0φ TURN 214641
10.0φ 352 214652

THE SAILING PLAN HAS BEEN STARTED.

ESTAC/12
RANGE φ1.55 BEARING φ27 TIME 213146
COURSE 191 SPEED φ4
ESTO RECORD WAΦ6 WAS CREATED

ELISA/WHITE/=213216,=213311,=213146,=213131,=213116,=2131
φ1
RANGE φ1.37 BEARING φ27 TIME 213216
COURSE 212 SPEED 11
ESTO RECORD WAΦ8 WAS CREATED

DD continues to maneuver and makes more estimates, the last one using the interactive facility of the display.
TOO CLOSE TO FIRE TORPEDO-ATTEMPTING HEDGEHOG RUN OUT OF RANGE FOR BOTH WEAPONS TO CALCULATED INTERCEPT POSITION-
RANGE = .6981, BEARING = 62.22, AT TIME = 2136°2
TO OPPONENT'S POSITION-
RANGE = .7225, BEARING = 59.87

IF YOU TRAVEL AT +$2$ KNOTS, INTERCEPTION IN +$2$ MINUTES AT 213654 CRSE = +$138$.

TOO CLOSE TO FIRE TORPEDO-ATTEMPTING HEDGEHOG RUN TOO DISTANT FOR HEDGEHOGS, TOO CLOSE FOR TORPEDO TO CALCULATED INTERCEPT POSITION-
RANGE = .2338, BEARING = 134.7, AT TIME = 213636
TO OPPONENT'S POSITION-
RANGE = .2283, BEARING = 127.9

TORPEDO 2 HAS BEEN FIRED AT TIME = 2139°5, ON COURSE = 186
CALCULATED TO INTERCEPT OPPONENT AT TIME = 2144°7
AT LATITUDE = 25.29 DEG. N, LONGITUDE = 29.91 DEG. W
WEAP RECORD WHITE 'AND' TORP2 HAS BEEN GENERATED FIRING BASED ON ESTO RECORD WA$8$

DD can use only one of these estimates to fire a weapon.

CNFRM/WHITE&TORP2
CONFIRMATION OF TORP2 WEAPON MISSED OPPONENT

The torpedo misses the sub.
ELISA/WHITE/ = 213946, = 213931, = 213961, = 213846, = 213831, = 2138
16, = 213861, = 213746
RANGE 043 BEARING 211 TIME 213946
COURSE 255 SPEED 10
ESTIMATED AT THE BEARING WAS CREATED

WEAPo/TORPEDO/LASTA

BEARING OF CALCULATED INTERCEPTION POSITION IS 153.7
TORPEDO CANNOT BE FIRED AT THIS BEARING.
TO CALCULATED INTERCEPT POSITION-
RANGE = 2.221, BEARING = 228.5, AT TIME = 214433
TO OPPONENT'S POSITION-
RANGE = 1.618, BEARING = 217.1

DD obtains another estimate but is not positioned correctly to fire a torpedo.
- 101 -

**SLOW/5/4/5**

<table>
<thead>
<tr>
<th>SPEED</th>
<th>COURSE</th>
<th>START TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>18.00°</td>
<td>TURN</td>
<td>214445</td>
</tr>
<tr>
<td>18.00°</td>
<td>72</td>
<td>214504</td>
</tr>
<tr>
<td>18.00°</td>
<td>TURN</td>
<td>214945</td>
</tr>
<tr>
<td>18.00°</td>
<td>32</td>
<td>215000</td>
</tr>
<tr>
<td>18.00°</td>
<td>TURN</td>
<td>215445</td>
</tr>
<tr>
<td>18.00°</td>
<td>82</td>
<td>215504</td>
</tr>
<tr>
<td>18.00°</td>
<td>TURN</td>
<td>215945</td>
</tr>
<tr>
<td>18.00°</td>
<td>42</td>
<td>220000</td>
</tr>
</tbody>
</table>

**THE SAILING PLAN HAS BEEN STARTED.**

**ESTAC/12**

**RANGE φ1.19 BEARING 207° TIME 214501**

**COURSE φ31° SPEED φ4**

**ESTO RECORD WA011 WAS CREATED**

**FULL/7/5/5**

<table>
<thead>
<tr>
<th>SPEED</th>
<th>COURSE</th>
<th>START TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>20.00°</td>
<td>TURN</td>
<td>214816</td>
</tr>
<tr>
<td>30.00°</td>
<td>142</td>
<td>214843</td>
</tr>
<tr>
<td>30.00°</td>
<td>92</td>
<td>215316</td>
</tr>
<tr>
<td>30.00°</td>
<td>TURN</td>
<td>215335</td>
</tr>
<tr>
<td>30.00°</td>
<td>162</td>
<td>215816</td>
</tr>
<tr>
<td>30.00°</td>
<td>TURN</td>
<td>220316</td>
</tr>
<tr>
<td>30.00°</td>
<td>112</td>
<td>220335</td>
</tr>
</tbody>
</table>

**THE SAILING PLAN HAS BEEN STARTED.**

**ESTAC/12**

**RANGE φ1.27 BEARING 232° TIME 214901**

**COURSE φ51° SPEED φ5**

**ESTO RECORD WA012 WAS CREATED**

**WEAPO/TORPEDO/LASTA**

**BEARING OF CALCULATED INTERCEPTION POSITION IS 239.°**

**TORPEDO CANNOT BE FIRED AT THIS BEARING.**

**TO CALCULATED INTERCEPT POSITION**

**SETP/FULL/23/4/SLOW/S**

<table>
<thead>
<tr>
<th>SPEED</th>
<th>COURSE</th>
<th>START TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>30.00°</td>
<td>TURN</td>
<td>215287</td>
</tr>
<tr>
<td>30.00°</td>
<td>230</td>
<td>215316</td>
</tr>
<tr>
<td>18.00°</td>
<td>230</td>
<td>215637</td>
</tr>
</tbody>
</table>

**THE SAILING PLAN HAS BEEN STARTED.**

**DD maneuvers but still cannot fire a weapon**
ESTAC/12
RANGE 2.02 BEARING 294 TIME 215246
COURSE 75 SPEED 5
ESTO RECORD WA013 WAS CREATED

WEAPON/TORPEDO/LASTA

TORPEDO 3 HAS BEEN FIRED AT TIME= 215341, ON COURSE= 315
CALCULATED TO INTERCEPT OPPONENT AT TIME= 215712
AT LATITUDE= 25.32 DEG. N LONGITUDE= 29.90 DEG. W
WEAP RECORD WHITE 'AND' TORP3 HAS BEEN GENERATED
FIRING BASED ON ESTO RECORD WA013

DD fires another torpedo,

CNFRM/WHITE&TORP3
CONFIRMATION OF TORP3
WEAPON MISSED OPPONENT

but misses again.
Figure 6-13
The Submarine Overlayed With the Estimates WA012, WA013, WA014 Made by DD
ELISA/WHITE=215301,=215246,=215201,=215146,=215116,=2151
φ1,=215031,=215016,=214946
RANGE φ2. φ2 BEARING 297 TIME 215301
COURSE φ64 SPEED φ6
ESTO RECORD WA014 WAS CREATED

WEAPO/TORPEDO/LASTA

BEARING OF CALCULATED INTERCEPT POSITION IS 218.9
TORPEDO CANNOT BE FIRED AT THIS BEARING.
TO CALCULATED INTERCEPT POSITION-
RANGE= 2.649, BEARING= 11.09, AT TIME= 215739
TO OPPONENT'S POSITION-
RANGE= 2.396, BEARING= 1.876

ESTAC/12
RANGE φ2.52 BEARING 355 TIME 215746
COURSE 356 SPEED φ4
ESTO RECORD WA015 WAS CREATED

WEAPO/TORPEDO/LASTA

OUT OF TORPEDO RANGE-ATTEMPTING ASROC FIRING
OUT OF RANGE FOR BOTH WEAPONS
TO CALCULATED INTERCEPT POSITION-
RANGE= 2.737, BEARING= 357.2, AT TIME= 215822
TO OPPONENT'S POSITION-
RANGE= 2.619, BEARING= 357.1

DD attempts more firings but is either at
the wrong bearing or range.
Figure 6-14

The Firing of TORP3 and the Estimate WA013 Upon Which Its Course Was Based
DISPLAY

OPHIS COMPLETED

Figure 6-15

Active Sonar Readings
ESTAC/12
RANGE $3.27$ BEARING $347$ TIME $22\,\text{o}'216$
COURSE $271$ SPEED $12$
ESTO RECORD $\text{WA}\,16$ WAS CREATED

WEAP/LASTA

OUT OF TORPEDO RANGE-ATTEMPTING ASROC FIRING

ANTISUB ROCKET $2$ HAS BEEN FIRED AT TIME $= 22\,\text{o}'33$, ON COURSE $= 339$
CALCULATED TO INTERCEPT OPPONENT AT TIME $= 22\,\text{o}'57$
AT LATITUDE $= 25.32$ DEG. N LONGITUDE $= 29.93$ DEG. W
WEAP RECORD WHITE 'AND' ASRC$2$ HAS BEEN GENERATED
FIRING BASED ON ESTO RECORD $\text{WA}\,16$

INTER/\text{WA}\,16/=29$

IF YOU TRAVEL AT $+\theta\theta\,\text{KNOTS},$
INTERCEPTION IN $+\theta\theta\,\text{MINUTES AT } 22\,\text{h}43$ CRSE $= \,\text{\theta\theta}49.$

CNFRM/WHITE&ASRC$2$
CONFIRMATION OF ASRC$2$
WEAPON MISSED OPPONENT

DD fires another ASROC but misses as the submarine slows down from 15 to 5 knots.
Figure 6-16

Firing of ASRC2 Based on the Estimate WA016
6.4 Results of Game

The game was stopped at 22:10 without either player being successful in attaining his goal of sinking the enemy ship. The sailing histories of both ships for the entire game is shown below:
Both Ships for the Interval 20:30 to 21:35
Both Ships for the Interval 21:15 to 22:10
APPENDIX A

FOUR BEARING SOLUTION OF PASSIVE SONAR

A detailed derivation of the solution used by the passive sonar analyzer ESTPS is presented in the following appendix.

The assumptions used are:

1. The opponent is maintaining constant speed during the time period.
2. The opponent is maintaining a constant course during the interval.
3. The ship making the estimate has made a maneuver during the period.

The problem is to find the range to the opponent and his speed and course given only the position of the sensor ship and bearing to the opponent at four discrete times, which are referred to as $t_0$, $t_1$, $t_2$, $t_3$.

The solution given here is based on maneuvering board techniques; knowledge of the maneuvering board is not needed for understanding the method of solution as all necessary details are supplied. It is useful to know however that all calculations are relative to the position, course and speed of the sensor ship rather than on an absolute basis.

The major steps in the solution are the following:

I. Solution of own ship's course, speed and position on the maneuvering board.

II. Solution of triangles formed by the intersection of extended bearings and own ship's position vectors.

III. Evaluation of opponent's relative course.

IV. Evaluation of the range at $t_3$ knowing his relative course.

V. Evaluation of opponent's relative speed knowing his relative course and range at $t_3$.

VI. Calculation of opponent's absolute course and speed.
Part I: Solution of own ship’s course, speed and position on the maneuvering board. See Figure A1.

a) Calculate own ship’s speed and course using positions at $t_2$ and $t_3$

average (middle) latitude $= \frac{1}{3} \sum_{i=0}^{3} \text{lat}(t_i)$ average latitude of own ship

$\cos\text{lat} = \cos (\text{average latitude})$ cosine middle latitude

dlat = lat$(t_3) - \text{lat}(t_2)$
distance traveled in N-S direction between $t_2$ and $t_3$

dlong = (long$(t_3) - \text{long}(t_2)) \times \cos\text{lat}$
distance traveled in E-W direction between $t_2$ and $t_3$

course = $\arctan (dlong/dlat)$

speedo = $\sqrt{\text{dlat}^2 + \text{dlong}^2}$

own ship’s course
own ship’s speed in degrees per minute

b) projected position at times $t_0$ and $t_1$

$\text{lat}'(t_i) = \text{lat}(t_3) - \text{dlat} \times \frac{t_3 - t_i}{t_3 - t_2}$ $i = 0, 1$

$\text{long}'(t_i) = \text{long}(t_3) - \frac{\text{dlong}}{\cos\text{lat}} \times \frac{t_3 - t_i}{t_3 - t_2}$ $i = 0, 1$

c) difference between projected position and true position at $t_0$ and $t_1$

$\text{elat}(t_i) = \text{lat}(t_i) - \text{lat}'(t_i) = \text{lat}(t_i) - \text{lat}(t_3) + \text{dlat} \times \frac{t_3 - t_i}{t_3 - t_2}$ $i = 0, 1$

$\text{elong}(t_i) = \left[\text{long}(t_i) - \text{long}(t_3) + \frac{\text{dlong}}{\cos\text{lat}} \times \frac{t_3 - t_i}{t_3 - t_2} \right] \times \cos\text{lat}$ $i = 0, 1$

d) convert to radial coordinates

magnitude

$C_i = \sqrt{[\text{elat}(t_i-1)]^2 + [\text{elong}(t_i-1)]^2}$ $i = 1, 2$

angle

$\beta_i = \arctan \left[\frac{\text{elong}(t_i-1)}{\text{elat}(t_i-1)}\right]$ $i = 1, 2$
\[ C_1: \text{Distance between the true position at } t_0 \text{ and } t_1 \text{ and position extrapolated from the positions at } t_2, t_3. \]

\[ \beta_1: \text{Angle from extrapolated to true position.} \]
Part II. Solution of triangles formed by intersecting extended bearings and
own ship's position vector derived in Part I. See Figure A2.

a) evaluation of interior angles

\[ \alpha_i = \theta_i - \theta_{i-1} \]
\[ \sigma_i = \beta_i - \pi - \theta_i \]

since the sum of angles in a triangle is \( \pi \)

\[ \delta_i = \pi - \sigma_i - \alpha_i \]
\[ = \pi - (\beta_i - \pi - \theta_i) - (\theta_i - \theta_{i-1}) \]
\[ = 2\pi - \beta_i + \theta_{i-1} \]
\[ = \theta_{i-1} - \beta_i \]

b) evaluation of \( B_i \) using law of sines

\[ \frac{B_i}{\sin \delta_i} = \frac{C_i}{\sin \alpha_i} \quad \text{or} \quad B_i = \frac{C_i \sin (\theta_{i-1} - \beta_i)}{\sin \alpha_i} \quad i = 1, 2 \]
Figure A-2

Evaluation of $\beta_1$ From $\theta_2$, $\beta_1$, and $c_1$
Part III. Evaluation of $\phi_R$, the opponent's relative course. See Figure A3.

a) solve for interior angles of triangles formed by intersecting extended bearings.

$$\gamma_i = \pi + \theta_3 - \phi_R - \alpha_i \quad i = 1, 3$$

b) $V_R$, the opponent's relative velocity, is constant; we can therefore write

$$A_i = V_R(t_i - t_{i-1}) \quad i = 1, 3$$

c) using the law of sines to solve for $A_i$ in terms of $\alpha_i$ and $\gamma_i$ we get

$$\frac{A_i}{\sin \alpha_i} = \frac{R + B_i}{\sin \gamma_i} \quad i = 1, 3 \text{ and } B_3 = 0$$

making the substitutions indicated in a and b we get

$$\frac{V_R(t_i - t_{i-1})}{\sin \alpha_i} = \frac{R + B_i}{\sin(\pi + \theta_3 - \phi_R - \alpha_i)} = \frac{R + B_i}{\sin(\phi_R + \alpha_i - \theta_3)} \quad i = 1, 3$$

if $\sin \alpha_i \neq 0$ then

$$V_R = \frac{(R + B_i) \sin \alpha_i}{(t_i - t_{i-1}) \sin(\phi_R + \alpha_i - \theta_3)} \quad i = 1, 3$$

d) Setting the equations for $V_R$ for $i = 1, 2$ both equal to $V_R$ for $i = 3$ we get

$$\frac{(R + B_i) \sin \alpha_i}{(t_i - t_{i-1}) \sin(\phi_R + \alpha_i - \theta_3)} = \frac{R \sin \alpha_3}{(t_3 - t_2) \sin(\phi_R + \alpha_3 - \theta_3)}$$

$$i = 1, 2$$

grouping terms containing $R$ and using cross multiplication to clear the denominators we get

$$R[\sin \alpha_i (t_3 - t_2) \sin(\phi_R + \alpha_3 - \theta_3) - \sin \alpha_3 (t_3 - t_{i-1}) \sin(\phi_R + \alpha_i - \theta_3)] = -B_i \sin \alpha_i (t_3 - t_2) \sin(\phi_R + \alpha_3 - \theta_3) \quad i = 1, 2$$
solving both equations for $R$ we get

$$R = \frac{B_1 \sin \alpha_1 (t_3 - t_2) \sin (\phi_R + \alpha_3 - \theta_3)}{\sin \alpha_3 (t_3 - t_{i-1}) \sin (\phi_R + \alpha_1 - \theta_3) - \sin \alpha_1 (t_3 - t_2) \sin (\phi_R + \alpha_3 - \theta_3)}$$

$$i = 1, 2$$

elimination of range yields after cross multiplication

$$B_1 (t_3 - t_2) \sin \alpha_1 \sin (\phi_R + \alpha_3 - \theta_3) [(t_3 - t_1) \sin \alpha_3 \sin (\phi_R + \alpha_2 - \theta_3) - (t_3 - t_2) \sin \alpha_2 \sin (\phi_R + \alpha_3 - \theta_3)]$$

$$= B_2 (t_3 - t_2) \sin \alpha_2 \sin (\phi_R + \alpha_3 - \theta_3) [(t_3 - t_0) \sin \alpha_3 \sin (\phi_R + \alpha_1 - \theta_3) - (t_3 - t_2) \sin \alpha_1 \sin (\phi_R + \alpha_3 - \theta_3)]$$

dividing both sides by $\sin (\phi_R + \alpha_3 - \theta_3) (t_3 - t_2)$ we get

$$B_1 \sin \alpha_1 [(t_3 - t_1) \sin \alpha_3 \sin (\phi_R + \alpha_2 - \theta_3) - (t_3 - t_2) \sin \alpha_2 \sin (\phi_R + \alpha_3 - \theta_3)] =$$

$$B_2 \sin \alpha_2 [(t_3 - t_0) \sin \alpha_3 \sin (\phi_R + \alpha_1 - \theta_3) - (t_3 - t_2) \sin \alpha_1 \sin (\phi_R + \alpha_3 - \theta_3)]$$

there is only one unknown in this equation: $\phi_R$

regrouping similar terms we get

$$(B_1 - B_2) \sin \alpha_1 \sin \alpha_2 (t_3 - t_2) \sin (\phi_R + \alpha_3 - \theta_3) - B_1 \sin \alpha_1 \sin \alpha_3 (t_3 - t_1) \sin (\phi_R - \theta_3 + \alpha_2) + B_2 \sin \alpha_2 \sin \alpha_3 (t_3 - t_0) \sin (\phi_R + \alpha_1 - \theta_3) = 0$$

making the substitution $\xi = \phi_R - \theta_3$ the equation becomes

$$(B_1 - B_2) \sin \alpha_1 \sin \alpha_2 (t_3 - t_2) \sin (\xi + \alpha_3) - B_1 \sin \alpha_1 \sin \alpha_3 (t_3 - t_1) \sin (\xi + \alpha_2) + B_2 \sin \alpha_2 \sin \alpha_3 (t_3 - t_0) \sin (\xi + \alpha_1) = 0$$

using the identity $\sin (A + B) = \sin A \cos B + \cos A \sin B$, we get
\[
\left[ (B_1 - B_2) \sin \alpha_1 \sin \alpha_2 (t_3 - t_2) \cos \alpha_3 + B_2 \sin \alpha_2 \sin \alpha_3 (t_3 - t_2) \cos \alpha_1 - B_1 \sin \alpha_1 \sin \alpha_2 (t_3 - t_2) \cos \alpha_3 + B_2 \sin \alpha_2 \sin \alpha_3 \right] \sin \xi =
- \left[ (B_1 - B_2) \sin \alpha_1 \sin \alpha_2 (t_3 - t_2) \sin \alpha_3 + B_2 \sin \alpha_2 \sin \alpha_3 \sin \alpha_1 - B_1 \sin \alpha_1 \sin \alpha_2 (t_3 - t_2) \sin \alpha_3 + B_2 \sin \alpha_2 \sin \alpha_3 \right] \cos \xi
\]

if \( \sin \alpha_i \) \( i = 1, 3 \) are not zero, then we can divide by \( (\sin \alpha_1 \sin \alpha_2 \sin \alpha_3) \) and get

\[
\left[ (B_1 - B_2) (t_3 - t_2) \cot \alpha_3 + B_2 (t_3 - t_0) \cot \alpha_1 - B_1 (t_3 - t_1) \cot \alpha_2 \right] \sin \xi = - \cos \xi \left[ (B_1 - B_2) (t_3 - t_2) + B_2 (t_3 - t_0) - B_1 (t_3 - t_1) \right]
\]

solving for \( \tan \xi \) we get

\[
\tan \xi = - \frac{\left[ (B_1 - B_2) (t_3 - t_2) + B_2 (t_3 - t_0) - B_1 (t_3 - t_1) \right]}{(B_1 - B_2) (t_3 - t_2) \cot \alpha_3 + B_2 (t_3 - t_0) \cot \alpha_1 - B_1 (t_3 - t_1) \cot \alpha_2}
\]

the solution for \( \phi_R \) is therefore

\[
(A5) \quad \phi_R = \tan \left\{ \frac{(B_2 - B_1) (t_3 - t_2) - B_2 (t_3 - t_0) + B_1 (t_3 - t_1)}{(B_1 - B_2) (t_3 - t_2) \cot \alpha_3 + B_2 (t_3 - t_0) \cot \alpha_1 - B_1 (t_3 - t_1) \cot \alpha_2} \right\} + \theta_3
\]

This solution is valid if certain conditions are met.

1. \( B_1 \) and \( B_2 \) are not both zero.

   This is equivalent to the assumption that the sensor ship must be maneuvering.

2. None of the \( \alpha_i \) can be zero.

   This is equivalent to requiring that the bearings at times \( t_0, t_1 \) and \( t_2 \) not equal the bearing at \( t_3 \). This restriction also eliminates the possibility that \( V_R = 0 \), since \( V_R = 0 \) means the bearing is constant.

For further considerations of these conditions see Part VII of this appendix (page A12).
Figure A-3
Evaluation of Range to Opponent and His Relative Course and Speed Assuming Constant Course and Speed for Opponent
Part IV. Evaluation of Range at $t_3$ knowing $\phi_R$

Now that $\phi_R$ is known, $R$ can be determined directly from equations (A4) on page A7.

\[(A4) \quad R = \frac{B_i \sin \alpha_i (t_3-t_2)}{\sin \alpha_3 (t_3-t_{i-1})} \sin (\phi_R + \alpha_3 - \theta_3) - \sin \alpha_1 (t_3-t_2) \sin \phi_R + \frac{\alpha_3 - \theta_3}{i = 1, 2}\]

To avoid problems with a $B_i$ which is near zero, a combination of these two equations is used.

\[
R = \frac{B_1 \sin \alpha_1 + B_2 \sin \alpha_2}{\sin \alpha_3 (t_3-t_0) \sin (\phi_R + \alpha_3 - \theta_3) + (t_3-t_1) \sin (\phi_R + \alpha_3 - \theta_3)} - \sin \alpha_1 - \sin \alpha_2
\]

or

\[
R = \frac{B_1 \sin \alpha_1 + B_2 \sin \alpha_2}{(t_3-t_0) \sin (\phi_R + \alpha_3 - \theta_3) + (t_3-t_1) \sin (\phi_R + \alpha_3 - \theta_3)} - \sin \alpha_1 - \sin \alpha_2
\]

Since both $B_1$ and $B_2$ cannot be zero, then this equation is always valid even if one $B_i$ is zero.

Part V. Evaluation of Relative Speed, Knowing $\phi_R$ and $R$

Examination of equation (A2) on page A6 shows an easy way to get $V_R$.

\[(A2) \quad V_R = \frac{(R + B_i) \sin \alpha_i}{(t_3-t_{i-1}) \sin (\phi_R + \alpha_i - \theta_3)} \quad i = 1, 3\]

for $i = 3$ we get

\[V_R = \frac{R \sin \alpha_3}{(t_3-t_2) \sin (\phi_R + \alpha_3 - \theta_3)}\]

This is well behaved so it can be used directly as soon as $\phi_R$ and $R$ have been evaluated.
Part VI. Opponent's True Course and Speed

The opponent's estimated true course and speed can now be computed using simple vector addition.

\[
\vec{V}_{\text{TRUE}} = \vec{V}_{\text{OWN}} + \vec{V}_{R}
\]

or

\[
\begin{align*}
\phi_{\text{TRUE}} &= \phi_{\text{OWN}} + \phi_{\text{R}} \\
V_{\text{TRUE}} &= V_{\text{OWN}} + V_{\text{R}}
\end{align*}
\]

Since the own ship's velocity is known (\(V_{\text{OWN}}\) at course \(\phi_{\text{OWN}}\)) and \(V_{\text{R}}\) and \(\phi_{\text{R}}\) have just been computed, then \(V_{\text{TRUE}}\) and \(\phi_{\text{TRUE}}\) are calculated as follows:

\[
\begin{align*}
\phi_{\text{TRUE}} &= \text{atan} \left( \frac{V_{\text{OWN}} \sin \phi_{\text{OWN}} + V_{R} \sin \phi_{\text{R}}}{V_{\text{OWN}} \cos \phi_{\text{OWN}} + V_{R} \cos \phi_{\text{R}}} \right) \\
V_{\text{TRUE}} &= \left[ (V_{\text{OWN}} \sin \phi_{\text{OWN}} + V_{R} \sin \phi_{\text{R}})^2 + (V_{\text{OWN}} \cos \phi_{\text{OWN}} + V_{R} \cos \phi_{\text{R}})^2 \right]^{\frac{1}{2}}
\end{align*}
\]

This completes the analysis of the opponent's estimated course and speed for normal circumstances. Some consideration follows of areas where assumptions were necessary for easy calculation.
Part VII. Special Cases

It was noted that both \( B_i \) cannot be zero. An examination of the results of one \( B_i \) zero and the other non-zero shows that it is not necessary to use this as a special case.

Equation (A3) on page A6 is

\[
(A3) \quad R[\sin \alpha_1(t_3-t_2) \sin (\phi_R + \alpha_3 - \theta_3) - \sin \alpha_3(t_3-t_1) \sin (\phi_R + \alpha_1 - \theta_3)] = -B_i \sin \alpha_1(t_3-t_2) \sin (\phi_R + \alpha_3 - \theta_3) \text{ for } i = 1, 2
\]

If \( B_i = 0 \) then this becomes considerably simplified. If range is assumed non-zero then we get the equation:

\[
(A6) \quad \sin \alpha_1(t_3-t_2) \sin (\phi_R + \alpha_3 - \theta_3) - \sin \alpha_3(t_3-t_1) \sin (\phi_R + \alpha_1 - \theta_3) = 0
\]

The only unknown in equation (A6) is \( \phi_R \). It can be reduced by using the same trig identity as before \( \sin(A + B) = \sin A \cos B + \cos A \sin B \) and by letting \( \xi = \phi_R - \theta_3 \). This yields

\[
[\sin \alpha_3 \cos \alpha_1(t_3-t_1) - \sin \alpha_1 \cos \alpha_3(t_3-t_2)] \sin \xi = -\cos \xi [\sin \alpha_3 \sin \alpha_1(t_3-t_1) - \sin \alpha_1 \sin \alpha_3(t_3-t_2)]
\]

Assuming \( \sin \alpha_3 \sin \alpha_1 \) is non-zero as previously done, and dividing we get

\[
[\cot \alpha_1(t_3-t_1) - \cot \alpha_3(t_3-t_2)] \sin \xi = -\cos \xi [(t_3-t_1) - (t_3-t_2)]
\]

or

\[
\tan \xi = -\frac{(t_3-t_1) - (t_3-t_2)}{\cot \alpha_1(t_3-t_1) - \cot \alpha_3(t_3-t_2)}
\]

or

\[
\phi_R = \tan \left[ \frac{(t_3-t_1) - (t_3-t_2)}{\cot \alpha_1(t_3-t_1) - \cot \alpha_3(t_3-t_2)} \right] + \theta_3
\]

This is exactly the equation that results from setting a \( B_i \) to zero in the full equation (A5) (on page A8) and then cancelling the other \( B_i \). Thus one non-zero \( B_i \) is sufficient to allow solution for \( \phi_R \) using (A5).
APPENDIX B
INTERCEPTION CALCULATIONS

When a player is ready to fire a weapon, he knows his own position, the characteristics of the weapon he is firing, and has an estimate of the present position, course and speed of his opponent. Given this information, he needs to know if an interception is possible and if so, the course the weapon should be fired on for the earliest possible interception. The player would also like to know the position and time at which the weapon will intercept the enemy ship.

There are two basic types of weapons with which we are dealing. The first type (e.g. ASROC) travels through two media at two distinct velocities. They are characterized by the fact that they always travel a fixed distance (e.g. 1500 yards) at the second velocity. For example, if the interception position is calculated to be 5 miles away from you, then the ASROC would be in the air for (5 miles-1500 yards) and would travel the last 1500 yards in the water.

The other type of weapon (e.g. torpedos and hedgehogs) can be considered to be a specific case of the first. This is a weapon which travels only through one medium and at a fixed speed. This assumes that the fixed distance travelled in the second medium is zero.

Assume that your present position is \((x_1, y_1)\), the estimated position of your opponent is \((x_2, y_2)\) and the interception position will be \((x_3, y_3)\).

\[
\begin{align*}
\Delta x & : \text{Change in x} \\
\Delta y & : \text{Change in y} \\
\phi & : \text{Angle} \\
(x_2, y_2) & : \text{Initial position} \\
(x_1, y_1) & : \text{Player's position} \\
(x_3, y_3) & : \text{Interception position}
\end{align*}
\]

Figure B-1
Terminology
The following terminology is introduced here and will be used throughout the following derivations (see Fig. 1).

\[ \Delta \text{longitude} = x_2 - x_1 = \Delta x \]
\[ \Delta \text{latitude} = y_2 - y_1 = \Delta y \]
middle latitude = \( (y_1 + y_2)/2 \)
distanted = \( d =((\text{departure})^2 + (\Delta y)^2)^{\frac{1}{2}} \)
departure = \( \Delta \text{longitude}(\cos(\text{middle latitude})) \)

\[ \phi = \tan^{-1}(\text{departure}/\Delta \text{latitude}) \]

To compute the time and position of interception, the diagram of Fig. 2 is needed. The component sides of the triangle are 1) the distance between the two ships at the time the weapon is fired, 2) the distance travelled by the enemy ship to the interception point and 3) the distance travelled by the enemy ship to the interception point.

Consider the multispeed type of weapon. Define the following terms:

\( v_1 \): first velocity of interceptor (weapon)
\( d_1 \): fixed distance traversed by weapon in second medium
\( t_1 \): time needed for weapon to traverse first medium
\( t_2 \): elapsed time between firing and interception
\( \Delta t \): time taken to traverse second medium \( (t_2 - t_1) \)
\( v_t \): velocity of target ship
\( d \): distance between both ships at time of firing

\( \Delta x, \Delta y, \phi \): as defined in Fig. 1

\( \alpha \): course of target ship
\( \theta \): interior angle of triangle = \( \phi - \alpha \)
\( \beta \): course that weapon is fired on
Figure B-2
Interception with a Multi-Speed Weapon
From these definitions we can represent side 1 of the triangle by "d"; side 2 by "v_t(t_1+\Delta t)" or "v_t(t_1+\Delta t)"; and side 3 by "v_i(t_1+d_1)".

If we examine the triangle we see that every quantity is known except for $t_1$. Therefore, if we solve for $t_1$, this will give the distance travelled by the weapon and the target and will provide a means for computing the interception point and the weapon course.

Applying the law of cosines to the triangle:

$$(v_i(t_1+d_1))^2 = (v_t(t_1+\Delta t))^2 + d^2 - 2v_t(t_1+\Delta t) \cos \theta$$

Expanding:

$$v_t^2t_1^2 + 2v_t t_1 d_1 + d_1^2 = v_t^2(t_1+\Delta t)^2 + 2v_t(t_1+\Delta t)^2 + d^2 - 2v_t t_1 \cos \theta - 2d v_t \Delta t \cos \theta$$

group by powers of $t_1$:

$$t_1^2(v_i^2-v_t^2) + t_1(2v_i d_1-2v_t \Delta t+2v_t \cos \theta) + (d_1^2-v_t^2 \Delta t^2-d^2+2d v_t \Delta t \cos \theta) = 0$$

Solve for $t_1$ using quadratic formula:

$$t_1 = \frac{-2(v_i d_1-v_t^2 \Delta t-d v_t \cos \theta)}{2(v_i^2-v_t^2)} \pm \sqrt{(v_i d_1-v_t^2 \Delta t+d v_t \cos \theta)^2 - 4(v_i^2-v_t^2)(d_1^2-v_t^2 \Delta t^2-d^2+2d v_t \Delta t \cos \theta)}$$

or:

$$t_1 = \frac{(v_i d_1-v_t^2 \Delta t+d v_t \cos \theta)}{(v_i^2-v_t^2)} \pm \sqrt{(v_i d_1-v_t^2 \Delta t+d v_t \cos \theta)^2 + (v_t^2-v_i^2)(d_1^2-v_t^2 \Delta t^2-d^2+2d v_t \Delta t \cos \theta)}$$

$$
(v_i^2-v_t^2)$$
and since \( t_1 + \Delta t = t_2 \) we have computed the total elapsed time between the time of firing and the time of interception.

In the case of the single speed weapon, we have \( \Delta t = d_1 = 0 \) and if we let \( t_1 = t_2 = t \), we can see from Fig. 3 that the law of cosines simplifies to:

\[
(v_i t)^2 = (v_t t)^2 + d^2 - 2v_t t \cos \theta
\]

or:

\[
t^2(v_i^2 - v_t^2) + t(2v_t t \cos \theta) - d^2 = 0
\]

therefore:

\[
t = \frac{-2v_t d \cos \theta \pm \sqrt{4v_t^2 d^2 \cos^2 \theta + 4d^2(v_i^2 - v_t^2)}}{2(v_i^2 - v_t^2)}
\]

If we substitute \((1-\sin^2 \theta)\) for \(\cos^2 \theta\) and divide the \(2d\) out of the radical we get a simplified equation for \(t\):

\[
t = -v_t \cos \theta \pm \sqrt{v_i^2 - v_t^2 \sin^2 \theta} \left( d/(v_i^2 - v_t^2) \right)
\]

Two problems may arise which prevent the solution from being solved for \(t_1\) (or \(t\)) in this form:
1) Discriminant < 0

Conceptually, the only way we will not be able to solve for an interception time is if the target is moving at a speed and an angle such that the interceptor can never catch up.

If the first case, the discriminant less than zero means:

\[(v_i^2 - v_t^2) (d_1^2 - v_t^2 \Delta t^2 - d^2 + 2d v_t \Delta t \cos \theta) ^2 < (v_t^2 - v_i^2)^2 \]

To visualize what this means take the case of the torpedo. We have already solved this equation and know that the discriminant is \((v_i^2 - v_t^2 \sin^2 \theta)\).

Then if the discriminant is less than zero:

\[v_i^2 - v_t^2 \sin^2 \theta < 0\]

\[v_i^2 < v_t^2 \sin^2 \theta\]

\[v_i < v_t \sin \theta\]

Therefore there will be no solution if the target is moving away faster than the weapon can catch up. If this is the case then the player is notified that "No hit is possible".

2) \(v_i = v_t\)

If \(v_i = v_t\) then the denominator, \((v_i^2 - v_t^2)\), is zero and the fraction blows up. As an example, consider the case of the torpedo, when \(v_i = v_t\). We have an isosoles triangle:

\[
\begin{align*}
(x_1, y_1) & \quad (v_i = v_t = v) \\
(x_2, y_2) & \quad vt \\
(x_3, y_3) & \quad vt \\
\end{align*}
\]

Figure B-4 Velocity of Weapon is the Same as that of Target Ship

From Fig. 4 we see that \(\cos \theta = (d/2)/(vt)\). Therefore \(t = d/(2v \cos \theta)\), and we have a simplified solution.
After we have solved for the time that it will take for the weapon to reach the calculated interception point we still need to know the course on which to fire the weapon and we would like to know when the interception will take place. To determine the position consider the following triangle; where $\alpha$ is the course of the target ship.

\[ b = v_t t (=v_t t_2) \]

\[ \cos(\alpha) = \frac{a}{v_t t} \]

\[ v_t t \cos(\alpha) = a = \Delta y = \Delta \text{Latitude} \]

which means that $y_3 = y_2 + \Delta y$. To find the longitude we need the middle latitude where $\text{Middle latitude} = (\text{latitude of target ship} + \Delta \text{latitude})/2$.

Also, from Fig. 5 we have:

\[ \sin(\alpha) = \frac{b}{v_t t} \]

\[ v_t t \sin(\alpha) = b = \text{Departure} \]

and $\Delta \text{longitude} = \text{departure}/\cos(\text{mid. lat.}) = v_t t \sin(\alpha)/(\cos(\text{mid. lat.})) = \Delta x$

which means that $x_3 = x_2 + \Delta x$.

Once we know the latitude and longitude of interception determining the weapon course requires only drawing a straight line between $(x_1, y_1)$ and $(x_3, y_3)$.
From Fig. 6 we can see that the course $\beta$, that the weapon will be fired on is:

$$\beta = \tan^{-1} \frac{\text{DEPARTURE}}{\text{LATITUDE}}$$

where $\Delta$Latitude $= y_3 - y_1$

and the departure is $(x_3 - x_1) \cos \left( \frac{y_3 + y_1}{2} \right)$
APPENDIX C

DESCRIPTION OF GAME FUNCTIONS

This appendix contains information concerning the use of all player or referee functions as they currently exist. Additional examples can be found in Chapter 6 which contains a sample game.

The following table gives general information which applies to the numerical parameters of all programs except as noted for a particular function.

<table>
<thead>
<tr>
<th>Parameter Type</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>time</td>
<td>4 digits: left 2 specify hour (00-23); right 2 digits specify minute (00-59)</td>
</tr>
<tr>
<td>duration</td>
<td>minutes</td>
</tr>
<tr>
<td>speed</td>
<td>knots</td>
</tr>
<tr>
<td>course</td>
<td>degrees (0-359)</td>
</tr>
<tr>
<td>depth</td>
<td>feet</td>
</tr>
</tbody>
</table>

The presence of parenthesis around a parameter indicate that a substitution is to be made. Normally it can be either a number or one of a group of words.
IDENTIFICATION: INTCL: Initialize Clock - A referee function

PURPOSE: Full and exclusive control of the game clock

USAGE & ACTIONS FORMAT

<table>
<thead>
<tr>
<th></th>
<th>No previous INTCL</th>
<th>Was a Previous INTCL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Present time</td>
<td>Scale</td>
</tr>
<tr>
<td>1</td>
<td>INTCL</td>
<td>real time</td>
</tr>
<tr>
<td></td>
<td>INTCL/START</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>INTCL/(time)</td>
<td>specified time</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>INTCL/(scale)</td>
<td>real time</td>
</tr>
<tr>
<td></td>
<td>INTCL/START/(scale)</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>INTCL/(time)/(scale)</td>
<td>specified time</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>INTCL/STOP</td>
<td>---</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

RESTRICTION: Must be executed before any other function at the start of the game. Use is restricted to the referee.
IDENTIFICATION: INTLZ, Initialize the game - A referee function

PURPOSE: To create the player's OWN records

USAGE: INTLZ

RESTRICTION: This must be executed before each game. Execution must precede any use of START.
IDENTIFICATION: START, Set Initial position at START of game - A referee function

PURPOSE: Set initial position of a ship

USAGE: START/(player's identification)/(initial latitude)/(initial longitude)

Both the initial latitude and longitude are specified in degrees.
E latitude and N longitude are positive; W latitude and S longitude are negative.

RESTRICTION: Must be executed for each player at the start of each game.

SAMPLE: START/RED&OWN/= -30.02/=-40

Set the initial position of RED to 30.02°S, 40°W
IDENTIFICATION: SETSP, SET Sailing Plan, A player function

PURPOSE: Control of the maneuvers of a player's ship

USAGE: A. for surface vessels

1. SETSP/(future time specification)/(speed_1)/(course_1)/
   (duration_1)/(speed_2)/(course_2)/(duration_2)/.../
   (speed_n)/(course_n)

2. SETSP/(speed_1)/(course_1)/...

B. for submarines

1. SETSP/(future time specification)/(speed_1)/(course_1)/
   (depth_1)/(duration_1)/.../(speed_n)/(course_n)/(depth_n)

2. SETSP/(speed_1)/(course_1)/(depth_1)/...

There are two basic ways of specifying a future time of execution. In both cases the first word is TIME. If a particular time is chosen then follow TIME with the specified time (e.g. TIME/1215). If the second parameter is PLUS then the time is the present value plus the number of minutes specified in the third parameter (e.g. TIME/PLUS/5). If no time specification is given, the present time is used.

A speed specification can be either a number or any of the following words: SLOW (about 1/3 full), STANDARD (about 2/3 full), FULL, FLANK (about 10% above full) and SAME. If STOP is specified then the speed is set to 0 and the course and depth set to present value; STOP is followed by a duration since it is a full specification in itself.

Courses can be given either as an absolute course (a number between 0 and 360), by the word SAME, or as a relative course. Relative courses are given by a direction word (RIGHT or LEFT) followed by the amount of the turn (between 0 and 180).
SETSP continued:

Depths are given either as a depth in feet, the words SAME or SURFACE or as a relative depth change. Relative depths are given as UP or DOWN followed by the change in feet.

If a speed, course or depth is skipped by typing a '/' instead of one of the above then the word SAME is substituted.

All durations are in minutes.

SAMPLES: For a submarine

SETSP/TIME/1215/SLOW/RIGHT/70/100

at 1215 begin a 70° right turn at slow speed and 100 feet

SETSP/FULL/300/UP/100

begin a turn to 300° at full speed and go up 100 feet

For a surface vessel

SETSP/TIME/PLUS/5/FULL/LEFT/50/4/STANDARD/RIGHT/60

in 5 minutes begin a left turn of 50° at full speed.

Four minutes later make a 60° right turn at standard speed.
IDENTIFICATION: USESN, USE SENSOR - A player function

PURPOSE: Control of the simulated sonar sensors

USAGE: 1) USESN/(Player's color)&{ACTIVE, PASSIVE}/CONTN/(time interval)
   used to start taking continuous sonar readings

2) USESN/(Player's color)&{ACTIVE, PASSIVE}/STOP
   used to stop taking continuous sonar readings

3) USESN/(Player's color)&{ACTIVE, PASSIVE}/(number of readings)/(time interval)
   used to take a specified number of readings

   The time interval in minutes is optional; if unspecified then .25 minutes is assumed.

   The active and passive sonars are completely independent. A single USESN can control either one but not both.

SAMPLE: USESN/RED&PASSIVE/CONTN

   Start making continuous passive sonar reading for player RED
   (assumed separation .25 minutes).
IDENTIFICATION: LAYER, Set LAYER depth - A referee function

PURPOSE: The referee may set a layer depth. If this function is not used, it is assumed that there is no thermal layer.

USAGE:

   LAYER/number of feet

   LAYER/NONE

The parameter NONE eliminates the thermal layer.
IDENTIFICATION: ESTPS, Estimate Passive Sonar

PURPOSE: Estimate position, course and speed of opponent's vessel using available passive sonar readings.

USAGE: ESTPS/(time of last reading)/(ESTO record name)

Both parameters are optional.

<table>
<thead>
<tr>
<th>Parameter Number</th>
<th>Parameter Type</th>
<th>Meaning Assumed if missing</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>time of last reading</td>
<td>present time</td>
<td>format [hr hr min min]</td>
</tr>
<tr>
<td>2</td>
<td>ESTO Record name</td>
<td>generate name internally</td>
<td></td>
</tr>
</tbody>
</table>

An ESTO record is created only when reliability is satisfactory.

RESTRICTIONS: Player should be signed in as either RED or WHITE.
IDENTIFICATION: ESTAC, ESTimate ACTive sonar, A player function

PURPOSE: Estimate position, course and speed of opponent's vessel using available active sonar readings.

USAGE: ESTAC/(number of readings)/(time of last reading)/(ESTO record name)

All parameters are optional.

<table>
<thead>
<tr>
<th>Parameter Number</th>
<th>Parameter Type</th>
<th>Assumed Value</th>
<th>Specification Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Number of readings</td>
<td>4</td>
<td>$1 \leq n \leq 12$</td>
</tr>
<tr>
<td>2</td>
<td>Time of last reading</td>
<td>present time</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>ESTO record name</td>
<td>generate</td>
<td>format (hr hr min min)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>automatically</td>
<td></td>
</tr>
</tbody>
</table>

An ESTO record is created whenever 2 or more readings are available.

RESTRICTION: Player should be signed in as either RED or WHITE.
IDENTIFICATION: ASCOM, Active Sonar Compare - A player function

PURPOSE: Compare active sonar readings with an estimate to determine when a new estimate is needed.

USAGE: ASCOM/(estimate description)/(parameter 1)/.../(parameter n)

The estimate descriptions are one of:

1. Estimate Name

   2a. LASTA  Last Active estimate
   b. LASTP  Last Passive estimate
   c. LAST  Last estimate

   3a. CONTINUE  Continue with the estimate used on the previous call.

If the ESTO described is the same as the name of the last ESTO examined by ASCOM, then CONTINUE is substituted, providing no ESTO description is equivalent to CONTINUE. If a CONTINUE is attempted when the estimate already failed then the description LASTA is tried to see if a newer estimate has been created.

The parameter list consists of two distinct parameter classes:

1. A single word non-positional parameter which can appear anywhere in the parameter list.

2. Parameters which can be set to a value by single words in sequence. If any of the sequence are to be skipped then the description of the next parameter to be set is inserted and the sequence jumps to the specified parameter. The first null parameter stops the processing of the list.

Parameter Description

CLASS I

1. DESCRIBE  Print a description of the parameters and their current values

2. LIST  Print the current values of the parameters
ASCOM continued:

3a. MAX  Maximum output level
b. NORMAL  Normal print output level
c. FULL  Slightly edited output
d. MIN  Minimum output level
e. EDIT  Reduce the printing one level
f. NOEDIT  Increase printing one level
4. SET  Make the parameters for this run the new assumed parameters for this user.

CLASS II

The following parameters can be set by supplying a number at the proper point in the sequence. OFF and ON are also valid and have the obvious effect. All parameters in this list are saved when SET is typed.

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Parameter Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>M  See 3, 4, 5 below</td>
</tr>
<tr>
<td>2</td>
<td>N  &quot; &quot;</td>
</tr>
<tr>
<td>3</td>
<td>D  Reject ESTO when M out of last N points exceed D</td>
</tr>
<tr>
<td>4</td>
<td>D1  Reject when M/N points exceed D1 (component of D parallel to estimate - SPEED sensitive) in the same direction</td>
</tr>
<tr>
<td>5</td>
<td>D2  Reject when M/N points exceed D2 (component of D perpendicular to estimate - COURSE sensitive) in the same direction</td>
</tr>
<tr>
<td>6</td>
<td>AVGD  Reject when the average D exceeds this value</td>
</tr>
<tr>
<td>7</td>
<td>RMSD  Reject when the RMS average D exceeds this value</td>
</tr>
<tr>
<td>8</td>
<td>SUMD1  Reject when the signed sum of D1 exceeds this value</td>
</tr>
<tr>
<td>9</td>
<td>SUMD2  Reject when the signed sum of D2 exceeds this value</td>
</tr>
</tbody>
</table>
ASCOM continued:

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Parameter Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 AGE</td>
<td>Modifier of LASTA, LAST, LASTP: maximum age of estimate</td>
</tr>
<tr>
<td>11 RELIB</td>
<td>Modifier of LASTA, LAST, LASTP: minimum reliability of estimate</td>
</tr>
<tr>
<td>12 ASSEL</td>
<td>Time span to be used for automatic ASSEL execution</td>
</tr>
<tr>
<td>13 ESTAC</td>
<td>Time span of readings used for automatic ESTAC execution</td>
</tr>
</tbody>
</table>

SAMPLES:

This is the statement which is assumed before any parameters are typed.

1. ASCOM/LASTA/3/5/.1/.15/.10/.05/OFF/OFF/OFF/OFF/OFF/OFF/OFF/1.5
   
   This statement sets the values for M, N, D1, D2, AVGD, RMSD, SUMD1, SUMD2, AGE, RELIB, ASSEL and ESTAC respectively.

2. ASCOM/MM/AVGD/.06
   
   Set print level to max.
   
   Change AVGD from .05 to .06.
   
   These changes are effective for one run only.

3. ASCOM/NORMAL/RMSD/.055/SET
   
   Set print level to NORMAL
   
   Set RMSD to .055 and turn it on.
   
   These changes are effective until changed again.

4. ASCOM
   
   Continue with the estimate used in the previous call and with the parameters as they were last set.
IDENTIFICATION: OPHIS, OPPonent HIstory - A player or referee function

PURPOSE: Display sonar readings, estimates of opponent and/or one's own sailing history.

USAGE: 1) OPHIS/(any number of estimates and/or OWN history)/(options to be used)/INCL/first time/final time

2) OPHIS/(sonar record)/(options to be used)/BKSTEP/(time of last readings)/(number of readings to be displayed)

SAMPLES: OPHIS/WHITE&OWN/ALL/INCL/=2050/=2110
display WHITE's true position during the interval 2050 to 2110.

OPHIS/WHITE&OWN/WAOO~/STD/INCL/=2030/=2120
display both WHITE's true position and WHITE's estimate of RED called WAOO for period 2030 to 2120.

OPHIS/WHITE&ACTIVE&SONAR/ALL/BKSTEP/=2130/=20
display 20 active sonar readings ending at 2130 as received by WHITE.
IDENTIFICATION: ELISA, Estimate LIST Active - A player function

PURPOSE: Estimate position, course and speed of the opponent's vessel using specified active sonar readings.

USAGE: ELISA/(t_1),(t_2),(t_3),..., (t_n)/(ESTO name)

The times are specified in descending order - most recent times first. The times are specified in six digit format. The first two are hours, second two for minutes and third pair for seconds.

This function is normally used only with the display, which generates the request automatically.

The ESTO name is optional (see ESTAC).

RESTRICTIONS: The player should be signed in as RED or WHITE.

SAMPLE: ELISA/\text{121531, 121546, 121601}

Make an active estimate using the three specified readings.
IDENTIFICATION: INTER, Find INTERception Course - A player function

PURPOSE: to find an interception course to an estimate of the opponent
at a specified speed.

USAGE: INTER/(ESTO name)/(speed of own vessel desired)

SAMPLE: INTER/RPO04/10

This requests an interception course with RED's estimate of
WHITE called RPO04 at a speed for RED of 10 knots.
IDENTIFICATION: OPINT, Find Opponent's Interception course - A player function

PURPOSE: To enable a player to calculate what course the opponent should use if the opponent is trying to intercept assuming a speed of the opponent.

USAGE: OPINT/(ESTO name)/(speed of opponent's vessel)

SAMPLE: OPINT/RPOO4/=10

This requests that a calculation be made of the course the opponent should follow if its position is as given by RPOO4 and if he travels at 10 knots.
IDENTIFICATION: RANGE, RANGE between positions - A player and referee function.

PURPOSE: To calculate and print the range and true bearing between the positions of two objects at specified times. The objects can be vessels, estimates of vessels or weapons.

USAGE:  
RANGE/(first record)/(second record)/(first time)/(time interval)/  
(number of times)

If the first time is not supplied then the program assumes the present time. If either the time interval or number of times is unspecified then only one time is considered.

SAMPLES:  
1)  RANGE/RED&OWN/RP003/=1215/=1/=10

This supplies the range and bearing from the estimate RP003 to the OWN position of RED for times from 1215 to 1224 separated by 1 minute.

2)  RANGE/RED&OWN/RP003

This provides the present range and bearing.
IDENTIFICATION: POS, POSITION - A player or referee function

PURPOSE: To print the position, speed, course and depth of a vessel at specified times. It can also be used on estimates of a ship or on a weapon's path.

USAGE: POS/(record description)/t_1/t_2/t_3/.../t_n

where t_1 is a four digit number; first two for hour and second two for minute.

SAMPLES: 1) POS/RED&OWN/=1215/=1220/=1225

This supplies the position, speed, course and depth of the true position of RED at 1215, 1220 and 1225.

2) POS/WPO04/=1830

This supplies the information as obtained from the ESTO record WPO04.

3) POS/RED&TORP2/=0900/=0905

This gives the data for RED's torpedo 2.
IDENTIFICATION: WEAP0, fire a WEAPON - A player function

PURPOSE: To fire weapon. The weapons available include ASROC torpedoes and hedgehogs.

USAGE: WEAP0/(type of weapon)/(ESTO item description )/(number of weapons )/(or bearing )/(speed between )/(vertical angle for ASROC expressed as distance )/(from your ship where it will splash down )/(NOFIRE OPTION )/(NUMBER OF SPREADS)

All parameters are optional. The value which is assumed if a parameter is unspecified can be obtained from the following table:

<table>
<thead>
<tr>
<th>Type of Ship</th>
<th>Assumed Weapon Type</th>
<th>ESTO Item Description</th>
<th>Number in Spread</th>
<th>Degrees Between Weapons</th>
<th>Vertical Angle for ASROC</th>
<th>Number of Spreads</th>
</tr>
</thead>
<tbody>
<tr>
<td>Submarine</td>
<td>Torpedo</td>
<td>Last active</td>
<td>1</td>
<td>2</td>
<td>NA</td>
<td>1</td>
</tr>
<tr>
<td>Destroyer</td>
<td>ASROC</td>
<td>Last active</td>
<td>1</td>
<td>2</td>
<td>3 miles</td>
<td>1</td>
</tr>
</tbody>
</table>

SAMPLES: 1) WEAP0/TORPEDO/LASTA

Fire a torpedo using the last active estimate to calculate interception position.

2) WEAP0/ASROC/=84.3/=2/=1.5/=5.0

Fire a spread of two asrocs with 1.5 degrees between them using a center bearing of 84.3 degrees. Compute the vertical angle of launch using assumption that asroc is to hit water 5 miles from point of firing.

3) WEAP0/TORPEDO/NEWA////NOFIRE

Make a new active estimate and give me all the interception information but don't fire the weapon.
IDENTIFICATION: WEPST, Weapon Status - A player or referee function

PURPOSE: To inform the player of the number of weapons remaining

USAGE: WEPST/(list of types of weapons)

If a list of weapon types is not supplied, then all weapons are individually considered.

SAMPLES: 1) WEPST/ALL

This supplies a count of all remaining torpedoes if the player making the request is a sub and supplies a count of asrocs, torpedoes and hedgehogs if the DD makes the request.

2) WEPST/TORPEDO'AND'ASROC

This supplies a count of all remaining asrocs and torpedoes to the DD commander.
IDENTIFICATION: CNFRM, CONFIRM a hit or miss - A player or referee function

PURPOSE: To determine the results of the firing of a weapon.

USAGE: CNFRM/NAME OF WEAPON

or

CNFRM/LAST/(TYPE OF WEAPON)

or

CNFRM/ALL/(TYPE OF WEAPON)

Type of weapon is optional. The weapon names are automatically generated and printed by WEAPO.

RESTRICTIONS: Confirmations are calculated up to present time. If it is too early, a message to that effect is printed.

SAMPLES: 1) CNFRM/TORP4

The results of the firing of torpedo 4 are computed and printed.

2) CNFRM/LAST/ASROC

Confirms last asroc fired.

3) CNFRM/LAST

Confirms last weapon fired.

4) CNFRM/ALL

Confirms all unconfirmed weapons.
IDENTIFICATION: WPARM, set Weapon Parameters - A player function

PURPOSE: To set decision parameters necessary to fire weapons

USAGE: WPARM/sequential list of parameter values separated by slashes or a list of parameter name/value/parameter name/value

SAMPLES: WPARM/WEP8/YES/EST6/=005
          WPARM/YES/NO/=01/YES/=3.0


BIBLIOGRAPHY (continued)


Distribution List for
"Man-Computer Problem Solving in Real-Time Naval Duels (U)"

Director of Defense Research & Engineering
Department of Defense
Washington, D.C. 20301
(Tactical Systems Plans and Analysis) 1

Advanced Research Projects Agency
Department of Defense
Washington, D.C. 20301
(Technical Library) 1

National Military Command System
Support Center
Defense Communications Agency
Washington, D.C. 20301 1

Defense Documentation Center
Cameron Station
Alexandria, Virginia 22314 20

Chief of Naval Operations
Department of the Navy
Washington, D.C. 20350
(Op-31) 1
(Op-06C) 1
(Op-07) 1
(Op-07T) 1
(Op-713) 1
(Op-93) 1
(Op-95) 1
(Op-96) 1

Office of Naval Research
Arlington, Virginia 22217
(Code 437) 1
(Code 460) 1
(Code 462) 2
(Code 455) 1

Naval Material Command
Department of the Navy
Washington, D.C. 20360
(PM-1) 1
(PM-13) 1
(PM-14) 1
(ASW-14) 1
(ASW-23) 1
(ASW-24) 1

Enclosure (2)
Naval Postgraduate School  
Monterey, California 93940  
(Professor J. Borsting)  

Naval War College  
Newport, Rhode Island 02884  

Naval Ordnance Laboratory  
Silver Spring, Maryland 20910  

Naval Research Laboratory  
Washington, D.C. 20390  
(Technical Information Division)  
(Code 2029)  

Naval Weapons Laboratory  
Dahlgren, Virginia 22438  
(Code KCU)  
(Code KP)  
(Code KWP)  

Naval Ship Research and Development Center  
Washington, D.C. 20007  

Naval Training Device Center  
Orlando, Florida 32813  
(Mr. M. Fisher)  

Naval Undersea Warfare Center  
San Diego, California 92152  

Pasadena Laboratory  
Naval Undersea Warfare Center  
3202 East Foothill Boulevard  
Pasadena, California 91107  

Navy Underwater Systems Center  
Newport, Rhode Island 02840  

Fleet Numerical Weather Facility  
Naval Postgraduate School  
Monterey, California 93940  

Chief of Staff, U. S. Air Force (AFGOA)  
Department of the Air Force  
Washington, D.C. 20330  
(Mr. R. S. Thackeray)  
(Mr. C. Thomas)
Computer Applications Group
Assistant Chief of Staff for
Studies and Analysis
Department of Air Force
Washington, D.C. 20331
(Mr. E. W. Lee)

Air Force Cambridge Research
Laboratories (CRRB)
L. G. Hanscom Field
Bedford, Massachusetts 01730

National Aeronautical and
Space Administration
400 Maryland Avenue, S.W.
Washington, D.C. 20546

National Bureau of Standards
Washington, D.C. 20234
(Dr. S. Alexander)

Center for Naval Analyses
1401 Wilson Boulevard
Arlington, Virginia 22209
(Library)

Naval Warfare Research Center
Stanford Research Institute
Menlo Park, California 94025

Program in Logistics
George Washington University
707 22nd Street, N. W.
Washington, D.C. 20037

University of California at Los Angeles
Los Angeles, California 90024
(Professor J. Marschak)

Director, Data Processing and
Computing Center
Columbia University
New York, New York 10027

Massachusetts Institute of Technology
School of Industrial Management
Cambridge, Massachusetts 02139
(Operations Research Center)

Director, Project MAC
Massachusetts Institute of Technology
School of Industrial Management
Cambridge, Massachusetts 02139
Director, Computer Center
Stanford University
Palo Alto, California 94305

Computer Center
University of Wisconsin
Madison, Wisconsin

Computer Command and Control Company
1717 Pennsylvania Avenue, N. W.
Washington, D. C. 20006

Cornell Aeronautical Laboratory, Inc.
P. O. Box 235
Buffalo, New York 14221
(Operations Research Department)

General Dynamics Corporation
Electric Boat Division
Groton, Connecticut 06340
(Mr. T. Downie)

Lincoln Laboratory
Massachusetts Institute of Technology
Lexington, Massachusetts 02173
(Dr. C. R. Wieser, Systems Research)

The MITRE Corporation
Box 208
Bedford, Massachusetts 01730
(Technical Library)

Operations Research Incorporated
1400 Spring Street
Silver Spring, Maryland 20910
(Mr. R. Johnson)

Planning Research Corporation
7600 Old Springhouse Road
Westgate Research Park
McLean, Virginia 22101

Systems Development Corporation
5720 Columbia Pike
Falls Church, Virginia 22041

TRW, Inc.
1875 Connecticut Avenue, N. W.
Washington, D. C. 20009