UNCLASSIFIED

CIVILIAN PERSONAL COMPUTER PROGRAM CASTAG - A COMPUTER ASSISTED INTERACTIVE NAVAL WARGAME (U)

MAR 80  K J KELLEY

NL
CASTAG
A COMPUTER ASSISTED INTERACTIVE NAVAL WARGAME
by
Kevin John Kelley
March 1980
Thesis Advisor: A. Andrus
Approved for public release; distribution unlimited.
CASTAG is an interactive two-sided computer wargame which models the movement and detection functions for the manual naval wargame SEATAG, developed at the Naval War College. CASTAG is...
programmed for the CP/CMS time sharing system on the IBM-360 at the Naval Postgraduate School.
CASTAG
A Computer Assisted Interactive Naval Wargame

by

Kevin John Kelley
Lieutenant, United States Navy
B.A., University of Texas, 1973

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
March 1980

Author

Kevin J. Kelley

Approved by:

Alvin J. Andary
Thesis Advisor

Ellen J. Roland
Second Reader

Michael J. Forester
Chairman, Department of Operations Research

Dean of Information and Policy Sciences
ABSTRACT

CASTAG is an interactive two-sided computer wargame which models the movement and detection functions for the manual naval wargame SEATAG, developed at the Naval War College. CASTAG is programmed for the CP/CMS time sharing system on the IBM-360 at the Naval Postgraduate School.
# TABLE OF CONTENTS

I. INTRODUCTION AND BACKGROUND -------------------------- 9

A. THE NEED FOR AN INTERACTIVE COMPUTER ASSISTED WARGAME AT THE NAVAL POSTGRADUATE SCHOOL -------------------------- 9

B. APPROACHES TO THE PROBLEM AND LIMITS OF THE THESIS -------------------------- 10

C. SELECTION OF SEATAG -------------------------- 10

D. CHAPTER OUTLINE -------------------------- 11

II. SEATAG, THE MANUAL GAME -------------------------- 12

A. HISTORY AND DEVELOPMENT OF SEATAG -------------------------- 12

B. DESCRIPTION OF THE PLAYING AREA, SCALE, ENVIRONMENT AND PLATFORM CHARACTERISTICS OF SEATAG -------------------------- 12

C. GAME FLOW, AIRCRAFT CARRIER AND SUBMARINE OPERATIONS, AND SURVEILLANCE/INTELLIGENCE PLAY -------------------------- 14

D. DETECTIONS -------------------------- 16

E. PLAY OF ENGAGEMENTS AND CALCULATION OF DAMAGE -------------------------- 16

F. TIME UTILIZATION AND SOME OBSERVATIONS ON ACTUAL MANUAL PLAY -------------------------- 17

III. THE CASTAG PROGRAM -------------------------- 18

A. OVERALL PROGRAM OPERATION -------------------------- 18

1. Overview of CASTAG Inputs -------------------------- 18

2. Dual Terminal Operation -------------------------- 20

3. Description of the CASTAG EXEC Routine --- 20

B. SELECTION OF THE PROGRAMMING LANGUAGES -------------------------- 21

C. FILE SPACE AND CPU TIME REQUIREMENTS OF THE CASTAG PROGRAM -------------------------- 21
D. CREATION AND ORGANIZATION OF THE DATA BASE ---- 23
   1. Information Required for the Data Base ---- 23
   2. Organization and Format of the Data Base --- 23
E. PLAYING AREA, TIME AND DISTANCE UNITS --------- 24
F. DESCRIPTION OF FORTRAN PROGRAM SEARCH -------- 27
   1. MAIN Program ----------------------------- 28
   2. Subroutine MOVE -------------------------- 36
   3. Subroutine NUMBER ------------------------ 39
   4. Subroutine DETECT ------------------------ 43
   5. Subroutine SPOOK -------------------------- 52
   6. Subroutine ESM ---------------------------- 57
   7. Subroutine DETRAD ------------------------- 62
   8. Subroutine VISUAL ------------------------- 68
   9. Subroutine ASONAR ------------------------- 73
  10. Subroutine NLPCPA ------------------------- 79
  11. Subroutine PSONAR ------------------------- 92
G. DESCRIPTION OF FORTRAN PROGRAM CHANGE -------105
H. DESCRIPTION OF FORTRAN PROGRAM UPDATE -------109
IV. USER'S GUIDE TO CASTAG ----------------------112
   A. CASTAG TERMINAL INPUTS ---------------------112
   B. DECODING/ENCODING TYPE NUMBERS --------------117
   C. CASTAG TERMINAL OUTPUT ---------------------121
   D. CASTAG OFFLINE PRINTER OUTPUT -------------124
      1. Umpire Summary of Detections -------------125
      2. Red/Blue Cartesian Plot -------------------125
      3. Status of Forces Table -------------------126
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>SEATAG Game Flow</td>
<td>15</td>
</tr>
<tr>
<td>2.</td>
<td>Overall CASTAG Program Flow</td>
<td>19</td>
</tr>
<tr>
<td>3.</td>
<td>Flowchart of SEARCH MAIN Program</td>
<td>35</td>
</tr>
<tr>
<td>4.</td>
<td>Flowchart of Subroutine MOVE</td>
<td>38</td>
</tr>
<tr>
<td>5.</td>
<td>Comparison of Detection Curves</td>
<td>40</td>
</tr>
<tr>
<td>6.</td>
<td>Flowchart of Subroutine NUMBER</td>
<td>42</td>
</tr>
<tr>
<td>7.</td>
<td>Flowchart of Subroutine DETECT</td>
<td>47</td>
</tr>
<tr>
<td>8.</td>
<td>Flowchart of Subroutine SPOOK</td>
<td>54</td>
</tr>
<tr>
<td>9.</td>
<td>Flowchart of Subroutine ESM</td>
<td>59</td>
</tr>
<tr>
<td>10.</td>
<td>Flowchart of Subroutine DETRAD</td>
<td>64</td>
</tr>
<tr>
<td>11.</td>
<td>Flowchart of Subroutine VISUAL</td>
<td>71</td>
</tr>
<tr>
<td>12.</td>
<td>Flowchart of Subroutine ASONAR</td>
<td>76</td>
</tr>
<tr>
<td>13.</td>
<td>Calculation of Point (X1,Y1)</td>
<td>81</td>
</tr>
<tr>
<td>14.</td>
<td>Relative Locations of DRM Endpoints</td>
<td>84</td>
</tr>
<tr>
<td>15.</td>
<td>Four Quadrant TAU Calculation</td>
<td>86</td>
</tr>
<tr>
<td>16.</td>
<td>Flowchart of Subroutine NLPCPA</td>
<td>89</td>
</tr>
<tr>
<td>17.</td>
<td>Comparison of Passive Sonar Detection Curves</td>
<td>94</td>
</tr>
<tr>
<td>18.</td>
<td>Flowchart of Subroutine PSONAR</td>
<td>100</td>
</tr>
<tr>
<td>19.</td>
<td>Flowchart of CHANGE Program</td>
<td>107</td>
</tr>
<tr>
<td>20.</td>
<td>Flowchart of UPDATE Program</td>
<td>111</td>
</tr>
</tbody>
</table>
I. INTRODUCTION AND BACKGROUND

A. THE NEED FOR AN INTERACTIVE COMPUTER ASSISTED WARGAME AT THE NAVAL POSTGRADUATE SCHOOL

Several curricula at NPS study wargaming techniques. The courses include manual wargames and computer simulations as part of the course material. At the inception of this study, none of these courses had access to an interactive computer assisted wargame that could handle the mundane and time consuming plotting and computational chores of wargame bookkeeping.

The Warfare Environmental Simulator, WES, is currently being made available at the C³ laboratory at the Naval Postgraduate School. The WES program is run at the Naval Ocean Systems Command (NOSC) in San Diego with outlying command centers such as the C³ Lab and CINCPACLFT in Hawaii served by secure data link. WES games will be played on a strategic global or hemispheric scale and will try to approach real time data handling.

WES has several drawbacks for use in a general unclassified wargaming class at NPS. WES must be played in the C³ laboratory and depends on an outside program located in NOSC's computers. The security requirements and limited use of the C³ laboratory rule out extended use of WES at NPS. In addition, scheduling of WES games to support all curricula would require intercurricular scheduling and concurrence of NOSC.
There is therefore a void in interactive computer assisted wargames available at the Naval Postgraduate School. A medium scale naval wargame, tactical in scope and using the computer for bookkeeping functions is needed to fit the requirements of the wargaming courses. An additional requirement would be for a game to aid training of naval officers in tactical decision making. A game is needed that serves the traditional wargames functions of tactical analysis and education.

B. APPROACHES TO THE PROBLEM AND LIMITS OF THE THESIS

In this thesis the existing manual game SEATAG is partially programmed and implemented on the hardware currently installed at the Naval Postgraduate School's computing facility. The program requires less than two cylinders of disk space and is a self-contained game employing two terminals with appropriate screening of information. Changes in game parameters can be entered during overall program execution. The program calculates movement and processes all detections, with actual engagement results and damage determinations computed manually. The program also provides hard copy output for the participants.

C. SELECTION OF SEATAG

Selection of an existing manual game to convert to an interactive computer assisted program rather than the development of a new game was necessitated by this project's
completion date. SEATAG, a publication of the Naval War College, was selected. SEATAG has the advantage of being an official publication with reasonable and accepted assumptions and methodology. The SEATAG model used for this thesis is unclassified and does not contain classified performance data on the various platforms. SEATAG appealed to the author since it emphasized those items considered important without getting involved in a morass of detail.

Although there was a constant temptation to improve SEATAG in writing this thesis, the computer program is as consistent as possible with the rules, procedures and assumptions of the manual game.

Throughout the remainder of the thesis, SEATAG will refer to the manual game in its second edition from the Naval War College and CASTAG, for Computer Assisted SEATAG, will refer to the game as programmed in this thesis.

D. CHAPTER OUTLINE

Chapter II of this thesis examines SEATAG in terms of the playing area, scale, game flow and procedures for engagements and damage. Chapter III describes CASTAG with descriptions of all CASTAG subprograms and models as well as how interactive two terminal operation was achieved. Chapter IV is a user's guide to the CASTAG program with emphasis on the inputs and outputs. Chapter IV with a copy of the SEATAG rules is sufficient documentation to play CASTAG. Chapter V includes comments on playability and suggestions for additions and improvements to the CASTAG program.
II. SEATAG, THE MANUAL GAME

A. HISTORY AND DEVELOPMENT OF SEATAG

SEATAG, Sea Control Tactical Analysis Game, is currently in its second edition. SEATAG is published by the Center for Advanced Research of the Naval War College.

SEATAG was originally developed in 1975 as a research and analytical tool for evaluating the then new Harpoon anti-shipping missile tactics. The second edition is an improved and revised game which is available in an unclassified version.

B. DESCRIPTION OF THE PLAYING AREA, SCALE, ENVIRONMENT AND PLATFORM CHARACTERISTICS OF SEATAG

SEATAG is a manual naval wargame employing two opposing sides Red, Blue, and an umpire Control Group. SEATAG requires three rooms with sufficient desk space for plotting and record keeping, one room for the Control Group and one for each side, Red and Blue. The minimum number of personnel required to play SEATAG is three.

The Control Group performs the umpire functions of intelligence dissemination, detection and damage disclosures and the analysis of interactions during weapons engagements. A Game Director heads the Control Group.

SEATAG comes complete with several grid sheets and cardboard playing pieces to represent force units to be used as the master plot. The grid square scale may vary from
five to thirty nautical miles per grid square side. Locations are coded by a four digit code representing the center of the square and squares can be subdivided into one hundred sub units.

Capabilities of the various platforms are provided on characteristics in Annex C of the SEATAG Rulebook. Characteristics of forty types of modern United States and Soviet ships and aircraft are supplied. The SEATAG data has been taken from editions of *Jane's All the Worlds Fighting Ships* and from *Combat Fleets of the World 1978/1979: Their Ships, Aircraft, and Armament*.

SEATAG is limited to two environmental conditions, "good" weather and "bad" weather. One or both may be used in a game. Bad weather effects aircraft and helicopter operations in the same manner and limits all surface units to 15 knots as well. Bad weather degrades sensor performance in terms of range. In addition, each twenty-four hour day is divided into light, from 0600 to 1800, with the remaining 12 hours dark or night. The only effect of dark is to prohibit visual sightings, and any air operations must be made by IFR (Instrument Flight Rules) capable aircraft.

Rules of engagements with several options are also included in the SEATAG Rulebook.

A supplemental resource variation is included with hypothetical costs for the units. These costs are useful in determining the relative balance of a given scenario.
C. GAME FLOW, AIRCRAFT CARRIER AND SUBMARINE OPERATIONS, AND SURVEILLANCE/INTELLIGENCE PLAY

The game flow is graphically depicted in Figure 1.

The initial decision for each turn is the time duration of that turn. This is specified by the umpires who look ahead to see what the engagement intensity is likely to be. The minimum time step is twenty minutes if the smallest grid size is being used. A one hour minimum is specified for the larger grid scales. Red and Blue then make their moves and specify any engagement intentions to the Control Group who then record the moves and disclose any detections to the appropriate sides. The Red and Blue command centers then adjust moves based on this information. Umpires play out the adjusted moves, play engagements, and provide each side with results, additional detections and the time length of the next move. The adjustment phase allows units to respond to detections as they occur rather than wait until the end of the game move.

The SEATAG rules provide a number of tactical hints on ship positioning, task group organization and submarine warfare. In addition a CV operations section provides a simplified but realistic approach to CV cyclical air operations. An around-the-clock defensive posture is provided along with a selection of surge capabilities against various threats.

Surveillance and intelligence has been distilled into three levels. Each side is assigned or allowed to procure Level I, II, or III. At each day of play at times 0001,
RED/BLUE SIDES

Command centers make moves and specify engagement intentions.

Command centers adjust moves based on surveillance data.

Command centers make next move.

TIME

CONTROL GROUP

Game Director specifies length of the move to each side.

Umpires for each side play out moves and disclose real-time surveillance information to each side.

Umpires play out engagements and disclose all other intelligence and detections obtained during the move. Game Director specifies the length of the next move.

SEAATAG GAME FLOW

Figure 1
0601, 1201, and 1801 a random number is drawn for each surface ship, radiating aircraft and submarine traveling in excess of fifteen knots. In addition another random number is drawn each time a unit uses HF equipment. The number drawn in both cases is used with supplied tables to determine the level of information to be provided to the other side. The level ranges from location information only to complete identification, course and speed. Units in EMCON are less likely to have information divulged than those radiating.

Civilian ships and aircraft may be included in the scenario and are operated by the Control Group. Merchants are not allowed to operate in a military capacity except as part of a convoy.

D. DETECTIONS

A detection opportunity occurs when a unit comes within range of any sensor. The information provided when a detection occurs consists of bearing only for ESM detections and complete information including course, speed, altitude or depth and grid posit for most other sensors. Sonobuoys are an exception, with the amount of information determined by the RN generated when the detection was processed. Additional information, like ship or aircraft type and other characteristics, is left totally to the discretion of the umpires.

E. PLAY OF ENGAGEMENTS AND CALCULATION OF DAMAGE

In SEATAG, engagement analysis is segregated into unit versus unit interactions which are treated in a logical order
determined by when they take place. The Control Group is responsible for this function. The Control Group includes a Red and a Blue chief umpire responsible for playing the engagements for their "side", including tactical decisions as required within the framework of specific operation orders or rules of engagement.

Tables and curves are provided to calculate required information such as targeting, weapon expenditures and number of hits inflicted on a target.

F. TIME UTILIZATION AND SOME OBSERVATIONS ON ACTUAL MANUAL PLAY

SEATAG documentation states that 30 to 60 minutes will be required for the Control Group to process each move or turn. Most likely, 80% of that time is spent plotting and doing movement and Closest Point of Approach (CPA) calculations. However, engagement analysis, once the CPA calculation is made, is relatively straightforward and quick.
III. THE CASTAG PROGRAM

A. OVERALL PROGRAM OPERATION

CASTAG consists of the CASTAG EXEC and three FORTRAN programs - SEARCH, CHANGE, and UPDATE. CASTAG EXEC executes the FORTRAN programs in the specified order. SEARCH processes the input, movement, detection and output functions for one game turn. CHANGE provides the user the option to change any unit parameters prior to the next game move. UPDATE combines each terminal's inputs into an updated data base ready for the next game move.

Prior to execution, each terminal must have access to a disk containing a compiled version of the three FORTRAN programs and the CASTAG EXEC program. In addition, one terminal must have access to the data base.

Figure 2 provides an overall program flowchart showing terminal interaction, file management, and input/output.

1. Overview of CASTAG Inputs

CASTAG inputs can be considered in two groups. First is the data base created by the Control Group. The data base is discussed in Section D of this chapter. Second is the terminal inputs required by CASTAG EXEC, SEARCH, and CHANGE. The program functions associated with these inputs are discussed in this chapter under the appropriate program. Chapter IV discusses these same inputs from the point of view of user data entry and as game parameters.
<table>
<thead>
<tr>
<th>RED TERMINAL</th>
<th>RED EXEC Routine</th>
<th>BLUE EXEC</th>
<th>BLUE TERMINAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enter Blue user ID</td>
<td>READ Blue ID</td>
<td>READ Red ID</td>
<td>Enter Red user ID</td>
</tr>
<tr>
<td>Enter &quot;T&quot; to transfer database**</td>
<td>Transfer file FT02F001 to Blue</td>
<td>Transfer file FT04F001 to Red</td>
<td>Enter &quot;R&quot; to read database</td>
</tr>
<tr>
<td>Enter inputs for program SEARCH</td>
<td>Execute SEARCH which creates file FT02F001</td>
<td>Execute SEARCH which creates file FT02F001</td>
<td>Enter Inputs for program SEARCH</td>
</tr>
<tr>
<td></td>
<td>Offline print file FT02F001</td>
<td>Offline print file FT02F001</td>
<td></td>
</tr>
<tr>
<td>Enter inputs for Program CHANGE</td>
<td>Execute CHANGE</td>
<td>Execute CHANGE</td>
<td></td>
</tr>
<tr>
<td>Enter &quot;R&quot; for Red</td>
<td>Transfer file FT03F001 to Blue</td>
<td>Transfer file FT04F001 to Red</td>
<td>Enter &quot;B&quot; for Blue</td>
</tr>
<tr>
<td>Enter &quot;Y&quot; when file ready to be read</td>
<td>Read file FT04F001 from Blue</td>
<td>Read File FT03F001 from Blue</td>
<td>Enter &quot;Y&quot; when file ready to be read</td>
</tr>
<tr>
<td></td>
<td>Execute UPDATE Rewind FT03, FT04</td>
<td>Execute UPDATE Rewind FT03, FT04</td>
<td></td>
</tr>
<tr>
<td></td>
<td>READ &quot;Y&quot; or &quot;N&quot;</td>
<td>READ &quot;Y&quot; or &quot;N&quot;</td>
<td></td>
</tr>
<tr>
<td>STOP</td>
<td>STOP</td>
<td>STOP</td>
<td>STOP</td>
</tr>
</tbody>
</table>

**Transfer may occur the other way or not at all if both sides have the data base on file.
2. **Dual Terminal Operation**

Two terminal operation can only be accomplished by each terminal accessing and executing identical programs in tandem. The programs are designated Red or Blue by an input at the terminal. At the NPS computer center this is the only manner in which this kind of a two sided game can be played.

3. **Description of the CASTAG EXEC Routine**

CASTAG EXEC executes the three compiled FORTRAN programs and manages the file transfers between terminals.

The CASTAG program begins with a one time interactive sequence in which the User Identification number of the other player is input to identify the other terminal for data transfers.

If required, the data base is transferred to the terminal without it.

The program CHANGE is then executed at each terminal by CASTAG EXEC. CHANGE processes changes entered by the user to a unique data file containing the Red or Blue part of the data base at each terminal. Following an entry to identify the terminal as Red or Blue, an interactive sequence at each terminal manages the cross transfer of these two data files so each terminal has the same information. The UPDATE program is then executed resulting in each terminal having a new but identical data base. Finally, the user is given the option to play another turn or stop.
B. SELECTION OF THE PROGRAMMING LANGUAGES

FORTRAN IV (G) was selected as the programming language for CASTAG because it is a widely used and standardized language. The choice of FORTRAN allows the utilization of CASTAG on other machines and systems in locations other than NPS.

The CASTAG EXEC routine is written in the Control Program - 67/Cambridge Monitoring System (CP/CMS) EXECUTIVE language. Implementation of CASTAG on another computer system not using CP/CMS would require reprogramming in that system's file handling language.

C. FILE SPACE AND CPU TIME REQUIREMENTS OF THE CASTAG PROGRAM

CASTAG uses approximately 186k bytes or 233 records of disk storage. This requirement is compatible with the normal two cylinders of disk space assigned to a user at the NPS Computer Center.

Table 1 lists the program and data files and record lengths which make up the CASTAG program. The source code (FORTRAN) versions are listed for reference but are not required to remain on the disk once compiled. They are not included in the total space requirement.

The largest of the three is SEARCH which takes 40 seconds to compile and from 10 to 12 seconds of CPU time to execute. UPDATE and CHANGE run very vast. The time they take is dependent on the number of user inputs.
Table 1
Cylinder Space Requirements for CASTAG

<table>
<thead>
<tr>
<th>FILE NAME</th>
<th>FILE TYPE</th>
<th>NO. RECORDS (20 units/side)</th>
<th>NO. RECORDS (50 units/side)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CASTAG</td>
<td>EXEC</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>SEARCH</td>
<td>FORTRAN</td>
<td>77</td>
<td>77</td>
</tr>
<tr>
<td>SEARCH</td>
<td>TEXT</td>
<td>63</td>
<td>63</td>
</tr>
<tr>
<td>UPDATE</td>
<td>FORTRAN</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>UPDATE</td>
<td>TEXT</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>CHANGE</td>
<td>FORTRAN</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>CHANGE</td>
<td>TEXT</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>FILE</td>
<td>FT01F001</td>
<td>10</td>
<td>125</td>
</tr>
<tr>
<td>FILE</td>
<td>FT02F001</td>
<td>5</td>
<td>12</td>
</tr>
<tr>
<td>FILE</td>
<td>FT03F001</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>FILE</td>
<td>FT04F001</td>
<td>3</td>
<td>6</td>
</tr>
</tbody>
</table>
D. CREATION AND ORGANIZATION OF THE DATA BASE

The data base contains current game time, the total number of units on each side, and all the parameters for every unit in the game. The data base stores the information from turn to turn to provide game continuity.

The data base is created prior to the start of the game using CP/CMS input and edit functions and placed on a data file. This file is created by the player and must be labeled "FILE FT02F001". Once CASTAG execution has begun, the data base normally is never accessed or changed by the user throughout the entire game.

1. Information Required for the Data Base

   Game start time, number and type of units per side, each unit's initial position, course, and speed are required from the Control Group or Game Director.

   Each unit must have a unique four digit identification number assigned during creation of the data base. In addition, each unit is assigned a second identifying number called a type number. Directions for assigning and decoding these type numbers are located in Chapter IV of the thesis.

   The rest of the parameters required for each unit are found in Annex C of the SEATAG rulebook. Characteristics of two additional units are included in Appendix A of the thesis.

2. Organization and Format of the Data Base

   The first row always contains the current game time in minutes, the number of Red units and the number of Blue
units. Each of the following rows represents one ship, submarine, or aircraft. All Red units are listed first followed by all Blue units. Units are not required to be in any particular order as regards aircraft, ships, or submarines. Each column in a row contains the parameters for a particular unit. Each column's description is provided in Table 2.

The current game time is read from the data base in format F6.0 by the FORTRAN programs. The number of Red units and the number of Blue units is read in format 2I3. All the rest of the data base is read in by row as 15I4. All entries must be right justified. An integer format was selected to facilitate input and output formatting. Table 3 is an example of how a correctly formatted data base appears on file.

The data base shows 6 Red and 6 Blue units. Game time is 420 minutes.

E. PLAYING AREA, TIME AND DISTANCE UNITS

The game area is the positive quadrant of the cartesian plane. A unit's location is expressed as an (x, y) co-ordinate to the nearest nautical mile. The maximum X or Y value is 9999. Units with an X or Y value greater than 1000 will not be plotted on the output.

Game time is computed to the nearest minute and it is therefore possible to play game turns of as little as one minute in duration. Speeds are computed to the nearest whole knot.
TABLE 2
Data Base Column Descriptions

<table>
<thead>
<tr>
<th>ARRAY COLUMN</th>
<th>FIELD</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1-4</td>
<td>Unit Identification number. Each unit in the game must have a unique 1 to 4 digit unit ID. An ID of 0000 is reserved for units not in play. The unit ID is not used by the program except to check if a unit is in play.</td>
</tr>
<tr>
<td>2</td>
<td>5-8</td>
<td>Unit's current speed in knots.</td>
</tr>
<tr>
<td>3</td>
<td>9-12</td>
<td>Altitude or depth of the unit. Altitude for aircraft is in thousands of feet. Surface units must be at 0. Submarines may be deep (2), shallow (1), or surfaced (0).</td>
</tr>
<tr>
<td>4</td>
<td>13-16</td>
<td>Unit's course in degrees true.</td>
</tr>
<tr>
<td>5</td>
<td>17-20</td>
<td>X coordinate of the unit in n.m.</td>
</tr>
<tr>
<td>6</td>
<td>21-24</td>
<td>Y coordinate of the unit in n.m.</td>
</tr>
<tr>
<td>7</td>
<td>25-28</td>
<td>Type identification number. Each digit represents coded information which is tabulated in Table of Chapter IV.</td>
</tr>
<tr>
<td>8</td>
<td>29-32</td>
<td>Unit's maximum speed in knots.</td>
</tr>
<tr>
<td>9</td>
<td>33-36</td>
<td>Surface search radar maximum range in n.m.</td>
</tr>
<tr>
<td>10</td>
<td>37-40</td>
<td>Air search radar maximum range in n.m.</td>
</tr>
<tr>
<td>11</td>
<td>41-44</td>
<td>Sonar maximum range in n.m.</td>
</tr>
<tr>
<td>12</td>
<td>45-48</td>
<td>Status of surface search radar. 1 = On, 0 = Off.</td>
</tr>
<tr>
<td>13</td>
<td>49-52</td>
<td>Status of air search radar. 1 = On, 0 = Off.</td>
</tr>
<tr>
<td>14</td>
<td>53-56</td>
<td>Status of sonar. 1 = On, 0 = Off.</td>
</tr>
<tr>
<td>15</td>
<td>57-60</td>
<td>Status of HF communication equipment. 1 = On, 0 = Off.</td>
</tr>
</tbody>
</table>
Table 3

Sample Data Base

|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 420 | 6 | 6 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 50 | 30 | 0 | 160 | 563 | 4511332 | 34 | 25 | 150 | 10 | 1 | 1 | 1 | 0 |   |   |   |   |   |   |   |
| 51 | 10 | 0 | 160 | 545 | 4881333 | 35 | 20 | 150 | 10 | 0 | 0 | 0 | 0 |   |   |   |   |   |   |   |
| 52 | 10 | 0 | 160 | 547 | 4481346 | 34 | 20 | 150 | 10 | 0 | 0 | 0 | 0 |   |   |   |   |   |   |   |
| 60 | 7 | 1 | 100 | 546 | 4701110 | 30 | 100 | 0 | 30 | 0 | 0 | 0 | 0 |   |   |   |   |   |   |   |
| 0 | 0 | 20 | 90 | 900 | 5001281 | 450 | 200 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |   |   |   |   |   |   |
| 71 | 10 | 1 | 225 | 510 | 5101293 | 120 | 30 | 30 | 2 | 1 | 0 | 1 | 1 |   |   |   |   |   |   |   |
| 101 | 10 | 1 | 95 | 550 | 4732110 | 30 | 10 | 0 | 30 | 0 | 0 | 0 | 0 | 0 |   |   |   |   |   |   |
| 102 | 2 | 2 | 90 | 537 | 4762111 | 34 | 10 | 0 | 30 | 0 | 0 | 0 | 0 | 0 |   |   |   |   |   |   |
| 333 | 11 | 0 | 90 | 495 | 4802333 | 30 | 20 | 150 | 5 | 0 | 0 | 1 | 0 |   |   |   |   |   |   |   |
| 444 | 12 | 0 | 90 | 510 | 4762321 | 33 | 25 | 150 | 5 | 1 | 1 | 0 | 1 |   |   |   |   |   |   |   |
| 555 | 17 | 0 | 90 | 521 | 4912347 | 27 | 20 | 150 | 14 | 1 | 1 | 0 | 0 |   |   |   |   |   |   |   |
| 666 | 30 | 0 | 90 | 537 | 4612347 | 27 | 20 | 150 | 14 | 1 | 1 | 1 | 0 |   |   |   |   |   |   |   |
All distances in CASTAG are expressed in nautical miles (n.m.) and all speeds in knots.

F. DESCRIPTION OF FORTRAN PROGRAM SEARCH

SEARCH models the movement and detection elements for one game turn of the manual game. SEARCH is the first program executed by CASTAG EXEC.

SEARCH consists of a main program and ten subroutines. These are:

1. Subroutine MOVE (TIME)
2. Subroutine NUMBER
3. Subroutine DETECT (TIME)
4. Subroutine SPOOK
5. Subroutine ESM (I)
6. Subroutine DETRAD (I)
7. Subroutine ASONAR (I)
8. Subroutine NLPCPA (I, TIME)
9. Subroutine VISUAL (I, ITYPE)
10. Subroutine PSONAR (I, IPLAT)

The main program will be described first followed by the ten subroutines which make up SEARCH. Each of the eleven sections includes a list of local variables used and a plain language flowchart for that subroutine.

SEARCH uses two computer library routines, UTPLOT and LLRANDOM. UTPLOT is a utility plotting routine available in the SSPLIB library at the NPS Computer Center. This routine will plot more than one set of points on a graph with a different symbol for each call.

LLRANDOM is used to deliver arrays of single precision \( U(0,1) \) distributed pseudo random numbers at various locations in SEARCH. LLRANDOM is accessed by CALL RANDOM.
SEARCH uses an extensive common block, which is identical for all the subroutines. Many variables and arrays in common are used repeatedly throughout the program. Some are used in every subroutine. Definitions of the variables in the common block are found in Table 4. The arrays, all of which are in the common block, are defined in Table 5. Referring back to these two tables will be necessary as the information is not repeated in the subroutine description.

1. MAIN Program

The MAIN program dimensions all arrays, initializes variables, zeros out arrays, reads the data base from the file, reads the user's terminal inputs, and either calls subroutine NUMBER or reads the random number file. Following the call to subroutine MOVE, the data base with the new X,Y locations for each unit is read back onto the original file and two new files are created, each with only Red or Blue data. Next, a call to subroutine DETECT calculates all detections for the game turn. Finally, UTPLOT is utilized to output two plots on the offline printer.

The MAIN program is executed by CASTAG EXEC. MAIN calls three subroutines directly: NUMBER, MOVE, and DETECT. This turn's game time is supplied to MOVE and DETECT as an argument. Library routine UTPLOT is called as follows:

CALL UTPLOT (XPL, YPL, NUN, R, 1, NGRAPH)
Table 4

Description of Variables in the SEARCH Common Block

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DTIME</td>
<td>Current game turn's length in minutes</td>
</tr>
<tr>
<td>EPS</td>
<td>Used to check for zero in the denominator prior to division. Set to .001.</td>
</tr>
<tr>
<td>ID</td>
<td>Tells program whether it is Red or Blue. Also used as third subscript to access the friendly information in array DATA.</td>
</tr>
<tr>
<td>IFOE</td>
<td>Third subscript used to access opponents information in array DATA. Set opposite of ID.</td>
</tr>
<tr>
<td>ISEED1</td>
<td>Random number generator seed used to determine permanent percentage of maximum detection range.</td>
</tr>
<tr>
<td>ISEED2</td>
<td>Random number seed for numbers generated determining HFDF detections.</td>
</tr>
<tr>
<td>ISEED3</td>
<td>Random number seed for numbers generated to determine intelligence detections.</td>
</tr>
<tr>
<td>NRED</td>
<td>Number of Red units. Read from the data base.</td>
</tr>
<tr>
<td>NBLUE</td>
<td>Number of Blue units. Read from the data base.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Maximum number of columns allowed to be used for information storage in the data base and in array DATA. Set to 15. Maximum allowed by array dimension is 20.</td>
</tr>
<tr>
<td>NMAX</td>
<td>Maximum number of units that will have random numbers generated for them. Set to 20. Maximum allowed by array dimension is 50.</td>
</tr>
<tr>
<td>NWRITE</td>
<td>Used to designate the off-line printer as the output device for a FORTRAN WRITE statement. Set to 6.</td>
</tr>
<tr>
<td>ZTIME</td>
<td>Current cumulative game time in minutes. First number read from data base.</td>
</tr>
</tbody>
</table>
Table 5
DEFINITIONS OF ARRAYS IN SEARCH COMMON BLOCK

<table>
<thead>
<tr>
<th>Array Description</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPA (50, 2)</td>
<td>Read array holding calculated ranges in column 1 and bearings in column 2 of CPA's for up to 50 units.</td>
</tr>
<tr>
<td>DATA (50, 20, 2)</td>
<td>Integer array in which the 2nd and subsequent lines of the database are stored. Each row holds the information for one unit, usually subscripted by I. The jth column holds the jth parameter for the Ith unit. The columns have the same meaning as in the database. The K = 1 level is all the Red units' data. The K = 2 level is all the Blue units' data. Note that only 15 of the 20 available spaces for unit data are used.</td>
</tr>
<tr>
<td>DICE (50)</td>
<td>Real array used to hold up to fifty ( U(0,1) ) random numbers from a call to LLRANDOM.</td>
</tr>
<tr>
<td>ESMDET (100, 5)</td>
<td>Integer array holding ESM detection data. Each row holds all the information for a single detection. Columns one through five are interpreted as follows:</td>
</tr>
<tr>
<td></td>
<td><strong>Column</strong></td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>5</td>
</tr>
<tr>
<td>HFDF (50, 4)</td>
<td>Integer array holding HFDF detection data. Each holds all the information for a single detection of a unit. Columns one through four are interpreted as follows:</td>
</tr>
<tr>
<td></td>
<td><strong>Column</strong></td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>4</td>
</tr>
</tbody>
</table>
Table 5 (Continued)

INTEL (50, 6) Integer array holding data revealed during an intelligence segment. Each row contains all the information on a single detected unit. Columns one through six are interpreted as follows:

<table>
<thead>
<tr>
<th>Column</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Unit ID of detected unit</td>
</tr>
<tr>
<td>2</td>
<td>First two digits of type number of detected unit</td>
</tr>
<tr>
<td>3</td>
<td>Second two digits of type number of detected unit</td>
</tr>
<tr>
<td>4</td>
<td>Course of unit detected to nearest 10 degrees</td>
</tr>
<tr>
<td>5</td>
<td>X coordinate of unit detected</td>
</tr>
<tr>
<td>6</td>
<td>Y coordinate of unit detected</td>
</tr>
</tbody>
</table>

MINDEX (10) Integer array. First nine locations not used. MINDEX (10) holds the total number of detected units that will be plotted.

NROW (2) Integer array holding number of units on each side. Set to NROW (1) = NRED, NROW (2) = NBLUE

PRMAX (50, 50, 5) Real array holding the calculated permanent percentage of maximum detection range for each sensor of each unit against every enemy unit. With (I,J,K) as the subscripts, the number represents the Kth sensor on the Ith unit trying to detect the Jth enemy unit. The Kth sensor are:

- 1: Surface search radar
- 2: Air search radar
- 3: ESM
- 4: Active sonar
- 5: Passive sonar and passive sonar ESM

R (4) Real array holding the grid limits of the X and Y values for the plotting routines.

- R(1) = 1000.0 Maximum X
- R(2) = 0.0 Minimum X
- R(3) = 1000.0 Maximum X
- R(4) = 0.0 Minimum Y
RADAR (50, 20) Integer array holding all radar and visual detection data. The first ten columns are used for radar data, the second ten for visual data. Each half row represents the detection of a unit. The columns are interpreted as follows:

<table>
<thead>
<tr>
<th>Column</th>
<th>Radar</th>
<th>Column</th>
<th>Visual</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Unit ID</td>
<td>11</td>
<td>Unit type number</td>
</tr>
<tr>
<td>2</td>
<td>Unit type number</td>
<td>12</td>
<td>Unit altitude</td>
</tr>
<tr>
<td>3</td>
<td>Radar type</td>
<td>13</td>
<td>Course in deg T (air/surf)</td>
</tr>
<tr>
<td>4</td>
<td>Course (T)</td>
<td>14</td>
<td>X coord</td>
</tr>
<tr>
<td>5</td>
<td>X coord</td>
<td>15</td>
<td>Y coord</td>
</tr>
<tr>
<td>6</td>
<td>Y coord</td>
<td>16</td>
<td>Bearing at CPA</td>
</tr>
<tr>
<td>7</td>
<td>Speed</td>
<td>17</td>
<td>Composition</td>
</tr>
<tr>
<td>8</td>
<td>Target composi-</td>
<td>18</td>
<td>Range at CPA</td>
</tr>
<tr>
<td></td>
<td>tion</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Range at CPA</td>
<td>19</td>
<td>Unit ID</td>
</tr>
<tr>
<td>10</td>
<td>Bearing at CPA</td>
<td>20</td>
<td>Not used</td>
</tr>
</tbody>
</table>

SONAR (50, 20) Integer array holding active sonar, passive sonar, and sonar ESM detection data. The first ten columns are used for active detections and the second ten for passive and ESM detections. Each half row represents the detection of a unit. Columns used as follows:

<table>
<thead>
<tr>
<th>Column</th>
<th>Active</th>
<th>Column</th>
<th>Passive</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Unit ID</td>
<td>11</td>
<td>Unit ID</td>
</tr>
<tr>
<td>2</td>
<td>Unit type</td>
<td>12</td>
<td>Target type number</td>
</tr>
<tr>
<td>3</td>
<td>Unit depth</td>
<td>13</td>
<td>Unit depth</td>
</tr>
<tr>
<td>4</td>
<td>Unit course</td>
<td>14</td>
<td>Unit course</td>
</tr>
<tr>
<td>5</td>
<td>X coord</td>
<td>15</td>
<td>X coord</td>
</tr>
<tr>
<td>6</td>
<td>Y coord</td>
<td>16</td>
<td>Y coord</td>
</tr>
<tr>
<td>7</td>
<td>Unit speed</td>
<td>17</td>
<td>Unit speed</td>
</tr>
<tr>
<td>8</td>
<td>Range at CPA</td>
<td>18</td>
<td>Range at CPA</td>
</tr>
<tr>
<td>9</td>
<td>Bearing at CPA</td>
<td>19</td>
<td>Bearing at CPA</td>
</tr>
<tr>
<td>10</td>
<td>Not used</td>
<td>20</td>
<td>Not used</td>
</tr>
</tbody>
</table>

XPLOT (50) Real arrays holding the X coordinates of units to be plotted by UTPLOT.

YPLOT (50) Real arrays holding the Y coordinates of units to be plotted by UTPLOT.
XPLOT and YPLOT are explained in Table 5. R and NUNIT are explained in the local variable definitions. The fifth parameter, 1, tells the routine to plot every (X,Y) point. NGRAPH can equal 1, 2, or 3 indicating if this is the first (1), last (3), or an intermediate (2) set of points to be plotted on a graph.

MAIN reads all the data from disk files required by SEARCH except for three user supplied terminal inputs. These are a random number seed, the length in minutes of the current game turn, and whether the terminal is Red or Blue. These are discussed in Chapter IV. However, the turn length and game time control some additional program functions not apparent to the user.

The program expects to begin the first turn at time 0000. In order to generate the initial PRMAX array either ZTIME must be zero or the turn length (TIME) must be zero. Consequently, to start at an initial game time other than 0000, the length of the first turn must be input as zero. Note that in this case no movement or detections are carried out; only the random numbers are generated by NUMBER. Any time a game turn of zero minutes is played, the only effect is to regenerate the PRMAX array with different values.

Table 6 lists the local variables in this routine. Figure 3 is a flowchart of this routine.
Table 6

MAIN Program, Local Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
<th>File Type</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISK</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IDISK</td>
<td>1</td>
<td>FT01F001</td>
<td>Stores DRMAX array</td>
</tr>
<tr>
<td>NDISK</td>
<td>2</td>
<td>FT02F001</td>
<td>Stores ZTIME, NRED, NBLUE and DATA array</td>
</tr>
<tr>
<td>RDISK</td>
<td>3</td>
<td>FT03F001</td>
<td>Stores Red part of DATA array</td>
</tr>
<tr>
<td>BDISK</td>
<td>4</td>
<td>FT04F001</td>
<td>Stores Blue part of DATA array</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMIN</td>
<td>Number of minutes into latest hour played thus far.</td>
</tr>
<tr>
<td>KSEED</td>
<td>Random number seed entered by user.</td>
</tr>
<tr>
<td>KTIME</td>
<td>Calculated number of hours played thus far.</td>
</tr>
<tr>
<td>M</td>
<td>Number of opposing units detected that are to be plotted on the first plot.</td>
</tr>
<tr>
<td>NGRAPH</td>
<td>Parameter for plotting routine designating number of sets of points on that particular plot.</td>
</tr>
<tr>
<td>NREAD</td>
<td>Input device for FORTRAN READ statement. Set to 5.</td>
</tr>
<tr>
<td>NSHIP</td>
<td>Maximum number of units arrays can hold. Set to 50.</td>
</tr>
<tr>
<td>NTERM</td>
<td>Output device for FORTRAN WRITE statement. Set to 5.</td>
</tr>
<tr>
<td>NUNIT</td>
<td>Number of opposing units to be plotted on second plot.</td>
</tr>
<tr>
<td>TIME</td>
<td>Length of current turn in minutes; entered by user.</td>
</tr>
</tbody>
</table>
Flowchart of SEARCH MAIN Program

Figure 3
2. **Subroutine MOVE**

Subroutine MOVE calculates new X and Y coordinates for each Red and Blue unit as a function of course and distance traveled. If a unit crosses the X or Y axis from the positive quadrant, the Y or X coordinate respectively is set to zero, keeping the unit in the positive quadrant. Equation (1) shows the calculation of a new X coordinate; the new Y coordinate is calculated in a similar manner using the cosine function.

\[
\text{New } X = \text{Old } X + (.5 + \text{Distance} \times \sin (\text{course})) \quad (1)
\]

One-half is added so that when the value is truncated for storage as an integer, it in effect rounds off to the nearest nautical mile.

Subroutine MOVE is called once per game turn from the MAIN program. The turn's length in minutes is passed as an argument. Table 7 defines the local variables and Figure 4 is the flowchart for this subroutine.
Table 7

Subroutine MOVE, Local Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>THOUR</td>
<td>Length of the turn in hours.</td>
</tr>
<tr>
<td>RAD</td>
<td>Converts degrees to radians. Set to $\pi/180$.</td>
</tr>
<tr>
<td>DIST</td>
<td>Distance the Ith unit moved this turn.</td>
</tr>
<tr>
<td>THETA-</td>
<td>Course of the Ith unit converted to radians.</td>
</tr>
<tr>
<td>I</td>
<td>Subscripts the row (unit) in array DATA.</td>
</tr>
<tr>
<td>K</td>
<td>Subscripts array DATA designating Red or Blue.</td>
</tr>
</tbody>
</table>
THOUR = TIME / 60.
RAD = PI / 180.0

For every Red unit I = 1 NUNIT, DO 10

Distance = speed * time
Theta = course * RAD

X = X + .5 + distance * sin (Theta)
Y = Y + .5 + distance * cos (Theta)

If X negative, set X to zero.
If Y negative, set Y to zero.

10 CONTINUE

Repeat above for every Blue unit.

RETURN

Figure 4
Flowchart of Subroutine MOVE (TIME)
3. **Subroutine NUMBER**

Subroutine NUMBER draws a U(0,1) random number and calculates the percent of maximum range at which a detection can occur for every sensor on every unit versus every target unit. A U(0,1) random number is drawn for every possibility, regardless of whether a particular unit has a particular sensor or can ever be detected by a particular sensor.

SEATAG uses an empirically derived curve to manually transform the random number into a percent of maximum range. This curve was approximated by the hyperbola in Equation (1), where \( X \) equals the percent of maximum detection range (PRMAX) and \( Y \) is a U(0,1) random number.

\[
\frac{x^2}{(.9)^2} + \frac{y^2}{(.9)^2} = 1.0 \quad (1)
\]

The hyperbola is graphed with the original curve for comparison in Figure 5.

Note that the \( X \) and \( Y \) axes in Figure 5 have different scales. The maximum error, determined graphically, is 2% of the maximum range. Subroutine NUMBER uses Equation (2) which is Equation (1) solved for \( X \) to calculate the actual PRMAX values.

\[
X = .81 - Y^2 \quad (2)
\]

Subroutine NUMBER is usually called by MAIN only on the first game turn. Thereafter the calculated values are stored from turn to turn on a file.
Figure 5

COMPARISON OF DETECTION CURVES
With the exceptions of subscripts I, J, and K, no local variables are used in this subroutine. Figure 6 is the flowchart for subroutine NUMBER.
FOR K = 1, NMAX (for every Kth sensor) DO 30

FOR J = 1, NMAX (for every Jth target) DO 20

Call RANDOM for 50 U(0,1) random numbers with seed ISEED1 and store in array DICE

FOR I = 1, NMAX (for every Ith unit) DO 10

Set PRMAX(I, J, K) = 0.0

IS random number greater than .9? 

YES

NO

Set PRMAX(I, J, K) = .81 - (DICE(I)^2)

10 CONTINUE

20 CONTINUE

30 CONTINUE

RETURN

Figure 6
Flowchart for Subroutine NUMBER
4. **Subroutine DETECT**

The most important and complex subroutine in SEARCH is the DETECT subroutine. The subroutine comprises three functional sections. First, it must decide if the current turn includes a surveillance opportunity. Next, intelligence from enemy units using HF communication is determined. Last, every eligible friendly unit is given the opportunity to detect eligible targets subject to sensor, speed, and platform limitations.

For HFDF intelligence purposes, the program assumes that each side has procured level I intelligence. Accordingly, the only information provided is the grid position of the enemy unit detected.

DETECT performs the umpire function of deciding which sensors may be used against which targets under what circumstances. Some ground rules for this are implicit in the flowchart and some explicitly explained in the following paragraphs.

The program gives submarines at shallow depth some of the capabilities associated with a submarine at periscope depth. In particular, shallow submarines may make ESM and surface search (S/S) radar detections. This represents the ability of the submarine to extend its ESM and S/S radar mast above the water in the same manner as the periscope. Except for ESM against radar emissions, a shallow submarine is still undetectable by means other than sonar. The program does not allow for visual detection by periscope due to the limited maximum range at which such a detection could occur.
Fighter, strike (attack), interceptors and ECM aircraft may only make visual detections. In general radars on board these aircraft are fire control radars without a 360 degree search capability. Their ESM equipment is primarily short range threat warning oriented and is used to provide a warning of fire control or missile homing acquisition or lock on. Also these aircraft are never detectable by ESM in the program due to the narrow sector of the emitters.

Arbitrary limits of ten knots maximum speed and 1000 feet or less altitude were imposed on the SH-3 and KA-25 helicopters before they may attempt a dipping sonar detection.

Subroutine DETECT is called once per turn from the MAIN program. Subroutine DETECT calls the following subroutines.

1. SPOOK
2. ESM (I)
3. DETRAD (I)
4. ASONAR (I)
5. NLPCPA (I, TIME)
6. VISUAL (I, ITYPE)
7. PSONAR (I, IPLAT)

The I argument passed to all the subroutines called except SPOOK designates the row subscript of the unit searching with that particular sensor.

The second argument passed to NLPCPA is the same length of the current turn passed to DETECT. The second
argument passed to VISUAL and PSONAR indicates what kind of unit the Ith unit is.

Table 8 contains the local variables used in DETECT and Figure 7 is a flowchart for DETECT.
Table 8
Subroutine DETECT, Local variables

<table>
<thead>
<tr>
<th>VARIABLE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>HR</td>
<td>Counter for six hour surveillance periods.</td>
</tr>
<tr>
<td>ITYPE</td>
<td>Last three digits (or last two if an aircraft) of the Ith units type number.</td>
</tr>
<tr>
<td>I</td>
<td>Subscript of Ith friendly unit in do loop 90, Ith enemy unit in do loop 16.</td>
</tr>
<tr>
<td>IUNIT</td>
<td>Total number of friendly units.</td>
</tr>
<tr>
<td>LIGHT</td>
<td>Set by the program to 0 = Dark, 1 = Daylight as a function of game time.</td>
</tr>
<tr>
<td>N</td>
<td>Counter for the number of HFDF detections.</td>
</tr>
<tr>
<td>NN</td>
<td>Total number of enemy units.</td>
</tr>
<tr>
<td>NX</td>
<td>Used in do loop 11 to calculate six hour periods for surveillance periods.</td>
</tr>
<tr>
<td>Z2</td>
<td>The time in minutes the previous turn ended.</td>
</tr>
</tbody>
</table>
Flowchart of Subroutine DETECT

Figure 7
Increment N by 1
Set values in the Nth row of HFDF to those of the Ith unit
Put co-ordinates of detected unit into X:LOT and YPLOT

16 CONTINUE

Set MINDEX(10)=N
If there were any detections output the HFDF array on the terminal

Set LIGHT=0
Set LIGHT=1 if ZTIME between 0600 and 1800 hours
Set IUNIT to the number of friendly units

For each I=1, IUNIT friendly units DO 90
If ID=0, unit not in play

Set ITYPE to the last three digits of the type number

Is unit a sub?

Calculate CPA of all target units (Call NLPCPA (I))

If sonar is passive (off) and speed less than 23 knots, Call PSONAR (I,1) to determine any passive sonar detections

If sub is shallow or surfaced, Call ESM(I) to determine ESM detections

If sub is shallow, surfaced and s/s radar is on, Call DETRAD (I) to determine radar detection
If sub is surfaced and it is day, Call VISUAL (I) to determine visual detections

If sonar is on and speed is \( \leq 15 \) knots, Call ASONAR (I)

to determine active sonar detections

GO TO 90

Is unit a surface ship?

NO

YES

Call NLCPA (I)

Call SMS (I)

If day, Call VISUAL (I, ITYPE)

If either A/S or S/S radar on

Call DETRAD (I)

If sonar off and speed \( \lt 18 \),

Call P(SONAR(I,2))

If sonar on and speed \( \lt 16 \)

Call ASONAR (I)

GO TO 90

50
Set ITYPE to last two digits of ITYPE

Call NLPCPA (I,TIME)
If daylight, Call VISUAL (I, ITYPE)

If unit is fighter attack or interceptor

Call ESM (I)
Call DETRAD (I)

Is aircraft a class 3 helo? (SH3 or KA-25)

If helo speed LE 30 KTS, and ALT. LE 1000 ft
Call PSONAR (I,5)

90 CONTINUE

END
5. **Subroutine SPOOK**

This subroutine is played daily at 0001, 0601, 1201, and 1801 hours. In game terms this is an abstracted representation of sources of military intelligence available beyond the level of command represented in the game. The CASTAG program only plays level I of the three surveillance levels available in SEATAG.

All ships, submarines going over 15 knots and radiating aircraft are subject to detection. A U(0,1) random number is drawn for each unit and if that unit is in one of the three preceding categories, it may be detected. If the unit is in EMCON, i.e., emitting no radiation including sonar, the chance of detection is reduced twenty percent.

If a unit is detected, grid location and general/type information is provided. In some cases additional information consisting of the units course to the nearest 10 degrees and the last two digits of the unit's type number is provided. This information is a condensed version of Table II-5 in the SEATAG documentation.

Subroutine SPOOK is called at most once per game turn from subroutine DETECT only if the current game turn included one of the four times that surveillance is played.

Table 9 describes the local variables used in this subroutine. Figure 8 is a flowchart of subroutine SPOOK.
Table 9
Subroutine SPOOK, Local Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>Row subscript of the unit being checked.</td>
</tr>
<tr>
<td>ID2</td>
<td>Set to first two digits of the Ith unit's type number.</td>
</tr>
<tr>
<td>J</td>
<td>Varying column subscript of array DATA in do loop 9.</td>
</tr>
<tr>
<td>MCON</td>
<td>Sum of the number of active emitters on a unit. MCON = 0 means unit is in EMCON.</td>
</tr>
<tr>
<td>N</td>
<td>Counter for number of surveillance detections.</td>
</tr>
<tr>
<td>NUNIT</td>
<td>Set to total number of enemy units.</td>
</tr>
</tbody>
</table>
Set NUNIT to the number of enemy units
Set N to zero
Generate 50 U(0,1) R.N. using RANDOM

For each I = 1, NUNIT enemy unit DO 11
  Set MCON = 0

If the Ith R.N. is LT. .59
  NO

If the Ith unit is not in play
  NO

Add one to MCON for each active sensor or communications device on the Ith unit

Flowchart of subroutine SPOOK

Figure 5
If 1th unit is in EMCON and the 1th R.N. is L.T. .79

NO

Is the 1th unit a sub and less than 15 knots?

NO

Is the 1th unit an aircraft in EMCON?

NO

A surveillance detection has occurred. Set N=N+1

TO C

TO D
C

If the
Ith R.N. is GT.
.8

YES

NO

Put grid posit and broad type
information into array INTELL
and XPLOT, YPLOT

FROM 11

Put additional specific type
and heading information into
array INTEL for the Ith unit

11 CONTINUE

Increment MINDEX(10) by N
Write detections, if any,
on the terminal

RETURN

RETURN

RETURN

RETURN
6. **Subroutine ESM**

Subroutine ESM determines all possible ESM detections by the Ith friendly unit of radar emissions from each enemy unit.

The maximum range for ESM detections is calculated as twice the maximum range of the emitter times the percent of maximum range:

\[
\text{Range} = 2 \times \text{PRMAX} (I,N,3) \times (\text{Radar Range})
\]

The subscripts for PRMAX are I denoting the Ith detecting unit, N the Nth target and 3 the ESM sensor. This is compared to the range of the CPA to determine if a detection occurs. Detections are stored in array ESMDET.

This subroutine is called from DETECT when the Ith unit is eligible to make ESM detections. Subroutine NLPCPA must have been called previously from DETECT so that the range and bearing of the CPA of every enemy unit is available in array CPA. The percent of maximum range for the Ith units ESM system must be available in array PRMAX.

Table 10 contains the local variables used in this routine. Figure 9 is a flowchart of subroutine ESM.
Table 10
Local Variables of Subroutine ESM

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>Argument passed from DETECT denoting the Ith friendly unit as the unit attempting to make the detections.</td>
</tr>
<tr>
<td>II, JJ</td>
<td>Row and column subscripts for printing out array ESMDET.</td>
</tr>
<tr>
<td>M</td>
<td>Counter for number of ESM detections for the Ith unit.</td>
</tr>
<tr>
<td>N</td>
<td>Row subscript of DATA denoting the Nth enemy unit.</td>
</tr>
<tr>
<td>NTYPE</td>
<td>Last three digits of the Nth unit's type number.</td>
</tr>
<tr>
<td>NUNIT</td>
<td>Set to total number of enemy units.</td>
</tr>
<tr>
<td>RANGE</td>
<td>Range at which detection may take place.</td>
</tr>
</tbody>
</table>
Flowchart of Subroutine ESM(I)

Figure 5
Is the range at CPA .GT. the calculated RANGE?

Increment M by 1
Put information from Nth unit into Mth row of ESMDET array

Is the Nth unit's air search radar on?

Calculate RANGE for A/S radar

Is range at CPA .GT. RANGE?

TO D

TO E
Increment M by 1
Put information from Nth unit into Mth row of ESMDET

If there were any detections...

Output player's version of ESMDET to the terminal
Output umpire's version of ESMDET to the offline printer

RETURN
7. **Subroutine DETRAD**

Subroutine DETRAD checks all possible radar detections of enemy units for the Ith friendly unit. Air search radars are only allowed to detect aircraft. Surface search radars only detect ships and surfaced submarines. The detection range of both air and surface search radars is calculated as the maximum range times the percent of maximum range from array PRMAX.

A detection takes place when the detection range of the radar is greater than the CPA of the target. For submarines, this range is then halved making them more difficult to detect. Detections are stored sequentially in the first ten columns of integer array RADAR.

The only argument passed to DETRAD is I. I specifies the row of the friendly unit attempting to make the radar detections and is used as the row subscript to access information in array DATA.

DETRAD is called from DETECT whenever a unit has either an air or surface search radar on. Subroutine NLPCPA must have been executed last for unit I. Values from PRMAX with the third subscript 1 for surface search and 2 for air search radars must be available.

Table 11 lists the local variables of DETRAD and Figure 10 is a flowchart of the routine.
## Table 11
DETTRAD, Local Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARANGE</td>
<td>A/S Radar calculated final detection range.</td>
</tr>
<tr>
<td>I</td>
<td>External argument passed from DETECT. Designates the Ith unit which is doing the detecting.</td>
</tr>
<tr>
<td>II, JJ</td>
<td>Row and column subscripts for printing out first 10 columns of array RADAR.</td>
</tr>
<tr>
<td>ITGT</td>
<td>Last three digits of the Nth unit's type number.</td>
</tr>
<tr>
<td>J</td>
<td>Column subscript in do loop 30 for transferring detection data to array RADAR.</td>
</tr>
<tr>
<td>M</td>
<td>Counter for total number of detections for unit I.</td>
</tr>
<tr>
<td>N</td>
<td>Subscripts the row, of the Nth enemy unit.</td>
</tr>
<tr>
<td>NUNIT</td>
<td>Set to total number of enemy units.</td>
</tr>
<tr>
<td>SRANGE</td>
<td>S/S radar calculated final detection range.</td>
</tr>
</tbody>
</table>
FOR each N=1, NUNIT enemy unit, DO 50

Is Nth unit in play?

YES

SetITGT to last 3 digits of type number

Is Ith unit's S/S radar on?

NO

YES

TO A

TO B

TO C
A

Is the Nth target on the surface?

YES

Calculate SRANGE = PRMAX * Radar MAX range

If target is a submarine, Set SRANGE = \( \frac{1}{2} \) SRANGE

NO

Is CPA of target less than SRANGE?

YES

Increment M by 1
Put information on Nth unit into the Mth row of RADAR & into XPLOT and YPLOT

NO

TO D

TO D

65
Is it unit's air search radar on?

Is the target an aircraft?

Calculate ARANGE

Is target CPA less than ARANGE?

Increment X by 1
Put information on the Nth unit into the Nth row of RADAR and into XPLOT and YPLOT

50 CONTINUE

FROM

50
If there were any detections...

Increment MINDEX(10) by M

Output player's version of RADAR on the terminal. Output Umpire's version of RADAR on the offline printer.

RETURN
8. **Subroutine VISUAL**

Subroutine VISUAL processes all possible visual detections of enemy units for the Ith friendly unit. Ships and surfaced submarines were arbitrarily assigned an unmodified visual detection range of 10 n.m. Aircraft were assigned a 20 n.m. range. Although an aircraft or ship can discern a target at more than the above ranges, visual detection within the framework of SEATAG implies classification opportunities as well.

Classification implies the aircraft or ship closes to read hull numbers and count masts. Rather than try to make the information provided a function of range, 10 and 20 miles were selected as reasonable values considering the aggregate level of the game. All units with the exception of submerged subs are detectable by this routine.

A detection occurs when the calculated CPA of the target is less than the assigned visual detection range. The information on the detection is stored in columns 11-20 of array RADAR.

The two arguments passed to VISUAL are I which designates the row in array DATA of the unit attempting to make the detection and ITYPE which tells the subroutine if the Ith unit is an aircraft or surface unit.

This subroutine is called from DETECT when the Ith unit is an aircraft, ship, or a surfaced submarine and game time is between 0600 and 1800 hours. Subroutine NLPCPA must have been executed last for unit I.
Table 12 lists the local variables and Figure 11 is a flowchart of subroutine VISUAL.
### Table 12
Subroutine VISUAL, Local Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>Argument passed from DETECT denoting the Ith unit doing the searching.</td>
</tr>
<tr>
<td>II, JJ</td>
<td>Row and column subscripts for printing out columns 11-20 of array RADAR.</td>
</tr>
<tr>
<td>ITYPE</td>
<td>Last two digits of Ith unit type number if aircraft, last three digits if unit submarine or ship.</td>
</tr>
<tr>
<td>M</td>
<td>Counter for number of visual detections for the Ith unit.</td>
</tr>
<tr>
<td>N</td>
<td>Row subscript denoting the Nth enemy unit.</td>
</tr>
<tr>
<td>NTYPE</td>
<td>Last three digits of Nth unit's type number.</td>
</tr>
<tr>
<td>NUNIT</td>
<td>Total number of enemy units.</td>
</tr>
<tr>
<td>RANGE</td>
<td>Set to 10 n.m. if Ith unit on surface, 20 n.m. if Ith unit an aircraft.</td>
</tr>
</tbody>
</table>
Set $RANGE=10$
$M=0$
$NUNIT=NROW$ (IFOE)

If Ith unit is an A/C set $RANGE=20$

For each $N=1,NUNIT$ DO 50

Is the Nth unit in play?

YES
Calculate NTYPE

Is the Nth unit a submerged sub?

YES

TO A

Flowchart of Subroutine VISUAL

Figure 11
Increment M by 1
Put data from the Nth unit into the Mth row of array
RADAR and into XPLT/YPLOT

50  CONTINUE

Increment MINDEX (10) by M
If there were any detections.

Output players version of
RADAR on terminal
OUTPUT umpire version of
RADAR on offline printer

RETURN
9. **Subroutine ASONAR**

ASONAR, for active sonar, determines possible active sonar detections by the Ith friendly unit. The unit's sonar must be active and the unit's speed less than 15 knots prior to calling ASONAR.

Surface ships, submarines and helicopters with dipping sonars may make active detections. Only submerged submarines, either friendly or enemy, may be detected by an active sonar. No differentiation is made between a friendly or enemy submarine detection.

Level 5 of PRMAX is used for active sonar detections of own units although it is normally used in determining passive performance. This is to avoid correlating active sonar performance against the Nth enemy unit with that against the Nth friendly unit. This avoids adding another level \( K = 6 \) to array PRMAX for what amounts to a rare occurrence.

The only argument passed to ASONAR is \( I \). This specifies the row of the friendly unit attempting to make the active sonar detections and is used as the row subscript to access the Ith unit's data in array DATA.

ASONAR is called from DETECT. Values for percent of maximum range from array PRMAX with the third subscript equal to four and five are required before the subroutine is called. NLPCPA must have been called last for the Ith unit prior to calling ASONAR.

Prior to the second time through do loop 60, NLPCPA is called from ASONAR. After IFOE is set to ID, NLPCPA
calculates the CPA of all friendly units and stores the values in array CPA. This is the only instance of a subroutine being called from a subroutine other than from DETECT in program SEARCH. Note that ASONAR must be the last subroutine called for detections by the Ith unit in DETECT due to its calling NLPCPA as the values in array CPA will be those of friendly units.

Detections for the Ith unit are stored in the first ten columns of array SONAR by row. Table 13 lists the local variables of ASONAR. Figure 12 is a flowchart of ASONAR.
Table 13
Subroutine ASONAR, Local Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>Argument passed from DETECT denoting the Ith unit attempting active sonar detections.</td>
</tr>
<tr>
<td>IA</td>
<td>Column subscript in do loop 40 to transfer detection data to array SONAR.</td>
</tr>
<tr>
<td>II, JJ</td>
<td>Row and column subscripts for printing out array SONAR.</td>
</tr>
<tr>
<td>ISAVE</td>
<td>Saves value of IFOE while IFOE = ID.</td>
</tr>
<tr>
<td>ITGT</td>
<td>Last three digits of the Nth unit's type number.</td>
</tr>
<tr>
<td>K</td>
<td>Counts twice through do loop 60.</td>
</tr>
<tr>
<td>KUNIT</td>
<td>Set to total number of friendly units.</td>
</tr>
<tr>
<td>M</td>
<td>Counter for total number of active sonar for the Ith unit.</td>
</tr>
<tr>
<td>N</td>
<td>Row subscript denoting the Nth enemy when K = 1 or friendly when K = 2 unit.</td>
</tr>
<tr>
<td>NUNIT</td>
<td>Set to total number of enemy units when K = 1, total number of friendly units when K = 2.</td>
</tr>
<tr>
<td>RANGE</td>
<td>Active sonar calculated final maximum detection range.</td>
</tr>
<tr>
<td>TIME</td>
<td>Argument passed to NLPCPA denoting this turn's length in minutes.</td>
</tr>
</tbody>
</table>
Flowchart of Subroutine ASONAK

Figure 12
Is target a sub not on the surface?

Yes

RANGE = PRMAX * sonar max range

Is CPA of target greater than RANGE?

Yes

Increment M by 1
Put data from the Nth unit into the Mth row of array SONAR

No

50 CONTINUE

FROM 50

50 CONTINUE

TO C
C

If this is the second time through the 60 Loop

FROM 60

IFOE = ID
NUNIT = KUNIT
Call NLP CPA (I, TIME)

60 CONTINUE

IFOE = ISAVE

If there were any detections,
Print out players version of SONAR on the terminal

Print out umpires version of SONAR on the offline printer
Put X, Y co-ord of detections in X PLOT and Y PLOT

Increment MINDEX (10) by M

RETURN

78
10. **Subroutine NLPCPA**

NLPCPA calculates the range in n.m. and the bearing in degrees of the CPA of every enemy unit to the Ith friendly unit.

The final CPA is in fact the CPA during the last turn. In most instances this turns out to be the initial or final position of the Nth unit, depending on if that unit is opening or closing the Ith unit.

The subroutine uses each unit's (X,Y) coordinates, course and speed as well as the elapsed time and clock time of this turn. Recall that since subroutine MOVE is called before NLPCPA, the values of the coordinates are where the units are after the move.

The method used in this subroutine to determine the CPA takes the approach commonly used underway where the relative position of the target is known and the target's course, speed and CPA are determined from that. This will appear somewhat round about as the target course and speed are already available and a normal trigonometric approach would be feasible.

Point (A,B) is the position of the Ith unit at time ZTIME. (X2,Y2) is the relative as well as the geographic position of the Nth unit at ZTIME. Both of these points are available in array DATA. A previous relative position of the Nth unit to the Ith unit must be calculated. This point (X1,Y1), will be calculated as the relative position of the target at time (ZTIME-TIME), or the game time
at the beginning of this turn. \((X_1,Y_1)\) is not the position of the target at the beginning of this turn. It is a relative position at which the target may never have been. A geometric diagram of the calculation of \((X_1,Y_1)\) is provided as Figure 13.

To calculate \((X_1,Y_1)\) the X and Y distance traveled by the target is subtracted from point \((X_2,Y_2)\) giving the position of the target at the start of this turn. Then the X and Y distance traveled by the Ith unit is added, resulting in \(X_1\) and \(Y_1\):

\[
X_1 = X_2 - D_1 \times \text{SIN(THETA)} + X\text{DIST} \tag{1}
\]

\[
Y_1 = Y_2 - D_1 \times \text{SIN(THETA)} + Y\text{DIST} \tag{2}
\]

To determine the CPA, the problem is to find the shortest distance from point \((A,B)\) to the line defined by the two relative points \((X_1,Y_1)\) and \((X_2,Y_2)\). This line is called the DRM for Direction of Relative Movement. The slope is called \(DX\). Formulating the problem as a non-linear programming problem gives

\[
\min f(x,y) = \sqrt{(X-A)^2 + (Y-B)^2} \tag{3}
\]

subject to \(Y - Y_1 = DX \times (X - X_1)\) \tag{4}

Using the LaGrange multiplier technique, we minimize the square function minus lambda times the constraint
\[ X_1 = X_2 - \text{XDIST} - D_1 \times \sin(\theta) \]
\[ Y_1 = Y_2 - \text{YDIST} - D_1 \times \cos(\theta) \]

**CALCULATION OF \((X_1,Y_1)\)**

Figure 13
(Eq. 5) which will have the same minimum as Equation 3.

\[
F(x,y) = (x-A)^2 + (y-B)^2 - L * (y-y_l - DX(x-x_l))
\]

Taking partial derivatives yields

\[
\frac{\partial F}{\partial x} = 2x - 2A + L * DX = 0 \quad (6)
\]

\[
\frac{\partial F}{\partial y} = 2y - 2B - L = 0 \quad (7)
\]

\[
\frac{\partial F}{\partial L} = DX * (x-x_l) + y_l - y = 0 \quad (8)
\]

Equation (7) gives an expression for L and (8) gives an expression for Y in terms of X. Substituting these in (6) and solving for X yields

\[
x = \frac{(A + B * DX + X_l * DX^2 - y_l * DX)/(1 + DX^2)}
\]

Equation (9) is how X is calculated in NLPCPA. Y is computed from (8). (X,Y) is the coordinate of the CPA.

In the calculation of DX, if the absolute value of \(x_2 - x_l\) is within .001 of zero, the CPA calculation is not needed since DX is infinite or nearly so. Practically speaking in this case, \(x_2 = x_l\) and the DRM is vertical, i.e., a CPA of \((y_2, b)\).
If both \((X_2 - X_1)\) and \((Y_2 - Y_1)\) are less than \(EPS\), then \(X_2 = X_1\) and \(Y_2 = Y_1\) implying the two units involved are on the same course and speed. The CPA is set to \((X_2, Y_2)\) in this case.

In Figure 13, the CPA falls conveniently between \((X_1, Y_1)\) and \((X_2, Y_2)\). In fact we have three cases to consider in deciding on the final CPA. Figure 14 shows the three possibilities. The relative positions of \((X_1, Y_1)\) and \((X_2, Y_2)\) may or may not be reversed, giving six possibilities to check.

Two series of four logical FORTRAN IF statements branch to assign the appropriate final value for \((X, Y)\).

Once the final \((X, Y)\) is determined, the distance from \((X, Y)\) to \((A, B)\) is calculated using the standard cartesian distance formula. Distances less than \(EPS\) are set to \(EPS\) to avoid dividing by zero when calculating the bearing of the CPA. The CPA range for the \(N\)th unit is stored in \(CPA(N, l)\).

Finally the bearing of the point \((X, Y)\) from \((A, B)\) must be calculated and converted to degrees. \(\text{T\alpha\mu}\) is the angle between line segment \(((A, B), (X, Y))\) and \(((X, B), (X, Y))\) and is calculated using the \text{AR\Sigma\NIN} function as

\[
\text{T\alpha\mu} = \text{AR\Sigma\NIN}((X - A)/\text{CPA}(N, l)) \tag{10}
\]

In the first quadrant \(\text{T\alpha\mu}\) will be the correct bearing. If \((X, Y)\) is in one of the other three quadrants
Relative Locations of DRM Endpoints

Figure 14

Case | CPA Point
--- | ---
1   | (X,Y)
2   | (X2,Y2)
3   | (X1,Y1)
of (A,B) TAU must be modified as shown in Figure 15. Finally TAU is converted to degrees and stored in CPA(N,2). The argument I passed to NLPCPA specifies the row of the friendly unit. The CPA's are calculated relative to this unit. TIME is the elapsed time for this turn.

This subroutine is called from DETECT once for each Ith (friendly) unit that is able to make any ESM, radar or sonar detections. In addition, NLPCPA may be called from ASONAR in circumstances defined in that subroutine.

Figure 16 is a flowchart for NLPCPA and Table 14 lists the local variables.
Four Quadrant Tau Calculation

Figure 15
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Current X coordinate of the Ith unit.</td>
</tr>
<tr>
<td>B</td>
<td>Current Y coordinate of the Ith unit.</td>
</tr>
<tr>
<td>D1</td>
<td>Calculated distance the Nth unit moved this turn.</td>
</tr>
<tr>
<td>D2</td>
<td>Calculated distance the Ith unit moved this turn.</td>
</tr>
<tr>
<td>DX</td>
<td>Calculated slope of the DRM line.</td>
</tr>
<tr>
<td>ETA</td>
<td>Course of the Ith unit in radians.</td>
</tr>
<tr>
<td>I</td>
<td>Argument denoting the Ith unit.</td>
</tr>
<tr>
<td>NUNIT</td>
<td>Set to total number of enemy units.</td>
</tr>
<tr>
<td>RAD</td>
<td>Set to (\pi/180). Used as degree to radian conversion factor.</td>
</tr>
<tr>
<td>T</td>
<td>Calculated time of this turn in hours.</td>
</tr>
<tr>
<td>TAU</td>
<td>Angle of the line from (A,B) to (X,Y) with the line from (X,Y) to (X,B).</td>
</tr>
<tr>
<td>THETA</td>
<td>Course of the Nth unit in radians.</td>
</tr>
<tr>
<td>TIME</td>
<td>Argument containing the elapsed time of this turn in minutes.</td>
</tr>
<tr>
<td>X</td>
<td>X coordinate of the calculated and final CPA.</td>
</tr>
<tr>
<td>X2</td>
<td>Current X coordinate of the Nth unit.</td>
</tr>
<tr>
<td>XDIST</td>
<td>Distance along the X-axis the Ith unit moved this turn.</td>
</tr>
<tr>
<td>Y</td>
<td>Y coordinate of the calculated and final CPA.</td>
</tr>
<tr>
<td>Y1</td>
<td>Y coordinate of the relative position of the Nth unit to the Ith at the beginning of this turn.</td>
</tr>
</tbody>
</table>
Table 14 (Continued)

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y2</td>
<td>Current Y coordinate of the Nth unit.</td>
</tr>
<tr>
<td>YDIST</td>
<td>Distance along the Y-axis the Ith unit moved this turn.</td>
</tr>
</tbody>
</table>
START

T = Time / 60
RAD = Pi / 180
NUNIT = NROW (IFOE)
A = DATA (I, 5, ID)
B = DATA (I, 6, ID)

Calculate for the Ith unit,
D2 = TIME (T) * speed
ETA = course * Pi / 180
XDIST = D2 * SIN (ETA)
YDIST = D2 * COS (ETA)

For N=1, NUNIT DO 50

FROM 50

Calculate for the Nth unit
THETA = course * Pi / 180
D1 = T * speed
X2 = DATA (N, 5, IFOE)
Y2 = DATA (N, 6, IFOE)

Calculate (X1, Y1), the relative (to the Ith unit) position that the Nth unit would have been in at ZTIME - TIME as follows:

X1 = (X co-ord of Nth unit now) + (X Distance Ith unit traveled this turn) - (X Distance Nth unit traveled), similarly for Y1
Y1 = Y2 + YDIST - D1 * COS (THETA)

Flowchart of Subroutine NLPCPA
Figure 16
Is the denominator of the slope equation close to 0?

YES

NO

Calculate \( \text{DY} = Y_2 - Y_1 / X_2 - X_1 \), the slope of the line between \((X_1, Y_1)\) and \((X_2, Y_2)\).

Calculate \((X, Y)\), the co-ord of the CPA

See if the calculated CPA \((X, Y)\) occurs on the line segment between \((X_1, Y_1)\) and \((X_2, Y_2)\). If not, set X, Y to the closest endpoint

YES

NO

Is \((X, Y)\) closer to \((X_1, Y_1)\) than to \((X_2, Y_2)\)?

YES

NO

\( \theta \)

To C

To \( \beta \)

Slope of DRK is infinite Set \( X = X_2 \) \( Y = B \)

Units on same CSE and SPD and at CPA

Is \( Y \) closer to \( Y_1 \) than to \( Y_2 \)?

YES

NO

\( \gamma \)

To C
Calculate Distance from (X,Y) to (A,B) and store in CPA(N,1)

If CPA is less than EPS, set CPA to EPS.

Calculate TAU, the angle made by line segments ((A,B),(X,Y)) and ((X,Y),(A,B)) as ARSIN(((X-A)/(CPA(N,1))))

If (X,Y) in fourth quadrant of (A,B), set TAU=TAU+2Pi

If (X,Y) in the second quadrant of (A,B), set TAU=Pi-TAU

If (X,Y) bears 180 from (A,B), set TAU=Pi

If (X,Y) in the third quadrant of (A,B), set TAU=Pi-2Pi

Convert TAU to degrees and store in CPA(N,2)

TO DO

CONTINUE

RETURN
11. **Subroutine PSONAR**

Subroutine PSONAR determines all possible passive sonar detections by the Ith friendly unit. Ships and submarines at any depth attempt passive detections if their sonar is off (passive) and their speed is less than 17 and 22 knots respectively.

There are two kinds of passive detections possible in this subroutine. First, an active sonar may be detected with a passive sonar. This is analogous to an ESM detection of a radar. Range is calculated as:

\[
\text{Range} = 2 \times \text{(Percent effectiveness)} \times \text{(Range of active sonar)}
\]

If this is less than the calculated CPA, a detection occurs. Sonar ESM detections are stored in the last 10 columns of SONAR. An approximate range is provided to the detecting unit for an ESM passive detection. This is calculated as:

\[
\text{Approximate Range} = \text{CPA Range} + \text{CPA Range} \times (\text{DICE (N)} - 0.5)
\]

DICE (N) is a U(0,1) random number. The approximate range will be uniformly distributed between one-half and 1.5 the range at CPA.

The second type of passive detection is the passive tracking of a target by sonar. The target's range, bearing, course, and speed are determined by the tracking unit. Any target, with the exception of aircraft, may be detected.
To calculate if a passive detection has occurred, the unit's maximum detection range is calculated as a function of target speed, searcher speed and where the target and searcher are with respect to the layer.

The curves in the top half of Figure 17 are used when a shallow sub is attempting to detect a unit on the surface or two submerged subs are attempting to detect one another at the same depth. This is called Case 1. The curves were approximated by a family of ellipses whose parameters are functions of target speed. The solid curves are those given in the SEATAG manual and the dashed curves those of the ellipse approximations.

The ellipse parameters for Case 1 are calculated as follows:

\[
\begin{align*}
A & = \frac{(\text{Target Speed})}{5} + 15 \\
B & = \text{Target Speed} + 2 \\
X & = \text{Searching unit speed}
\end{align*}
\]

For the ellipse

\[
\frac{X^2}{A^2} + \frac{Y^2}{B^2} = 1
\]

Solving for \(Y = B \times (1 - (X^2/A^2))\) gives the maximum passive sonar detection range in n.m. calculated by the sub-routine. The maximum range error is plus or minus 3 n.m. from the original curves.
Comparison of Passive Sonar Detection Curves

Figure 17
The curves in the bottom half of Figure 17 represent a cross layer detection situation. This is referred to as Case 2. They are used for any unit on the surface trying to detect any submerged submarine, for deep subs trying to detect surface units, and for two submerged submarines at different depths trying to detect each other.

A series of ellipses was also used to approximate the original curves. The values of X and Y remain unchanged. The A and B parameters for Case 2 are calculated differently as shown below:

\[
A = \text{Target speed} + C \times (15 - \text{Target speed})
\]

where

\[
C = .4 \quad \text{if} \quad \text{target speed} < 16
\]
\[
C = .6 \quad \text{if} \quad \text{target speed} > 15 \quad \text{and} \quad \leq 24
\]
\[
C = .7 \quad \text{if} \quad \text{target speed} > 25
\]

and

\[
B = \text{Target speed} \quad \text{if} \quad \text{target speed} < 21
\]

or

\[
B = (\text{Target speed}/2) + 10 \quad \text{if} \quad \text{target speed} \geq 21
\]

The following table gives the A and B parameters as a function of target speed for 5 knot increments:
Case 2

<table>
<thead>
<tr>
<th>TGT SPD</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>9</td>
<td>5</td>
</tr>
<tr>
<td>10</td>
<td>12</td>
<td>10</td>
</tr>
<tr>
<td>15</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>20</td>
<td>17</td>
<td>20</td>
</tr>
<tr>
<td>25</td>
<td>18</td>
<td>22.5</td>
</tr>
<tr>
<td>30</td>
<td>19.5</td>
<td>25</td>
</tr>
</tbody>
</table>

In Case (2) the vertical marks where the curves are truncated at a maximum of an 18 knot searcher speed. This in effect means much of the poorer fitting 30 knot curve is not used. The worst distortion is for a 17 knot search speed against a 25 knot target which results in a 7 n.m. range vice the 5 n.m. range obtained using the curves manually.

In some cases it is possible the searched's speed will be greater than the calculated A values. In this case no detection should be possible so A is set equal to zero, insuring Y will be zero. If the calculated Y value is greater than the listed sonar max range on the characteristic card, Y is set to the maximum range from the characteristic card. This happens most frequently with surface ship short range sonars.

Once Y is determined, it is multiplied by the percent of maximum range (the PRMAX value) to give the final range. This is then compared with the CPA to determine if
a detection did in fact take place. Detections are then stored sequentially by row in the first ten columns of array SONAR.

The argument I passed to PSONAR specifies the row of the friendly unit attempting to make the passive detections and is used in PSONAR to access information for the Ith unit for array DATA. The argument IPLAT designates the searching unit as a surface ship or submarine.

Subroutine ASONAR is called from subroutine DETECT. Subroutine NLPCPA must have been previously called last for unit I. Values from array PRMAX with the third subscript \( K = 5 \) are used for the percent of maximum range.

Table 15 provides a list of local variables and Figure 18 a flowchart for subroutine PSONAR.
Table 15
Subroutine PSONAR, Local Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Parameter of ellipse in standard form.</td>
</tr>
<tr>
<td>B</td>
<td>Parameter of ellipse in standard form.</td>
</tr>
<tr>
<td>B2</td>
<td>Used to calculate B as a function of target speed.</td>
</tr>
<tr>
<td>C</td>
<td>Used to calculate A as a function of target speed.</td>
</tr>
<tr>
<td>I</td>
<td>Argument passed from DETECT denoting the Ith unit searching passively.</td>
</tr>
<tr>
<td>IA</td>
<td>Column subscript in do loop 33 used to transfer data on the Mth detection into array SONAR.</td>
</tr>
<tr>
<td>II, JJ</td>
<td>Row and column subscripts for printing out columns of array SONAR.</td>
</tr>
<tr>
<td>IK</td>
<td>Column subscript used in do loop 55 to transfer the MMth passive detection data into array SONAR.</td>
</tr>
<tr>
<td>IPLAT</td>
<td>Argument passed from DETECT denoting the type of searching platform, 1 for a submarine, 2 for a surface unit.</td>
</tr>
<tr>
<td>M</td>
<td>Counter for the number of passive sonar ESM detections.</td>
</tr>
<tr>
<td>ML</td>
<td>Lower bound on the row subscript of array SONAR when printing out passive detections.</td>
</tr>
<tr>
<td>MM</td>
<td>Counter for the number of passive sonar detections. Starts from M.</td>
</tr>
<tr>
<td>N</td>
<td>Row subscript denoting the Nth enemy unit used in do loop 35 and 60.</td>
</tr>
<tr>
<td>NTGT</td>
<td>Last three digits of the Nth unit's type number.</td>
</tr>
<tr>
<td>NUNIT</td>
<td>Set to total number of enemy units.</td>
</tr>
<tr>
<td>RANGE</td>
<td>Calculated final detection range in n.m.</td>
</tr>
</tbody>
</table>
Table 15 (Continued)

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>Searchers speed (Ith units speed in knots).</td>
</tr>
<tr>
<td>Y</td>
<td>Calculated detection range before modification by PRMAX.</td>
</tr>
</tbody>
</table>
M = 0
NUNIT = NROW (IFOE)
For every N=1, NUNIT DO 35

Is the Nth unit out of play?

YES

Is the Nth unit's sonar active?

NO

NO

Calculate RANGE equal to 2 * PRMAX * sonar range

YES

Is the CPA of the Nth unit less than RANGE?

NO

YES

Increment M by 1
Put information on Nth unit into Mth row of array SONAR

35 CONTINUE

Figure 18

Flowchart of Subroutine PSONAR
If there were any detections, output players version of SONAR on the terminal and umpires version on the offline printer

For each N = 1, NUNIT DO 60

Is the Nth unit out of play?

Calculate NTGT

Is the Nth target an aircraft?

TO B

TO C
Is the 1th (searching) unit a submerged submarine?

- **YES**
  - Is the Nth target a surface ship and the searcher shallow?
    - **NO**
    - Are the target and searcher at the same depth?
      - **YES**
        - B2=2
        - C=.5
        - A=TGT SPD/5+15
      - **NO**
    - Is searcher speed greater than 18?
      - **YES**
        - B2+(TGT SPD - 20)/-2
        - IF TGT SPD .LE. 20
        - B2=0.0
      - **NO**
        - C=.4
          - If TGT SPD .GT. 15
          - set C=.6, then set
          - A=C * (15 - TGT SPD) + TGT SPD

\[ B = \text{TGT SPD} + B2 \]
\[ X = \text{Searcher speed} \]

If \( X \gt A \),
Set \( A = X \)

Calculate
\[ Y = B^2 \times \sqrt{1 - (X/A)^2} \]

If \( Y \gt \text{searcher's maximum sonar range} \),
Set \( Y = \text{maximum sonar range} \)

\[ \text{RANGE} = Y \times \text{PRMAX} (I,N,5) \]

Is \( \text{CPA of Nth unit} \ \gt \text{RANGE} \)?

Increment \( MM \) by 1
Put information on the Nth unit into the MMth row of SONAR

\[ 62 \ \text{CONTINUE} \]
If there were any detections, set \( M_1 = M_1 + 1 \).

Print out on the terminal the players version of SONAR. Output the umpires version of SONAR on the offline printer.

RETURN
G. DESCRIPTION OF FORTRAN PROGRAM CHANGE

CHANGE allows the player to change a unit's course, altitude or depth and turn radars, sonar, and HF communications on or off. Units may be brought in and out of play by changing their ID. Units brought into play may have their X and Y coordinates changed to insure they enter play at an appropriate location. Units out of play are set to (0,0). CHANGE is executed once per game turn by CASTAG EXEC.

Integer inputs that are incorrect for a particular parameter of a unit are corrected as follows:

1. Speeds greater than listed maximum speed are set to unit's max speed.

2. An altitude other than 0 for a surface unit will be set to 0.

3. Any course greater than 360 will be set to 360.

A unit which has had a non-existent sensor activated will not affect the detection routines. The maximum range for that sensor will be zero in the data base resulting in a consistent zero detection range.

ID numbers and X or Y coordinates are not screened in any way and must be corrected by cycling that particular unit back through the change procedure. Figure 19 is a flowchart of CHANGE and Table 16 is a list of the variables used in CHANGE.
<table>
<thead>
<tr>
<th>Variable/Array</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATA (50, 20)</td>
<td>Integer array holding either Red or Blue data in the same format as in the data base.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Number of columns being used for data in array DATA. Set to 15.</td>
</tr>
<tr>
<td>NREAD</td>
<td>Used to designate the terminal as the read device for FORTRAN READ statements. Set to 5.</td>
</tr>
<tr>
<td>NWRITE</td>
<td>Used to designate the terminal as the write device for FORTRAN WRITE statements. Set to 6.</td>
</tr>
<tr>
<td>NROW</td>
<td>Maximum number of units (rows) array DATA will hold. Set to 50.</td>
</tr>
<tr>
<td>ZTIME</td>
<td>Game clock time in minutes. Printed out for reference.</td>
</tr>
<tr>
<td>IDENT</td>
<td>Input by user. 1 designates Red; 2 designates Blue.</td>
</tr>
<tr>
<td>IDISK</td>
<td>DSRN designating which file to access for information. Set to IDENT + 2, giving 3 for Red, 4 for Blue.</td>
</tr>
<tr>
<td>N</td>
<td>Input by user telling the program what column of array DATA will be changed.</td>
</tr>
<tr>
<td>ITYPE</td>
<td>Last three digits of Nth units type number.</td>
</tr>
<tr>
<td>IDNO</td>
<td>Row subscript of DATA designating that unit will have its ID number changed.</td>
</tr>
<tr>
<td>I</td>
<td>Used throughout the program as a variable row subscript for array DATA.</td>
</tr>
<tr>
<td>J</td>
<td>Used throughout the program as a variable column subscript for array DATA.</td>
</tr>
</tbody>
</table>
Initialize variables and zero array DATA
Read IDENT from terminal

Set IDISK = IDENT + 2
Read FILE FT03F001 if Red, FILE FT04F001 if Blue
Read N from terminal

N = 0?

Make changes to course, speed, alt/depth and status of sensors for unit N
Check changes for validity

GO TO read N

Figure 19
Flowchart of FORTRAN Program CHANGE

107
NO
Change ID of unit IDNO and enter new X,Y coord if necessary

YES
GO TO read IDNO

IDNO = 0

REWIND IDISK

Write ZTIME, NROW and DATA on FT03F001 or FT04F001

STOP
H. DESCRIPTION OF FORTRAN PROGRAM UPDATE

UPDATE combines two files into the new data base prior to the next execution of SEARCH. UPDATE is executed once at the end of the game turn by CASTAG EXEC. UPDATE READS the two files which contain the user entered changes to the data base (FT03F001 and FT04F001) into two arrays. File FT02F001 is prepared to be written on from the top by a REWIND command which destroys the existing information. File FT02F001 was the data base which had been saved up to this point to provide a backup if either of the two new files were inadvertently destroyed during execution of CHANGE.

Arrays RDATA and BDATA are written back on FT02F001 in the same format as the old data base. FT02F001 and FT04F001 are rewound. Note that should the users have entered different turn lengths during execution of SEARCH at the terminals, ZTIME will be fixed for next turn at both terminals at whatever was entered by the Blue player.

Table 17 lists the variables and arrays in UPDATE and Figure 20 is a flowchart of UPDATE.
<table>
<thead>
<tr>
<th><strong>Variable/Array</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>NCOL</td>
<td>Number of columns being used in the data structure. Set to 15.</td>
</tr>
<tr>
<td>NROW</td>
<td>Maximum number of units (rows) per side. Set to 50.</td>
</tr>
<tr>
<td>NRED, NBLUE</td>
<td>Total number of RED and BLUE units, respectively.</td>
</tr>
<tr>
<td>ZTIME</td>
<td>Total game time in minutes.</td>
</tr>
<tr>
<td>NDISK</td>
<td>DSRN for the data base. Set to 2 designating file FT02F001 for READ and WRITE statements.</td>
</tr>
<tr>
<td>RDISK, BDISK</td>
<td>Integers designating FT03 and FT04 as READ files.</td>
</tr>
<tr>
<td>RDATA (50, 20)</td>
<td>RED half of the original data base which has had the changes made.</td>
</tr>
<tr>
<td>BDATA (50, 20)</td>
<td>BLUE half of the original data base which has had the changes made.</td>
</tr>
<tr>
<td>I</td>
<td>Row subscript for RDATA and BDATA.</td>
</tr>
<tr>
<td>J</td>
<td>Column subscript for RDATA and BDATA.</td>
</tr>
</tbody>
</table>
START

Declare RDATA, BDATA integers
Dimension arrays and set initial values

Read ZTIME and NRED from FILE FT03FO01

Zero out arrays

Read FILE FT03FO01 into array RDATA

Read ZTIME and NBLUE from FILE FT04FO01

Read the rest of FILE FT04FO01 into array BDATA

REWIND FILE FT02FO01, destroying the information
in the data base

Write ZTIME, NRED, NBLUE on the first record
of FILE FT02FO01

Write RDATA on FILE FT02FO01

Write BDATA on FILE FT02FO01

REWIND FILES FT03FO01 and FT04FO01

RETURN

Flowchart of FORTRAN Program UPDATE

Figure 20

111
IV. USER'S GUIDE TO CASTAG

This chapter discusses terminal inputs, program outputs, and program recovery procedures. Some familiarity with SEATAG is assumed as well as some knowledge of FORTRAN format "I" and "F" input type declarations. The line and letter delete instructions refer specifically to the DECWRITER T-300 terminals in the NPS Computer Center. Other terminals should have analogous functions.

A. CASTAG TERMINAL INPUTS

The terminals have two keys which can be used to correct input errors before returning the carriage. The "@" symbol is used to delete the previous character in an input line. More than one "@" can be used per line. For example, entering a format 112 parameter, "1A" is typed. To correct the illegal character "A", type "@2" resulting in "1A@2" which will be read as "12" following the carriage return. The "\" symbol typed at the end of a line will cause the terminal to delete the line following the carriage return allowing the user to re-enter the line.

To begin execution of CASTAG, enter:

$ CASTAG

the terminal will print:

THIS IS THE SEATAG EXEC ROUTINE.
THIS PROGRAM WILL REQUIRE A DATA BASE WHICH HOLDS ALL THE APPROPRIATE SHIP AND AIRCRAFT CHARACTERISTICS.
ENTER THE USER NUMBER OF THE OTHER PLAY INCLUDING P OR G FOR USER TYPE, MAKING A TOTAL OF 5 DIGITS, E.G; "1059P".

The ">" symbol means the terminal is waiting for an input to CASTAG EXEC. An illegal input will cycle an EXEC program back to the input instruction. In this example, the user entered "1059P". The terminal replies with:

IS OTHER PLAYERS ID = 1059P ? ENTER Y OR N.

>Y

Answering with "N" would give the user a chance to re-enter the user ID. Following a "Y" entry, the terminal prints:

FILE FT02F001 IS THE DATA BASE. THIS FILE MUST BE SENT TO THE CONSOLE OF THE OTHER PLAYER IF NO COPY EXISTS IN HIS FILES. IF YOU ARE THE OTHER PLAYER, YOU WILL HAVE TO READ FILE FT02F001. IF NO TRANSFER OF DATA BASE REQUIRED ENTER N. TO TRANSFER DATA BASE TO OTHER PLAYER ENTER "T". TO READ THE DATA BASE FROM THE OTHER PLAYER ENTER "R".

>R

The example shows the user wants to read the data base from the other terminal.

FILES:- NO RDR, NO PRT, NO PUN
IF YOU HAVE A FILE SPOOLED WAITING TO BE READ, ENTER "Y". IF NOT ENTER ANY OTHER LETTER EXCEPT N. THE PROGRAM WILL CONTINUE TO RETURN HERE UNTIL THE FILE IS AVAILABLE OR UNTIL Y OR N IS ENTERED. TO GO AHEAD WITHOUT READING THE FILE ENTER N.

The message shows no files waiting to be read. Consequently, an entry of any letter except "Y" or "N" should be made. Either or both of the messages will indicate that there is a spooled file waiting. These are:

** CARDS XFERED BY 1059P05 **
Assuming the user had entered "N", and then received one of the two messages above, the terminal repeats:

IF YOU HAVE A FILE SPOOLED WAITING TO BE READ, ENTER "Y". IF NOT ENTER ANY OTHER LETTER EXCEPT N. THE PROGRAM WILL CONTINUE TO RETURN HERE UNTIL THE FILE IS AVAILABLE OR UNTIL Y OR N IS ENTERED. TO GO AHEAD WITHOUT READING THE FILE ENTER N.

Entry of "Y" will output a verification of the file read:

OFFLINE READ FILE FT02 F001 P1

An entry of "T" will print one message:

** CARDS XFERED TO 0000G **

and skip the preceding sequence.

Entry of "N" at any point skips the file transfer section and proceeds with program execution, even if there is no data base at one or both terminals.

SEARCH requires three entries. The first is the random number seed, of up to seven integers. A zero may be entered
in which case the program sets the random number seeds. Different random number seeds should be entered at the two terminals. The next entry is the duration of the next turn in minutes. Both terminals should enter the same value. The maximum value is 1440 minutes or 24 hours and the minimum is zero. Only the time specified by the Control Group should be entered. Last, a single digit entry, "1" or "2", tells the program whether it is Red or Blue. Following these entries, a printout of detection information for each side is printed out at the terminal. Interpretation of this output is covered in Section B of this chapter.

Below is an example of the inputs to SEARCH:

EXECUTION BEGINS...
ENTER THE RANDOM NUMBER SEED, FORMAT I7.
>1219503
ENTER THE TIME, IN MINUTES, OF THIS TURN, F4.0.
>60
ENTER "1" IF YOU ARE THE RED PLAYER, "2" IF BLUE. FORMAT I11.
>2
The entries show a seed of "1219503", a sixty minutes turn length and that this terminal is Blue this turn.

Conclusion of program SEARCH and the start of the next FORTRAN program is indicated by

"EXECUTION BEINGS... ENTER 1 FOR RED, 2 FOR BLUE, FORMAT I11".

The next FORTRAN program, CHANGE, has begun execution. This is an interactive sequence that allows the user to change parameters and bring units in and out of play. The initial access is by row, so the first unit in the data base is
unit "1". There can be up to 50 units so these entries are made in I2 format. A code number (2 through 8) is input to determine which value is to be changed followed by the new value when requested. All changes to a unit should be made first before moving on to another unit although it is possible to cycle through the changes as many times as required. Appendix B provides an example of an actual CHANGE execution.

Following entry of the last change to the last unit, program control is returned to CASTAG EXEC. The next sequence transfers the files altered by CHANGE.

NOW EACH PLAYER SENDS HIS UPDATED FILE TO THE OTHER PLAYER. TO INITIATE TRANSFER, ENTER "R" IF YOU ARE RED, "B" IF BLUE.

> b
YOU ENTERED B. IS THIS CORRECT? ENTER "Y" OR "N".

> w
The above sequence tells the CASTAG program if it is Red or Blue.

Two messages will indicate a file waiting to be read as in the database transfer.

** CARDS XFERED BY 1059P05 **
** CARDS XFERED TO 1059P **
FILES:- 01 RDR, NO PRT, NO PUN
DO YOU HAVE A FILE SPOOLED TO BE READ? ENTER "Y" OR "N".

> y
OFFLINE READ FILE FT03F001 P1

The XFERED BY" message may appear while still executing CHANGE. This means the other terminal user has finished his changes and initiated transfer. The other player still must wait to read your change file before continuing. The rest of the
sequence is carried out as if "R" was entered in the
transfer at the beginning of the program.

After both transfers have been completed, a list of
files is output for comparison to four filetypes. Each
terminal should have all four before proceeding to the
next turn.

<table>
<thead>
<tr>
<th>FILENAME</th>
<th>FILETYPE</th>
<th>MODE</th>
<th>NO.REC.</th>
<th>DATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>FILE</td>
<td>FT01F001</td>
<td>P1</td>
<td>10</td>
<td>2/19</td>
</tr>
<tr>
<td>FILE</td>
<td>FT02F001</td>
<td>P1</td>
<td>2</td>
<td>2/19</td>
</tr>
<tr>
<td>FILE</td>
<td>FT04F001</td>
<td>P1</td>
<td>1</td>
<td>2/19</td>
</tr>
<tr>
<td>FILE</td>
<td>FT03F001</td>
<td>P1</td>
<td>1</td>
<td>2/19</td>
</tr>
</tbody>
</table>

YOU SHOULD HAVE FILE TYPES FT02F001, FT03F001, FT04F001
AND FT01F001 ON YOUR DISK AT THIS TIME.

CASTAG executes UPDATE and prints:

TO PLAY ANOTHER TURN, ENTER Y. TO QUITE ENTER N.

This entry lets the user go back to SEARCH to play another
turn or to terminate the program.

B. DECODING/ENCODING TYPE NUMBERS

Each unit in CASTAG is given a four digit code for
identification as to the unit's side, platform, kind and
class. These numbers are called type numbers and are
assigned when the data base is created. They are decoded
by the participants to determine information about specific
detections. Table 18 provides definitions for the first
three digits. Table 19 gives the fourth digit assignment
for all the units currently available in CASTAG.
<table>
<thead>
<tr>
<th>DIGIT</th>
<th>NAME</th>
<th>VALUES</th>
<th>DEFINITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>First</td>
<td>Side</td>
<td>0</td>
<td>Neutral</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>Red</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>Blue</td>
</tr>
<tr>
<td>Second</td>
<td>Platform</td>
<td>1</td>
<td>Submarine</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>Aircraft/helicopter</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>Surface vessel</td>
</tr>
<tr>
<td>Third</td>
<td>Kind</td>
<td>0</td>
<td>Merchant or civilian</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>Submarine</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>Aircraft carrier</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>Large surface combatant (CG,CLG,DDG)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>Small surface combatant (FF,FFG, PG)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>Service force/amphibious force (AOE)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6</td>
<td>Fighter, Interceptor, Attack Aircraft</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7</td>
<td>ASW fixed wing aircraft (S-3,P-3)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8</td>
<td>Bomber, Recon or AEW aircraft</td>
</tr>
<tr>
<td></td>
<td></td>
<td>9</td>
<td>Helicopter</td>
</tr>
<tr>
<td>BLUE UNITS</td>
<td>NUMBER</td>
<td>CLASS</td>
<td></td>
</tr>
<tr>
<td>----------------------------</td>
<td>--------</td>
<td>------------------</td>
<td></td>
</tr>
<tr>
<td>Submarines</td>
<td>1</td>
<td>Los Angeles</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Sturgeon</td>
<td></td>
</tr>
<tr>
<td>Aircraft carrier</td>
<td>1</td>
<td>USS America</td>
<td></td>
</tr>
<tr>
<td>CG, CLG, DDG, DD</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Largest surface combatants)</td>
<td>1</td>
<td>Leahy</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>C. F. Adams</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>Coontz</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>Barry</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>Spruance</td>
<td></td>
</tr>
<tr>
<td>FF, FFG, PG, PGM</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Small surface combatants)</td>
<td>1</td>
<td>Perry</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Brooke</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>Knox</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>Garcia</td>
<td></td>
</tr>
<tr>
<td>Service force/Amphibious</td>
<td>1</td>
<td>Sacramento</td>
<td></td>
</tr>
<tr>
<td>Fighter/Attack aircraft</td>
<td>1</td>
<td>KA-6D Tanker</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>EA-6B ECM support</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>A-6 Corsair</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>A-7 Intruder</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>Phantom II</td>
<td></td>
</tr>
<tr>
<td>ASW fixed wing aircraft</td>
<td>1</td>
<td>S-3A</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>P-3C</td>
<td></td>
</tr>
<tr>
<td>Bomber, Recon, AEW aircraft</td>
<td>1</td>
<td>E-2C</td>
<td></td>
</tr>
<tr>
<td>Helicopters</td>
<td>1</td>
<td>SH-3H</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>SH-2D</td>
<td></td>
</tr>
<tr>
<td>RED UNITS</td>
<td>NUMBER</td>
<td>CLASS</td>
<td></td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>--------</td>
<td>--------------------</td>
<td></td>
</tr>
<tr>
<td>Submarines</td>
<td>1</td>
<td>Charlie SSGN</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Victor SSN</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>Echo II SSGN</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>November SSN</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>Juliett SSG</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>Foxtrot SS</td>
<td></td>
</tr>
<tr>
<td>Aircraft carriers</td>
<td>1</td>
<td>Kiev</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Moskva</td>
<td></td>
</tr>
<tr>
<td>CG, CLG, DDG, DD</td>
<td>1</td>
<td>Kara</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Kresta I</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>Kresta II</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>Krivak</td>
<td></td>
</tr>
<tr>
<td>FF, PFG, PG, PGM</td>
<td>1</td>
<td>Nanuchka</td>
<td></td>
</tr>
<tr>
<td>Service/Amphibious force</td>
<td>1</td>
<td>Chilikin (AOE)</td>
<td></td>
</tr>
<tr>
<td>Fighter/Attack aircraft</td>
<td>1</td>
<td>MIG 23</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>YAK 36 Forger</td>
<td></td>
</tr>
<tr>
<td>ASW fixed wing</td>
<td></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Bomber/Recon aircraft</td>
<td>1</td>
<td>Bear</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Badger</td>
<td></td>
</tr>
</tbody>
</table>
C. CASTAG TERMINAL OUTPUT

Following execution of SEARCH, the detection information for that side is output on the terminal. If there were no detections by a particular unit or no detections of a certain type there is no output. It is possible for there to be no detections, and hence no output, at all. All detections, with the exception of active sonar, are of opposing units.

The first output is surveillance information available only at 0001, 0601, 1201 and 1801 game time. The information is presented as follows:

<table>
<thead>
<tr>
<th>UNIT TYPE</th>
<th>X-COORD</th>
<th>Y-COORD</th>
<th>PLATFORM</th>
<th>COURSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>21</td>
<td>535</td>
<td>476</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>23</td>
<td>504</td>
<td>491</td>
<td>47</td>
<td>95</td>
</tr>
</tbody>
</table>

Unit type is the first two digits for the type number. In the example above, "21" is a Blue submarine and "23" is a Blue surface unit. The (X,Y) position is the current one. The platform is the last two digits of the type number. "0" indicates the information is not available. The course, if provided, is rounded to within 10 degrees of the true course.

Any HF detections are output next in the following format:

HF DETECTIONS OF ENEMY COMMS
X COORD  Y COORD
510     510

This example informs the user there was an enemy HF transmission originating from (510, 510) during this turn.
Each unit, in row order, will now have all of its detections printed. The output order varies with the platform as shown below:

<table>
<thead>
<tr>
<th>ORDER</th>
<th>SUBMARINE</th>
<th>SURFACE</th>
<th>AIRCRAFT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Passive Sonar</td>
<td>ESM</td>
<td>Visual</td>
</tr>
<tr>
<td>2</td>
<td>ESM</td>
<td>Visual</td>
<td>ESM</td>
</tr>
<tr>
<td>3</td>
<td>Radar</td>
<td>Radar</td>
<td>Radar</td>
</tr>
<tr>
<td>4</td>
<td>Visual</td>
<td>Passive Sonar</td>
<td>Active Sonar</td>
</tr>
<tr>
<td>5</td>
<td>Active Sonar</td>
<td>Active Sonar</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Passive sonar detections have the following format:

**DETECTIONS OF ACTIVE SONARS BY UNIT 111**

<table>
<thead>
<tr>
<th>DEPTH</th>
<th>BEARING</th>
<th>APPROX RANGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>89</td>
<td>10</td>
</tr>
</tbody>
</table>

**PASSIVE DETECTIONS BY UNIT 1011**

<table>
<thead>
<tr>
<th>COURSE</th>
<th>SPEED</th>
<th>DEPTH</th>
<th>RANGE</th>
<th>BEARING</th>
</tr>
</thead>
<tbody>
<tr>
<td>160</td>
<td>30</td>
<td>0</td>
<td>12</td>
<td>89</td>
</tr>
<tr>
<td>100</td>
<td>7</td>
<td>1</td>
<td>2</td>
<td>199</td>
</tr>
</tbody>
</table>

Course, speed, and depth are as per the information in the database. The bearing is in both cases the bearing at CPA, not necessarily the most current bearing. The passive detection range is the range at CPA. The approximate range for ESM detections of active sonars is a uniformly distributed random variable with the mean at the range at CPA. These detections are by unit ID number 111.

All ESM detections appear in the following format:

**ESM INTERCEPTS BY UNIT 333**

<table>
<thead>
<tr>
<th>RADAR</th>
<th>BEARING</th>
<th>PLATFORM</th>
</tr>
</thead>
<tbody>
<tr>
<td>13</td>
<td>91</td>
<td>33</td>
</tr>
<tr>
<td>12</td>
<td>26</td>
<td>29</td>
</tr>
</tbody>
</table>

RADAR TYPE 12 = SURFACE SEARCH TYPE 13 = AIR SEARCH
In the above example unit 333 has detected an air search radar (type 13) and a surface search radar. Both bearings are the radar bearings at CPA. The platform number is the middle two digits of the type number. "33" represents a Red CG/CLG and "29" is a Red helo.

Air and surface search radar detections appear jointly by unit. In the following example, unit 666 has detected two targets. The "12" designates the S/S radar and "13" the A/S radar. Notice that although the air target has a speed of 10 knots it must still be an air target of some type. It could be a helicopter of an aircraft presumed to be orbiting. "COMP" is the number of targets in the unit or its composition. All detections are single targets so this information is superfluous.

RADAR DETECTIONS BY UNIT 666

<table>
<thead>
<tr>
<th>RADAR TYPE</th>
<th>COURSE</th>
<th>X COORD</th>
<th>Y COORD</th>
<th>SPEED</th>
<th>COMP.</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>160</td>
<td>547</td>
<td>548</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>13</td>
<td>225</td>
<td>510</td>
<td>510</td>
<td>10</td>
<td>1</td>
</tr>
</tbody>
</table>

Active sonar detections may be either friendly or enemy submarines. An example of a detection printout is provided below.

ACTIVE SONAR DETECTIONS BY UNIT 50

<table>
<thead>
<tr>
<th>DEPTH</th>
<th>COURSE</th>
<th>X COORD</th>
<th>Y COORD</th>
<th>SPEED</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>95</td>
<td>535</td>
<td>473</td>
<td>10</td>
</tr>
<tr>
<td>1</td>
<td>100</td>
<td>534</td>
<td>472</td>
<td>7</td>
</tr>
</tbody>
</table>

Unit 50 has detected two submarines in close proximity. A depth of "1" indicates they are both shallow while a "2" would indicate deep. (X,Y) position, course and speed are as previously explained.
Visual identifications, although only occurring at short range, provide reasonably complete target identification. In the example provided, unit 333 has two detections. Complete type information identifies "1332" as a Red CG of the KARA class. "1293" is a KA-25 Hormone helicopter. The altitude in the helicopter's case is 1000 feet. Bearing is still the bearing at CPA. The rest of the information is self explanatory.

<table>
<thead>
<tr>
<th>TYPE</th>
<th>ALT</th>
<th>COURSE</th>
<th>X COORD</th>
<th>Y COORD</th>
<th>BEARING</th>
<th>COMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>1332</td>
<td>0</td>
<td>245</td>
<td>419</td>
<td>479</td>
<td>355</td>
<td>1</td>
</tr>
<tr>
<td>1293</td>
<td>1</td>
<td>255</td>
<td>497</td>
<td>476</td>
<td>244</td>
<td>1</td>
</tr>
</tbody>
</table>

An example of the terminal output for every kind of detection has been provided. This information is in fact what the players will be basing their tactical decisions on. Section D provides a similar discussion of the offline printer output.

D. CASTAG OFFLINE PRINTER OUTPUT

Each terminal prints five sections in order. Each section is usually no more than a page in length. The contents and use of each section is discussed in order of appearance.

A representative form the Control Group should collect the printouts and distribute sections two and three to their respective sides. Both copies of Sections one, four, and five should go to the Control Group.
1. **Umpire Summary of Detections**

   This output matches that of each terminal with the same detections in the same order as the originating terminal. The Control Group will need one from each terminal to get a complete picture of the preceding turn. All the information on the terminal printout is present with the addition of the ID number of the unit detected, the range and bearing of CPA and other pertinent information. The CPA range is especially important for playing engagements as this will be the range for all firing calculations. An example of the information provided in this printout is reproduced below.

   **UMPIRE COPY, PASSIVE SONAR DETECTIONS, UNIT 51**
   
<table>
<thead>
<tr>
<th>ID</th>
<th>SPD</th>
<th>DEPTH</th>
<th>CSE</th>
<th>X</th>
<th>Y</th>
<th>TYPE</th>
<th>RGE</th>
<th>BRG</th>
</tr>
</thead>
<tbody>
<tr>
<td>555</td>
<td>17</td>
<td>0</td>
<td>90</td>
<td>548</td>
<td>491</td>
<td>2347</td>
<td>5</td>
<td>60</td>
</tr>
</tbody>
</table>

   **UMPIRE VERSION OF VISUAL SIGHTINGS, UNIT 71**
   
<table>
<thead>
<tr>
<th>TYPE</th>
<th>ALT. COURSE</th>
<th>X COORD</th>
<th>Y COORD</th>
<th>BRG</th>
<th>COMP</th>
<th>CPA</th>
<th>RNG</th>
<th>ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>2333</td>
<td>0</td>
<td>90</td>
<td>513</td>
<td>480</td>
<td>67</td>
<td>1</td>
<td>11</td>
<td>333</td>
</tr>
</tbody>
</table>

   **ESMDET, UMPIRE VERSION ID = 71**
   
   **RADAR BEARING PLATFORM ID NO. CPA RANGE**
   
<table>
<thead>
<tr>
<th>RADAR</th>
<th>BEARING</th>
<th>PLATFORM ID NO.</th>
<th>CPA RANGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>88</td>
<td>32</td>
<td>444</td>
</tr>
<tr>
<td>13</td>
<td>88</td>
<td>32</td>
<td>444</td>
</tr>
<tr>
<td>13</td>
<td>70</td>
<td>34</td>
<td>555</td>
</tr>
<tr>
<td>13</td>
<td>101</td>
<td>34</td>
<td>666</td>
</tr>
</tbody>
</table>

   **RADAR TYPE 12 = SURFACE SEARCH TYPE 13 = AIR SEARCH**

   **UMPIRE VERSION OF RADAR DETECTIONS, UNIT 71**
   
<table>
<thead>
<tr>
<th>ID</th>
<th>TYPE</th>
<th>RADAR</th>
<th>CSE</th>
<th>X GRID</th>
<th>Y GRID</th>
<th>SPEED</th>
<th>COMP</th>
<th>CPA</th>
<th>RGE &amp; BRG</th>
</tr>
</thead>
<tbody>
<tr>
<td>333</td>
<td>2333</td>
<td>12</td>
<td>90</td>
<td>513</td>
<td>480</td>
<td>11</td>
<td>1</td>
<td>11</td>
<td>67</td>
</tr>
</tbody>
</table>

2. **Red/Blue Cartesian Plot**

   For each terminal, a plot of all the side's forces in a 1000 by 1000 mile cartesian plot in the positive quadrant is provided. All detections which specified an (X,Y) position
are also plotted on this page. The two sides' units are plotted with different symbols ("+" and "."). This is distributed to the players.

3. **Status of Forces Table**

Each side receives page 3 which lists their forces and the current game time in hours and minutes. An example is provided below:

<table>
<thead>
<tr>
<th>STATUS OF FORCES AS OF 8 HRS. 30 MIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNIT</td>
</tr>
<tr>
<td>ID</td>
</tr>
<tr>
<td>------</td>
</tr>
<tr>
<td>50</td>
</tr>
<tr>
<td>51</td>
</tr>
<tr>
<td>52</td>
</tr>
<tr>
<td>60</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>71</td>
</tr>
</tbody>
</table>

The last four columns are the maximum speed and sensor ranges for the unit in question. Note that the fifth is out of play.

4. **Umpire's Cartesian Plot**

A plot of all the units in the game including those out of play is provided on each terminal's output. The format is the same as that for the players. The copies from both terminals should be identical. Two are provided for the convenience of the Control Group.

5. **Data Base Printout**

File FT02F001 is printed out on each terminal as it appears following the execution of TURN but without the updating accomplished in the CHANGE program. Each terminal's copy should be identical. Both copies are provided for Control Group use.
E. PROGRAM FAILURES

Only two kinds of program failures should occur in CASTAG. Either a data entry error in the interactive phases or a CP/CMS shutdown may cause termination of the program. The critical consideration in restarting CASTAG is that both programs in two different terminals must be kept executing together. Suggestions for dealing with some of the various possibilities are provided in this section.

1. Data Entry Error

Data entries to the EXEC routine cannot normally cause termination as illegal entries cycle the program back to the input statement. Entry of the wrong side, i.e., "R" vice "B" or wrong user ID will prevent any file transfers. To recover, terminate the EXEC and manually make the required transfer. Next turn the programs can be restarted together.

Although only three entries are required for SEARCH, these may have the most damaging effect as some illegal character entries are read as zero. If the TURN program terminates, the CASTAG EXEC may be restarted. "1" and "2" are the only entries accepted for Red/Blue terminal designation. Should both terminals enter the same number, stop execution and restart execution with "CASTAG EXEC". A time of zero will cause no movement or detections and again, the program may be restarted. The worst case is if the programs start with two different times. The original database will be irrevocably changed to reflect movement. Short
of retyping the data base the only solution is to have the terminal with the shorter time execute SEARCH for the difference. The only effect on the game would be one more HFDF segment for the terminal executing twice.

CHANGE is where most errors usually occur due to the large number of entries required. All entries to CHANGE with the exception of the first may be changed again and again until the user terminates the program. The first entry which tells the program if it is Red or Blue will cause an error if the number does not match that entered in SEARCH. If CHANGE does terminate, it can be restarted with "CHANGE". The rest of the EXEC will have to be done manually until it is time to start the next turn. Then CASTAG EXEC can be restarted to get both terminals back together.

2. CP/CMS Failure

When a CP/CMS failure occurs, immediately check to see what condition file FT02F001 (the data base) is in. If both terminals were exactly in step and CP/CMS went down just after the file was rewound on both terminals but before the information was written back on the file, the data base will be lost.

Assuming the data base is intact on one terminal, execute CASTAG from the top utilizing the data base transfer section. The critical consideration is that both terminals always start from identical data bases.
V. FUTURE DEVELOPMENT AND RESULTS

A. IMPROVEMENTS AND EXTENSIONS OF CASTAG

Future work on CASTAG falls into three major categories. First, there is the inclusion of those parts of SEATAG not programmed. Second is improvements to facilitate play and umpire functions of the part of the game now covered. Last, improvements to the program to use less core and improve programming efficiency should be considered.

1. Tactical Improvements

In SEATAG, surface units with two air search radars are given no more capability than a unit with only one. Units with the dual capability should have some kind of enhanced detection capability against air targets.

The time constraint forced the omission of sonobuoy fields for ASW aircraft. These are treated as static detection areas in the game with a different detection methodology. Currently, they could be played manually in conjunction with CASTAG by the Control Group.

To make CASTAG a complete game requires the programming of the SEATAG engagement and damage methodology. In the author's opinion, an effort approximating the work done on this project would be required to accomplish this.

2. Display and Bookkeeping Improvements

The Control Group is required to monitor aircraft and diesel submarine time on station to prevent unrealistic
mission lengths. Time on station or time submerged could easily be kept in one of the unused columns of the DATA array.

To facilitate play of aircraft, the SEATAG aircraft carrier packages would speed play and reduce the number of units devoted to air assets. If the Blue player wants to use the SEATAG CV around the clock defensive package with a strike or cap surge, he will have to operate fifteen plus aircraft. The CV begins to approach the program limit of total units in a carrier strike warfare environment. In conjunction with programming the packages, multiple units where 10 strike aircraft would be one unit of composition 10 would be introduced.

The 1000 by 1000 mile plots should be made variable so the players can enter a midpoint and radius to designate the plot limits. Since most engagements take place at ranges less than 100 miles, a smaller scale output centered around the area of interest would be an improvement.

The player output should include a current range calculation whenever the (X,Y) position is given. As it now stands, the players have to make this determination themselves.

3. **Programming Improvements**

The input screening and interactive sequences have ample room for improvement. Some could be eliminated, particularly two of the three inputs telling the terminal whether it is Red or Blue.
The arrays RADAR, SONAR, ESMDET, INTELL and HFDF are redundant. Their function could have been handled by one array.

There are, no doubt, other inefficiencies the discerning programmer could spot.

B. RESULTS

The SEATAG Documentation suggests lunch breaks and related lectures as a way to utilize the time the two sides will spend waiting for the umpires to finish. This is an example of the principal difficulty with a manual wargame, the time factor.

CASTAG reduces the time to play a turn of SEATAG to less than fifteen minutes. All of the time saved comes from what was essentially idle time for the players while the Control Group plotted the moves. The game can be used as is in the various wargaming courses extant at the Naval Postgraduate School and perhaps elsewhere.
### APPENDIX A
#### ADDITIONAL UNITS

<table>
<thead>
<tr>
<th>Ship Class/Aircraft Name</th>
<th>KRESTA I (4)</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>CHARACTERISTICS</th>
<th>SENSOR FUNCTION</th>
<th>RANGE * (nm)</th>
<th>SENSOR FUNCTION</th>
<th>RANGE * (nm)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Good</td>
<td>Bad</td>
<td>Good</td>
</tr>
<tr>
<td>FLD/MTOW(T):</td>
<td>DON SURE RADAR</td>
<td>20</td>
<td>15</td>
<td>PEEL GRP</td>
</tr>
<tr>
<td>5000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L(ft):</td>
<td>HEAD NET AIR RDR</td>
<td>150</td>
<td>75</td>
<td>MUFF COB</td>
</tr>
<tr>
<td>510</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>W/B(ft):</td>
<td>BIG NET AIR RDR</td>
<td>150</td>
<td>75</td>
<td>BASS TILT</td>
</tr>
<tr>
<td>56</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D/H(ft):</td>
<td>FLINTH NET ?</td>
<td></td>
<td></td>
<td>SCOOP/</td>
</tr>
<tr>
<td>20</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ceiling(ft):</td>
<td>SONAR ASW</td>
<td>10</td>
<td>2</td>
<td>PAIR</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Spd: Max(kt):</td>
<td>SS-N-3 ASW/SSM</td>
<td>2</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>35</td>
<td></td>
<td></td>
<td></td>
<td>30</td>
</tr>
<tr>
<td>Cruise(kt):</td>
<td>SA-N-1 AAW/SAM</td>
<td>2</td>
<td>2</td>
<td>60</td>
</tr>
<tr>
<td></td>
<td>57mm GUN AAW</td>
<td>2</td>
<td>2</td>
<td>1000</td>
</tr>
<tr>
<td>1000</td>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>Patrol(kt):</td>
<td>20mm GUN ASMD</td>
<td>4</td>
<td></td>
<td>2000</td>
</tr>
<tr>
<td>18</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Mission Time(hr):</td>
<td>MBB2500 ASW</td>
<td>2</td>
<td>12</td>
<td>48</td>
</tr>
<tr>
<td>20 day</td>
<td></td>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Endurance at Cruise (nm):</td>
<td>MBB4500 ASW</td>
<td>2</td>
<td>6</td>
<td>24</td>
</tr>
<tr>
<td>5500</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Remarks:</td>
<td>21&quot; TT ASW/ASUW</td>
<td>2</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>100,000 SHP</td>
<td></td>
<td></td>
<td></td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>#170 NM WITH EXTERNAL TARGETING</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**NOTES:** * These notional values are not necessarily precise. They are considered representative and provided for game purposes only.

---

<table>
<thead>
<tr>
<th>Ship Class/Aircraft Name</th>
<th>KYnda CLASS (4)</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>CHARACTERISTICS</th>
<th>SENSOR FUNCTION</th>
<th>RANGE * (nm)</th>
<th>SENSOR FUNCTION</th>
<th>RANGE * (nm)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Good</td>
<td>Bad</td>
<td>Good</td>
</tr>
<tr>
<td>FLD/MTOW(T):</td>
<td>DON SURE RADAR</td>
<td>20</td>
<td>15</td>
<td>PEEL GRP</td>
</tr>
<tr>
<td>5700</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L(ft):</td>
<td>HEAD NET AIR RDR</td>
<td>150</td>
<td>75</td>
<td>MUFF COB</td>
</tr>
<tr>
<td>468</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>W/B(ft):</td>
<td>BIG NET AIR RDR</td>
<td>150</td>
<td>75</td>
<td>BASS TILT</td>
</tr>
<tr>
<td>52</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D/H(ft):</td>
<td>FLINTH NET ?</td>
<td></td>
<td></td>
<td>SCOOP/</td>
</tr>
<tr>
<td>17</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ceiling(ft):</td>
<td>SONAR ASW</td>
<td>10</td>
<td>2</td>
<td>PAIR</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Spd: Max(kt):</td>
<td>SS-N-3 ASW/SSM</td>
<td>2</td>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td>30</td>
<td></td>
<td></td>
<td></td>
<td>30</td>
</tr>
<tr>
<td>Cruise(kt):</td>
<td>SA-N-1 AAW/SAM</td>
<td>1</td>
<td>2</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>76mm GUN AAW</td>
<td>2</td>
<td>2</td>
<td>1000</td>
</tr>
<tr>
<td>15</td>
<td></td>
<td></td>
<td></td>
<td>16</td>
</tr>
<tr>
<td>Patrol(kt):</td>
<td>MBB2500 ASW</td>
<td>1</td>
<td>12</td>
<td>24</td>
</tr>
<tr>
<td>15 day</td>
<td></td>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Mission Time(hr):</td>
<td>21&quot; TT ASW/ASUW</td>
<td>2</td>
<td>3</td>
<td>12</td>
</tr>
<tr>
<td>15 day</td>
<td></td>
<td></td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Endurance at Cruise (nm):</td>
<td>#170 NM WITH EXTERNAL TARGETING</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**NOTES:** * These notional values are not necessarily precise. They are considered representative and provided for game purposes only.

---

132
APPENDIX B

CHANGE OUTPUT

EXECUTION BEINGS...
ENTER 1 FOR RED, 2 FOR BLUE, FORMAT 111.

>1
IT IS 360. Z TIME, YOU HAVE 6 UNITS.
ENTER THE ROW NUMBER OF THE UNIT YOU WISH TO CHANGE THE
STATUS OF, 112, "0" WILL INDICATE NO CHANGES TO ANY UNIT.

>03
CODE UNIT ID= 52
  2 SPEED IS 10 KTS.
  3 ALT/DEPTH= 0*10**3 / 1=SHALLOW,2=DEEP.
  4 COURSE IS 80 T.
  5 SURFACE SEARCH RADAR: 0 1=ON.
  6 AIR SEARCH RADAR: 0 0=OFF.
  7 SONAR: 0
  8 HF COMMUNICATIONS: 0
ENTER THE CODE NUMBER OF THE LINE YOU WANT TO CHANGE,
111. "0" MEANS NO CHANGES TO THIS UNIT.

>5
ENTER THE NEW VALUE FOR CODE NO. 5, 114.

>0001
CODE UNIT ID= 52
  2 SPEED IS 10 KTS.
  3 ALT/DEPTH= 0*10**3 / 1=SHALLOW,2=DEEP.
  4 COURSE IS 80 T.
  5 SURFACE SEARCH RADAR: 1 1=ON.
  6 AIR SEARCH RADAR: 0 0=OFF.
  7 SONAR: 0
  8 HF COMMUNICATIONS: 0
ENTER THE CODE NUMBER OF THE LINE YOU WANT TO CHANGE,
111. "0" MEANS NO CHANGES TO THIS UNIT.

>0
IT IS 360. Z TIME, YOU HAVE 6 UNITS.
ENTER THE ROW NUMBER OF THE UNIT YOU WISH TO CHANGE THE
STATUS OF, 112, "0" WILL INDICATE NO CHANGES TO ANY UNIT.

>0
IF YOU WISH TO BRING AN OUT OF PLAY UNIT
INTO PLAY OR PUT A UNIT OUT OF PLAY (I.E. SINK IT),
ENTER THE ROW NUMBER OF THAT UNIT, 112. "0" MEANS
NO UNITS TO PUT IN OR OUT OF PLAY.

>05

133
CODE  UNIT ID=  0
2  SPEED IS  0 KTS.
3  ALT/DEPTH=  20*10**3 / 1=SHALLOW, 2=DEEP.
4  COURSE IS  90 T.

ENTER THE NEW UNIT ID, "0" PUTS THE UNIT OUT OF PLAY. ANY OTHER VALUE BRINGS THE UNIT INTO PLAY WITH THAT ID NUMBER.

>0077
UNIT 5 IS NOW AT  900 500
ENTER FORMAT I4 FIRST THE NEW X COORDINATE THEN, AFTER THE SECOND APPEARS, THE NEW Y COORDINATE.

> 100
> 100
IF YOU WISH TO BRING AN OUT OF PLAY UNIT INTO PLAY OR PUT A UNIT OUT OF PLAY (I.E. SINK IT), ENTER THE ROW NUMBER OF THAT UNIT, 112. "0" MEANS NO UNITS TO PUT IN OR OUT OF PLAY.

> 3
CODE  UNIT ID= 52
2  SPEED IS  10 KTS.
3  ALT/DEPTH=  0*10**3 / 1=SHALLOW, 2=DEEP.
4  COURSE IS  80 T.

ENTER THE NEW UNIT ID, "0" PUTS THE UNIT OUT OF PLAY. ANY OTHER VALUE BRINGS THE UNIT INTO PLAY WITH THAT ID NUMBER.

>0
IF YOU WISH TO BRING AN OUT OF PLAY UNIT INTO PLAY OR PUT A UNIT OUT OF PLAY (I.E. SINK IT), ENTER THE ROW NUMBER OF THAT UNIT, 112, "0" MEANS NO UNITS TO PUT IN OR OUT OF PLAY.

>0
R; T=0.92/2.63 10.38.02

>
CASTAG EXECUTIVE PROGRAM

&TYPEOUT OFF
&GLOBAL SYSLIB IMSLP SSPLIB
&PRINT THIS IS THE SEATAG EXEC ROUTINE.
&PRINT THIS PROGRAM WILL REQUIRE A DATA BASE
&PRINT WHICH MCCLS ALL THE APPROPRIATE SHIP AND
&PRINT AIRCRAFT CHARACTERISTICS.
&PRINT
-MSG &PRINT ENTER THE USER NUMBER OF THE OTHER PLAYER INCLUD-
&PRINT ING P OR C FOR USER TYPE, MAKING A TOTAL OF 5
&PRINT DIGITS, EG: "1059P".
&READ ARGS
&ID = &1
&ERRCR &CONTINUE
&PRINT IS OTHER PLAYERS ID = &ID ? ENTER Y OR N.
&READ ARGS
&IF &1 NE Y &GOTO -MSG3
&PRINT FILE FTC2F001 IS THE DATA BASE. THIS FILE MUST BE
&PRINT SENT TO THE CONSOLE OF THE OTHER PLAYER IF AC COPY
&PRINT EXISTS IN HIS FILES. IF YOU ARE THE OTHER PLAYER,
&PRINT YOU WILL HAVE TO READ FILE FTC2F001.
-MSG4 &PRINT IF NC TRANSFER OF DATA BASE REQUIRED ENTER A.
&PRINT TC TRANSFER DATA BASE TO OTHER PLAYER ENTER "A".
&PRINT TO READ THE DATA BASE FROM THE OTHER PLAYER ENTER "R".
&READ ARGS
&IF &1 EQ T &GOTO -TRANS
&IF &1 EQ N &GOTO -MSG5
&IF &1 NE R &GOTO -MSG4
-&FILED &CONTINUE
CP QLERY F
&PRINT IF YOU HAVE A FILE SPOOLED WAITING TO BE READ.
&PRINT ENTER "Y", IF NOT ENTER ANY OTHER LETTER EXCEPT N.
&PRINT THE PROGRAM WILL CONTINUE TO RETURN HERE UNTIL THE
&PRINT FILE IS AVAILABLE OR UNTIL Y OR N IS ENTERED. TO
&PRINT GO AHEAD WITHOUT READING THE FILE ENTER N.
&READ ARGS
&IF &1 EQ N &GOTO -MSG5
&IF &1 NE Y &GOTO -FILED
O REAL +
&GOTO -MSG5
-TRANS &CONTINUE
CP XFER C TO &ID
O PUACHCC FILE FTC2F001
CP XFER C OFF
-MSG5 &CONTINUE
CLOSIC_Printer OFF
FILEREF 06_PTR
LOAD SEARCH (CLEAR XEQ ACMAP)
C PRINT FILE FT02F001
CLOSIC_Printer ON
FILEREF 06 CON
LOAD CHANGE (CLEAR XEQ ACMAP)
SUPRINT NOW EACH PLAYER SENDS HIS UPDATED FILE TO THE OTHER
SUPRINT PLAYER, TO INITIATE TRANSFER,
-MSG6 &PRINT ENTER "R" IF YOU ARE RED, "B" IF BLUE.
REAC ARCS
ENDIF &1 EQ R &SIDE = &1
ENDIF &1 EQ B &SIDE = &1
PRINT YCU ENTERED &1. IS THIS CORRECT? ENTER "Y" OR "N".
REAC ARCS
ENDIF &1 EQ N &GOTO -MSG6
ENDIF &SIDE EQ R &GOTO -RTOB
ENDIF &SIDE NE B &GOTO -MSG6
-RTCR &CONTINUE
CP XFER C TO &ID
O PUNCHCC FILE FTC4F001
CP XFER C OFF
&GOTC -MSG7
-RTOB &CONTINUE
CP XFER C TO &ID
O PUNCHCC FILE FTC3F001
CP XFER C OFF
-MSG7 &CONTINUE
CP CLR &F
&PRINT DO YOU HAVE A FILE SPOOLED TO BE READ? ENTER "Y" OR "N".
REAC ARCS
ENDIF &1 NE Y &GOTC -MSG7
&REID *
LIST FILE *
SUPRINT YOU SHOULD HAVE FILE TYPES FT02F001, FT03F001, FTC4FC01
SUPRINT AND FT01F001 ON YOUR DISK AT THIS TIME.
LOAD UPDATE (CLEAR XEQ ACMAP)
ERASE FILE FTC3F01
ERASE FILE FT04FC01
&PRINT TO PLAY ANOTHER TURN, ENTER Y. TO QUIT ENTER N.
REAC ARCS
ENDIF &1 EQ Y &GOTC -MSG5
&EXIT
INTEGER RADAR, SOAR, DISK, BDT, SKI, RDF, FD, DATA, ESPDT, PPDF, SQE, CBR
COMMON PRAX, PRAXD, RPRAX, RPRAXD, PRA, PRAXD, RPRAX, RPRAXD, PRA, PRAXD,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
COMMON PRAA, PRAXDAA, RPRAXAA, RPRAXDAA, PRAA, PRAXDAA, RPRAXAA, RPRAXDAA,
IF (ID .EQ. 1) IFCE = 2
IF (ID .EQ. 2) IFCE = 1

C ZERO OUT ARRAYS
DC 20 I=1,NSHIP
CFA (I,1) = 0.0
CFA (I,2) = 0.0
CICE (I) = C.0
CO 13 J=1,NCOL
RARAR (I,J) = 0
SONAR (I,J) = 0
DATA (I,J,1) = 0
DATA (I,J,2) = 0

10 CONTINUE
CO 14 K = I/4
MINDEX (K + 1) = 0
HDF (I,K) = 0

12 CONTINUE
CO 15 L = I/6
MINDEX (L + 1) = 0
INTEL (I,L) = 0

14 CONTINUE
CO 16 L = I/5
MINDEX (L + 1) = 0
ESHDET (I,L) = 0

16 CONTINUE

C READ DATA FROM DISK 2 INTO DATA ARRAYS
DC 30 I=1,REGION
READ (NOISK,101) (DATA (I,J), J=1,NCOL)

1C1 FCRMAT (1514)

2C CONTINUE
DC 32 I=1,2
READ (NOISK,101) (DATA (I,J), J=1,NCOL)

32 CONTINUE

C IF THIS IS TURN 1, CALL NUMBER. IF NOT READ TEISK
IF (ZTIME .LE. EPS .OR. TIME .LE. EPS) CALL NUMBER

1C5 FCRMAT (210F4.3)

33 REMIND DISK
C IF TIME IF OVER 24 HRS.
ZTIME = ZTIME + TIME
IF (ZTIME .GT. 1440.0) ZTIME = ZTIME - 1440.

C WRITE UPDATED INFO BACK ON MASTER FILE AND CREATE FTC3 OR FTC4 RESPECTIVELY. WRITE INFO CO FTC3 ALSO.
C REMIND DISK WITH "DATA BASE ON IT FIRST"
WRITE (IDISK, 109) (PRMAX (I,J,K), I=1,NMAX), J=1,NMAX), K = 1,5)
REINC NDISK
WRITE (NCISK,106) ZTIME,NRED,NBLUE
DC 35 I = 1,NRED
WRITE (NCISK,101) (DATA(I,J,1), J=1,NCOL)
CONTINUE
DC 37 I = 1,NBLUE
WRITE (NCISK,101) (DATA(I,J,2), J=1,NCOL)
CONTINUE
WRITE (REISK,110) ZTIME,NRED
IF (ID = 60, 4) GO TO 40
CONTINUE
WRITE (RDISK,101) (DATA(I,J,1), J = 1,NCOL)
CONTINUE
WRITE (BDISK,110) ZTIME,NBLUE
IF (ID = 60, 1) GO TO 97
CONTINUE
WRITE (BCISK,101) (DATA(I,J,2), J=1,NCOL)
CONTINUE
IF (TIME = GT, EPS) CALL DETECT (TIME)
R(1) = 1000.
R(2) = 0.0
R(3) = 1000.0
R(4) = 0.0
KTIME = ZTIME/60.
IMIN = 60. * ((ZTIME/60.) - FLOAT(KTIME))
Pt = MPINDEX(10)
NLINIM = NRON(ID)
WRITE(16,55)
CONTINUE
NCALL = 1
IF (M = EQ. 0) NCALL = 1
IF (M = EQ. 0) GO TO 56
CALL UTPLCT(XPLCT,YPLCT,M,R,1,1)
CONTINUE
56 DC 57 I = 1,NUNIT
XPLCT(I) = DATA(I,5,ID)
YPLCT(I) = DATA(I,6,ID)
CONTINUE
CALL UTPLCT(XPLCT,YPLCT,NUNIT,R,1,NCALL)
WRITE(16,58) KTIME,MIN
CONTINUE
58 FFORMAT(ID,1) STATUS CF FORCES AS CF ,I3, FRS ,I2, MIN ,/ ,
C UNIT SPEED DEPTH COURSE X Y TYPE MAX MAX MAX MAX
C ID KTS /HT DEG T GRID GRID SPE $S A/S SNR ,/}
DC 61 I = 1,NUNIT
WRITE (6,61) (DATA(I,J,ID),J=1,11)
CONTINUE
WRITE (6,62)
CONTINUE
WRITE (6,62)
62 FCRMAT(11H)
  CC 63 I =1, NRED
  PLOT(I) = DATA(1, 5, 1)
  PLOT(I) = DATA(1, 6, 1)
63 CONTINUE
  CALL UPTLOT(XPLOT, YPLOT, NRED, R, 1, 1)
CC 65 I = 1, NBLUE
  PLOT(I) = DATA(1, 5, 2)
  PLOT(I) = DATA(1, 6, 2)
65 CONTINUE
  CALL UPTLOT(XPLOT, YPLOT, NBLUE, R, 1, 3)
995 STCP
996 EAC
  SUBROUTINE MOVE(TIME)
  INTEGER RADAR, SONAR, RDISK, BDISK, HDISK, DATA, ESMEET
  DIMENSION PRMAX(50, 5C, 5), CICE(5C), NROW(2), CPA(5C, 2),
      RADAR(50, 2C), SONAR(50, 20), DATA(50, 20, 2), PLOT(50), YPLOT(50),
      FDF(50, 4), INTEL(50, 6), ESMEDET(100, 5), MINEC(10), R(4)
  COMMON PRMAX, DATA, RADAR, SONAR, FDF, INTEL, NROW, CPA,
      RRED, NBLUE, NCOL, ID, ZTIME, EPS, ISEE1, ISEE2, ISEE3,
      CICE, NWRITE, NMAX, ESMEDET, IFOE, CTIME, MINEC, R, XPLCT, YPLOT
  TIME = TIME + 0.0
  RAD = 3.14159/180.0
  DC 10 K = 1.2
  DC 5 1 = 1, NBLUE
  D1ST = FLOAT(DATA(I, 2, K) * THCUF
  THETA = FLOAT((DATA(I, 4, K) * RAD
  DATA(I, 5, K) = DATA(I, 5, K) + (3 * D1ST * SIN(THETA))
  DATA(I, 6, K) = DATA(I, 6, K) + (3 * D1ST * C0S(THETA))
  IF (DATA(I, 5, K) < 0.0) DATA(I, 5, K) = 0
  IF (DATA(I, 6, K) < 0.0) DATA(I, 6, K) = 0
5 CONTINUE
10 CONTINUE
  RETURN
  END
  SUBROUTINE NUMBER
  INTEGER RADAR, SONAR, RDISK, BDISK, HDISK, DATA, ESMEET
  DIMENSION PRMAX(50, 5C, 5), CICE(5C), NROW(2), CPA(50, 2),
      RADAR(50, 2C), SONAR(50, 20), DATA(50, 20, 2), PLOT(50), YPLOT(50),
      FDF(50, 4), INTEL(50, 6), ESMEDET(100, 5), MINEC(10), R(4)
  COMMON PRMAX, DATA, RADAR, SONAR, HDISK, INTEL, NROW, CPA,
      RRED, NBLUE, NCOL, ID, ZTIME, EPS, ISEE1, ISEE2, ISEE3,
      CICE, NWRITE, NMAX, ESMEDET, IFOE, CTIME, MINEC, R, XPLCT, YPLOT
  DE 30 K = 1.5
  DE 20 J = 1, NMAX
  CALL RANDOP (JSEE1, DICE, 50)
  DE 10 I = 1, NMAX
  PRMAX(I, J, K) = 0.0
IF (DICE(I) .GE. .9) GO TO 1C
FRMAX (I,J,K) = .9 * SQRT(1.0 - (CICE(I)**2/1.01))
1C CONTINUE
2C CONTINUE
3C CONTINUE
RETURN
END
SUBROUTINE DETECT(TIME)
INTEGER RADAR, SONAR, RDISSK, RDISSK, FDF, DATA, ESMDET
DIMENSION FRMAX (50,50,50), DICE(50), NROW(2), CPA(100,20),
C RADAR(50,20), SONAR(50,20), DATA(50,20,21), XPLCT(50), YPLCT(50),
C FDF(50,4), INTEL(50,6), ESMDET(100,5), MINDEX(10), R(4)
C CMAX(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I), CPA(I)
C CHECK PLATFORM VS PLATFORM DETECTIONS

23 LIGHT = 0
IF (ZTIME .GE. 360. .AND. ZTIME .LT. 1080.) LIGHT = 1
DC = SC = 1
UNIT = NRWO(I,1)
IF (DATA(I,1,1) .EQ. 0) GO TO 5C
ITYPE = DATA(I,7,1) - (1000 * IT)
C IS UNIT A SUBMARINE?
IF (ITYPE .GE. 200 .OR. ITYPE .LT. 100) GO TO 40
CALL NLPCPA(I,TIME)
C IF SONAR PASSIVE AND SPEED LE 22 KNOTS CALL PASSIVE
IF (DATA(I,2,1) .GE. 23) GO TO 25
IF (DATA(I,1,4,1) .EQ. 0) CALL FSONAR(I,1)
C IS SUB DEEP?
IF (DATA(I,3,1) .EQ. 2) GO TO 2C
CALL ESM(I)
C IF SURFACE SEARCH RADAR IS ON & SLG NOT DEEP...
IF (DATA(I,12,1) .EQ. 1) CALL DETRAD(I)
C IS SUB SURFACED & IT IS DAY CHECK VISUAL SIGHTINGS
IF (DATA(I,3,1) .EQ. 0 .AND. LIGHT .EQ. 1) CALL VISUAL(I,ITYPE)
C IF SONAR ACTIVE AND SPEED LE 15 KTS.
30 IF (DATA(I,1,4,1) .EQ. 1 .AND. DATA(I,2,1) .LE. 15) CALL ASONAR(I)
GO TO 50
C IS UNIT A SURFACE SHIP?
4C IF (ITYPE .LT. 300) GO TO 50
CALL NLPCPA(I,TIME)
CALL ESM(I)
IF (LIGHT .EQ. 1) CALL VISUAL(I,ITYPE)
C IS EITHER AIR OR SURFACE SEARCH RADAR ON?
IF (DATA(I,12,1) .EQ. 1 .OR. DATA(I,13,1) .EQ. 1) CALL DETRAD(I)
C IF SONAR PASSIVE AND SPEED LESS THAN 17 CALL PASSIVE SONAR
IF (DATA(I,14,1) .EQ. 0 .AND. DATA(I,2,1) .LE. 17) CALL PSCNAR(I,2)
C IF SONAR ACTIVE AND SPEED LT 15 KNOTS CALL ACTIVE SCNAR
IF (DATA(I,1,4,1) .EQ. 1 .AND. DATA(I,2,1) .LE. 15) CALL ASCNAR(I)
GO TO 90
5C ITYPE = ITYPE - 2C0
C UNIT IS AN AIRCRAFT. IF IT IS DAY, ALL AIRCRAFT CALL VISUAL
CALL NLPCPA(I,TIME)
IF (LIGHT .EQ. 1) CALL VISUAL(I,ITYPE)
C IF AIRCRAFT IS FIGHTER/ATTACK GO TC 90
IF (ITYPE .LT. 70) GO TO 90
C CALL OTHER TYPES MAKE ESM & RADAR CTECTIONS
CALL ESM(I)
CALL DETRAD(I)
C CALL SH-3 & HORMONE MAY USE DIPPING SCNAR IF SPEED LT 10 & ALT LT 1
IF (ITYPE .EQ. 93 .AND. DATA(I,2,1) .LE. 1C .AND.
C DATA(I,3,1) .EQ. 1) CALL ASCNAR(I)
5C CONTINUE
55 RETURN
FAC
SUBROUTINE SPOOK
INTEGER RADAR, SCNAR, RDISK, BDISK, IFOE, DATA, ESMDET
DIMENSION PRMAX (50, 5C), DIC(5C), NRDLW(2), CPA(5C, 2),
C RADAR(50, 20), SCNAR(50, 20), DATA(50, 20, 2), XPL(50), YPL(50),
C FDF(50, 4), IFOE(50, 4), ESMDET(100, 5), DEX(100, 5),
C COMMON PRMAX, DATA, RADAR, SCNAR, HCF, INTEL, NRCLW, CPA,
C RED, NBLUE, NCOL, I0D, I0TIME, EPS, ISEED1, ISEED2, ISEED3,
C XICE, NWRITE, NMAX, ESMDET, IFOE, ETIHE, MINFES, R, XPLCT, YFLCT
C THIS SUBROUTINE DETERMINES THE 0801, 0600, 12CC
C & 1800 INTELLIGENCE INFORMATICA.
NINIT = NROW(IFOE)
N = 0
CALL RANDOM (ISEED3, DICE, NUNIT)
C 111 = 1, NUNIT
         IF (DICE(1) .LT. .59) GO TO 11
         IF (DATA (1, 1, IFOE) .LE. 0) GO TO 11
         MCON = 0
         I0D = DATA(1, 7, IFOE) - (1000 * IFOE)
         CC 9 J=12, 15
         MCON + DATA (I, J, IFOE)
5 CONTINUE
         IF (DICE(1) .LT. .79 AND. MCON .LE. 0) GO TO 11
         C IS UNIT SUBJECT UNDER 15 KTS?
         IF (I0D .LT. 200) ANG, DATA(1, 2, IFOE) .LT. 15) GO TO 11
         C IS UNIT AIRCRAFT IN EMCON?
         IF (I0D .LT. 300) ANG, MCON .LE. C ) GO TO 11
         C AN INTELLIGENCE/SURVEILLANCE DETECTION HAS CCCUREC
         A=N+1
         IF (DICE (1) .GT. .8) GO TO 10
         C ADDITIONAL INTE
         IC2=FICAT(CATA(I, 7, IFOE)/100)
         INTEL(N, 5) = DATA(I, 7, IFOE) - ID2*100
         INTEL(N, 6) = ((DATA(I, 4, IFOE) +51100)/100
9 C BASE INFORMATION
9 INTEL(N, 1) = DATA(I, 1, IFOE),
9 INTEL(N, 2) = FICAT(DATA(I, 7, IFOE)/100)
9 INTEL(N, 3) = DATA(I, 5, IFOE)
9 INTEL(N, 4) = DATA(I, 6, IFOE)
9 XPLT(I0D + INDEX(I0L)) = DATA(I, 5, IFOE)
9 YPLT(I0D + INDEX(I0L)) = DATA(I, 6, IFOE)
10 CONTINUE
9 MINDEX(10) = MINDEX(10) + N
9 IF (N .EQ. C) GO TO 99
9 WRITE (NWRITE, 12)
12 FORMAT (10X, 'UNIT TYPE ', 3X, 'X COORD YCOORD', ' PLATFORM COURSE')
9 WRITE (NWRITE, 13) ((INTEL(I, J), J=2,6), I=1,N)
D002850
D002860
D002870
D002880
D002890
D002900
D002910
D002920
D002930
D002940
D002950
D002960
D002970
D002980
D002990
D003000
D003010
D003020
D003030
D003040
D003050
D003060
D003070
D003080
D003090
D003100
D003110
D003120
D003130
D003140
D003150
D003160
D003170
D003180
D003190
D003200
D003210
D003220
D003230
D003240
D003250
D003260
D003270
D003280
D003290
D003300
D003310
D003320
12 FORMAT (', 9X, 519, /)  
55 RETURN  
END  
SUBROUTINE ESM (I)  
INTEGER RADAR, SONAR, RDISK, BDISK, FOF, DATA, ESMDET  
DIMENSION PRMAX(50, 5C, 5) DICE(5C, 5C), NRT(2), CPA(50, 2),  
C RADAR(50, 2C), SONAR(50, 2C), DATA(50, 20, 2), YPLOT(50), YPLOT(50),  
C FOF(50, 4), INTEL(50, 6), ESMDET(100, 5), MINE(10), R(4)  
COMMON PRMAX, DATA, RADAR, SONAR, FOF, INTEL, ARCH, CPA,  
C RAR, NUSE, NCOL, ID, 2TIME, EPS, MEO, ISEE, R, ISEE, C  
C E I C, RWRITE, NM, ESMDET, IFOE, 2TIME, MINE, R, XPLCEL, YPLOT, YPLOT  
C THIS SUBROUTINE IS CALLED FROM DETECT. IT CALCULATES ALL POSSIBLE  
C ESM DETECTIONS BY THE ITH RED/BLUE UNIT. IT MUST HAVE THE  
C RESULTS OF THE CPA SUBROUTINE AND THE NUMBER SLOTTING.  
C DETECTIONS ARE PUT INTO ARRAY ESMDET  
C NLNIT = NROW(IFOE)  
M = 0  
C IF RADAR IS ON AND UNIT WILL BE WITHIN 2 * RMAX * CET  
C A DETECTION WILL OCCUR.*  
DC 40 N = 1, NUNIT  
C IF (DATA(N, 1, IFOE), 1E-0) GO TO 40  
ATYPE = DATA(N, 7, IFOE) - (1000 * IFOE)  
IF (DATA(N, 12, IFOE), EQ, 0) GO TO 35  
RANGE = 2 * DATA(N, 9, IFOE) * PRMAX(1, N, 3)  
IF (CPAIN(1), GT, RANGE) GO TO 35  
M = M + 1  
ESMDET(M, 1) = 12  
ESMDET(M, 2) = CPA(N, 2)  
ESMDET(M, 4) = DATA(N, 1, IFOE)  
ESMDET(M, 5) = CPA(N, 1)  
ESMDET(M, 3) = NTYPE/10  
C IF AIR SEARCH RADAR ON, SEE IF ANY DETECTIONS  
35 IF (DATA(N, 13, IFOE), 1E-0) GO TO 40  
IF (DATA(N, 10, IFOE) * PRMAX(1, N, 3))  
IF (CPAIN(1), GT, RANGE) GO TO 40  
M = M + 1  
ESMDET(M, 1) = 13  
ESMDET(M, 2) = CPA(N, 2)  
ESMDET(M, 4) = DATA(N, 1, IFOE)  
ESMDET(M, 3) = NTYPE/10  
ESMDET(M, 5) = CPA(N, 1)  
40 CONTINUE  
C IF (M = N) GO TO 60  
WRITE (AWRITE, 50) DATA(N, 1, ID)  
50 FORMAT (', *, ESM INTERCEPTS BY UNIT*, IS, 7X, *, RADAR*, 2X,  
C*, BEARING*, 6X, PLATFORM*)  
DC 56 JJ = 1, M  
WRITE (AWRITE, 55) (ESMDET(IJ, JJ), JJ = 1, 3)
56 FORMAT(*,'(I1)')  DET03810
52 CONTINUE   DET03820
WRITE(NWRITE,57)   DET03830
57 FORMAT(*=' RACAR TYPE 12 = SURFACE SEARCH TYPE 13 = AIR SEARCH',/)
WRITE(6,58) DATA(1,1),1,D)  DET03840
58 FORMAT(*=U8,UMPIRE VERSION ID = ',4/)
C EX = RADAR BEARING PLATFORM IC NO. CPA RANGE,/)  DET03850
C DC 61 II = 1,M  DET03860
WRITE(6,59) (ESMS(11, JJ), JJ = 1,5)  DET03870
56 FORMAT(*=' (I1)')  DET03880
61 CONTINUE   DET03890
WRITE(6,57)   DET03890
6C RETURN   DET03900
END
END
SUBROUTINE CETRAD (I)
INTEGER RADAR, SONAR, RDISK, BDISK, FD, DATA, ESMS(1)
DIMENSION PMAX(50,50,5), DICE(50), NROW(2), CPA(50,1),
C RADAR(50,2C), SONAR(50,20), DATA(50,50,2), XPL(50), YPL(50),
C FD(50,4), INTEL(50,6), ESMS(100,5), MINDEX(10), R(4)
COMMON PMAX, DATA, RADAR, SONAR, FD, INTEL, CPA,
C ARE(,B, NCL, ID, ZTINE, EPS, ISEED1, ISEED2, ISEED3),
C XICE, WRITE, NMAX, ESMS, IOFE, CTIME, MINDEX, R, XPL, YPL,
C THIS ROUTINE IS CALLED FROM DETECT. IT CALCULATES ALL
C POSSIBLE RADAR DETECTIONS OF THE ITH RED/BLUE UNIT AGAINST
C EACH BLUE/RED UNIT. NEEDS INPUTS FROM SEDICT.
C NUMBER AND CPA. AIR SEARCH RADARS ONLY DETECT AIR TARGETS AND
C AND SURFACE SEARCH RADARS ONLY DETECT SURFACE TARGETS.
C DETECTIONS ARE STORED IN ARRAY RADAR.
NUNIT = NROW(IOFE)
M = 0   DET03950
DC 50 N = 1,NUNIT
IF (DATA(N,1,IFOE),NE.0) GO TO 50
ITGT = DATA(N,7,IFOE) - (1000 + IOFE)   DET04120
C IF S/S RADAR OFF, SKIP TO NEXT RADAR
C IF DATA(N,12,TC), EQ. 0) GTC 25   DET04130
C IF DEPTH/HIT NOT EQUAL TO 0, GTG IS NOT ON SURFACE
C IF (DATA(N,3,IFOE),NE.0) GO TO 25
C RANGE = PMAX(I,N,1) * DATA(I,9,1ID)
C IF (ITGT + I, 200) RANGE = RANGE * .5
C IF (CPA(N,1) GT. RANGE) GTC 25
M = M+1   DET04170
FADAR(M+1) = DATA(N,1,IFOE)   DET04210
FADAR(M+2) = DATA(N,7,IFOE)   DET04220
FADAR(M+3) = 12   DET04230
FADAR(M+4) = DATA(N,4,IFOE)   DET04240
FADAR(M+5) = DATA(N,5,IFOE)   DET04250
FADAR(M+6) = DATA(N,6,IFOE)   DET04260
FADAR(M+7) = DATA(N,2,IFOE)   DET04270
FADAR(W,8) = 1
FADAR(W,9) = CPA(M,1)
FADAR(W,10) = CPA(M,2)
XPLT(W + INDEX(10)) = DATA(N,5,IFOE)
YPLT(W + INDEX(10)) = DATA(N,6,IFOE)
GO TO 50
25 IF AIR SEARCH RACAR CN AND TGT IS AN AIRCRAFT.
   IF (DATA(I,13,1D) = LE 0) GO TO 50
   IF (ITGT < LT. 200 OR ITGT > GE. 300) GO TO 50
   ARANGE = PRMAX(I1,4,N,2) * DATA(I,10,10)
   IF (CPA(N,1) = 1) ARANGE = GO TO 50
   M = M+1
   GO 30 J = 1,7
   RADAR(M,J) = DATA(N,J,IFOE)
30 CONTINUE
   XPLT(W + INDEX(10)) = DATA(N,5,IFOE)
   YPLT(W + INDEX(10)) = DATA(N,6,IFOE)
   RADAR(M,3) = 13
   RADAR(M,2) = DATA(N,7,IFOE)
   RADAR(M,7) = DATA(N,2,IFOE)
   RADAR(M,8) = 1
   RADAR(M,9) = CPA(N,1)
   RADAR(M,10) = CPA(N,2)
5C CONTINUE
   M = INDEX(10) + M
   M = INDEX(10) + M
   IF (M = EQ. C) GO TO 99
   WRITE(NWRITE,60) DATA(I,14,1D)
   WRITE(NWRITE,70) (RADAR(I1,J), JJ=3,8)
60 FCMAT(I,,14/) /
   RACAR DETECTIONS BY UNIT \\
   RACAR TYPE', '2X', 'COURSE', '3X', 'X COORD YCCRD', '2X', 'SPEED', \\
   '3X', 'COMP.' \\
   DC 75 II = 1, M
   WRITE (NWRITE,70) (RADAR(I1,J), JJ=3,8)
7C FCMAT (I,,8X,618)
75 CONTINUE
76 FCMAT (I,,14/) /
   UMPIRE VERSION OF RACAR DETECTIONS, UNIT \\
   RACAR TYPE', '2X', 'COURSE', '3X', 'X GRID Y GRID', \\
   '3X', 'COMP.', 'CPA RGE & RGE', 'EAC'.
   DC 78 II = 1, M
   WRITE (NWRITE,70) (RADAR(I1,J), JJ = 1,10)
77 FCMAT (I,,8I6,3X,216)
78 CONTINUE
55 RETURN
56 FNC
SLEROUTINE ASONAR(I)
INTEGER RACAR, ADDAR, RDIS, BDIS, FDFD, DATA, ESMDET
DIMENSION PRMAX (5C,5C,5), DICE (5C), NRNW (2), CPA (50,2)
C RADAR(50,20), SONAR(50,20), DATA(50,20,2), XPLT(50), YPLCT(50), DETC4770
C IFOE(50,4), INTEL(50,6), ESMOT(100,5), MENCE(10), R(4)
C COMMON PRMAX, DATA, RDCAR, FCE, INTEL, APC, CPA
C AREA, NPLS, NCEOT, ISCE1, ISCE2, ISCE3,
C CICE, NWRITE, NMAX, ESMOT, IFOE, CTIME, MENCE, R, XPLT, YPLCT
C THIS SUBROUTINE CALCULATES ALL POSSIBLE DETECTIONS
C FROM ACTIVE SONAR BY THE ITH UNIT. SONAR IS
C CHECKED TO BE ACTIVE AND SPEED LESS THAN 15 KNOTS
C PRIOR TO CALLING THE ROUTINE. BOTH RED AND BLUE SUBS MAY BE
C DETECTED REGARDLESS OF WHETHER I IS RED OR BLUE. THE
C DATA FROM NUMBER AND NLPAPA MUST BE AVAILABLE. THIS
C SUBROUTINE IS CALLED FROM DETECT AND CALLS NLPAPA.
C DETECTIONS ARE STORED IN ARRAY SONAR.
C NUNIT = NROW(IFOE)
CSAVE = IFOE
KUNIT = NROW(ID)
M = 0
CC 60 K = 1,2
DL 5G N = 1,NUNIT
IF (DATA(1,1,IFOE) - EQ. 0) GO TO 5C
IF (I .EQ. N .AND. K .EQ. 2) GO TO 50
C IS THE TARGET A SUBMARINE NOT ON THE SURFACE?
ITG = DATA(7,1,IFOE) - (1000 * IFOE)
IF (DATA(N,3,IFOE) .EQ. 0 OR. ITG .GE. 2CC) GO TO 1C
RANGE = PRMAX(I,N,3 + K1) + DATA(11,1,IFOE)
IF (CPI(1,1,IFOE) .GT. RANGE) GO TO 5C
C A DETECTION HAS OCCURRED
+ = +1
CC 40 IA = IA + 6
SONAR(M,IA) = DATA(1,IA,IFOE)
40 CONTINUE
SCAR(1,2) = DATA(N,7,IFOE)
SCAR(1,0) = CPA(1,0)
SCAR(1,1) = CPA(1,1)
SCAR(1,2) = CPA(1,2)
50 CONTINUE
IF (K .EQ. 2) GO TO 60
NUNIT = KUNIT
IFCE = ID
TIME = CTIME
CALL NLPAPA(I,TIME)
60 CONTINUE
IFCE = ISAVE
IF (M .EQ. 0) GO TO 99
C PRINT OUT SONAR
WRITE(NWRITE,ST) DATA(1,1,IFOE)
7C FORMAT(20,15X,'ACTIVE SONAR DETECTIONS BY UNIT: ',4T,17X,
C ,+ DEPTH,4X,'COURSE',4X,'X CCCRC',1X,'X V CCCRD',2X,'SPICE',1X)
C DETE5240
CC 85 II = 1. M
80 WRITE (NWRITE, 80) (SONAR(I, JJ), JJ = 3, 7)
80 FORMAT (' ', 1, 14X, '217, 6X, 317) DET05250
85 CONTINUE DET05260
86 WRITE(6, 86) DATA(I, 1, ID) DET05270
86 FORMAT(10/6, 1A5, 1/ 1A4, 1A3, 1/ 1A4) DET05300
C 2X*, CPA DIST & BRG*) DET05310
DC 88 II = 1.M DET05320
86 WRITE(6, 87) (SONAR(I, JJ), JJ = 1, 9)
87 FORMAT(10/6, 1A4, 2X, 216) DET05340
XPLG(MINDEX(10) + I I) = SONAR(I, 5) DET05350
YFLG(MINDEX(10) + II) = SONAR(I, 6) DET05360
88 CONTINUE DET05370
55 MINDEX(I0) = MINDEX(10) + M RETURN
ENC
SUBROUTINE ALPCA(I, TIME)
INTEGER RADAR, SONAR, ROI, BDISK, FDF, DATA, ESMDET
DIMENSON PRMAX (50, 5, 5), DICE(5C), NROW(2), CPA(50, 2),
C RADAR(50, 2C), SONAR(50, 20), DATA(50, 20, 2), XFLG(5C), YFLG(5C),
C -FDF(50, 4), INTEL(50, 6), ESMDET(100, 5), MIND EX(10), R(4)
CCMOM PRMAX, DATA, RADAR, SONAR, FDF, INTEL, ARC, CPA,
C RED, NMAX, NCOL, IDI, TIME, EPS, ISR, NMAX, NMAX, ISER, ISEED3,
C DICE, NWRITE, MAX, ESMDET, IFOE, CTIME, MIND, R, XLQ, YFLG
CC FOR THE IF+ RED/BLUE UNIT DETERMINE THE RANGE AND BEARING
CC OF THE CPA FOR EACH OPPCISNG UNIT. N SUBSCRIPTS THE
CC OPPOSING UNITS AND CPA(AB) STORES THE INFO WHERE R = RANGE
C A8 = BEARING IN DEGREES TRUE FROM THE IF+ UNIT.
RIF = 3.14159 / 180.0 DET05400
T = TIME/60. DET05410
NWRITE = NROW(IFOE)
A = FLOAT(DATA(I, 5, ID)) DET05420
B = FLOAT(DATA(I, 6, ID)) DET05430
D2 = T * FLOAT(DATA(I, 2, ID)) DET05440
ETA = RAD * FLOAT(DATA(I, 4, ID)) DET05450
XIST = D2 * SIN(ETA) DET05460
YIST = D2 * COS(ETA) DET05470
IF 50 N = 1, UNIT
CC CALCULATE XI AND X2, THE RELATIVE POSITION OF THE NTH UNIT
C 27 TIME - TIME.
C T-ETA = FLOAT(DATA(N, 4, IFOE)) * FAC
C J = T * (FLOAT(DATA(N, 2, IFOE))
C X2 = DATA(N, 5, IFOE)
Y2 = DATA(N, 6, IFOE)
X1 = X2 + XIST - C1 * SIN(T-ETA)
Y1 = Y2 + YIST - D1 * COS(T-ETA) DET05490
C CHECK IF SLOPE OF DRF LINE IS INFINITE DET05500
IF (ABS(X2-X1) .LE. EPS) GO TO 16
CALCULATE SLOPE CP DRM
D) = (Y2 - Y1)/(X2-X1)
GO TO 18
16 IF (ABS(Y2-Y1) .LT. EPS) GO TO 3C
X) = X2
Y) = B
IF IB .GT. Y1 .AND. B .GT. Y2 .AND. Y1 .LT. Y2) GC TC 35
IF IB .GT. Y1 .AND. B .GT. Y2 .AND. Y1 .LT. Y2) GO TO 3C
IF IB .LT. Y1 .AND. B .LT. Y2 .AND. Y1 .LT. Y2) GO TO 3C
GO TO 40
18 X) = (A + B*CX + X1*DX*DX - DX*Y1)/(1.0 + DX **2)
Y) = X) * DX + Y1
C SEE IF THE CALCULATED CPA (X,Y) OCCURS BETWEEN (X1,Y1)
C AND (X2,Y2). IF NOT SET X,Y TO THE CLOSEST ENDPOINT
C IF (X = LT. X1 .AND. X = LT. X2) CPA(X1,Y1) GO TO 3C
C IF (X = LT. X1 .AND. X = LT. X2) CPA(X2,Y2) GO TO 3C
C IF (X = LT. X1 .AND. X = LT. X2) CPA(X1,Y2) GO TO 3C
C IF (X = LT. X1 .AND. X = LT. X2) CPA(X2,Y1) GO TO 3C
GO TO 4C
3C X) = X1
Y) = Y1
GO TO 40
35 X) = X1
C CALCULATE DISTANCE TO (X,Y) FROM (A,B)
C CPA(N,1) = SORT (((X-A)**2 + (Y - B)**2)
C IF CPA IS VERY "SMALL", SET IT TO EPS
C IF (CPA(N,1) .LT. EPS) CPA(N,1) = EPS
TAG = ARSIN (((X-A)/CPA(N,1))
C IF (X .LT. A) .AND. (Y .GE. B) TAU = TAG + 2.0 * 3.14159
C IF (X .LT. A) .AND. (Y .LT. B) TAU = 3.14159 - TAU
C IF (X .GT. A) .AND. (Y .LT. B) TAU = 3.14159 - TAU
C IF (X .GT. A) .AND. (Y .GE. B) TAU = TAU/RAD
5C CONTINUE
RETURN
STOP
SUBROUTINE VISUAL(I,ITYPE)
INTEGER RDRAD, SOCAR, RDISK, BDISK, FD, DATA, ESMDET
DIMENSION PRMAX(50,50), ORIG(50), NRAD(4), CPA(50,2),
C RADAR(50,2), SNMARK[1024], DATA(50,20,2), XPLCT(50), YPLCT(50),
C FFCF(50,4), INTEL(50,6), ESMDET(100,5), MINE(10,4),
C CCNM, PRMAX, DATA, RDRAD, SOCAR, FD, DATA, ESMDET, IFD, CTIME, MINDEX, R, XPLCT, YPLCT
C THIS SUBROUTINE CALLED FROM DETECT. CPA'S FRCW
DETO573C
DETO574C
DETO575C
DETO576C
DETO577C
DETO578C
DETO579C
DETO580C
DETO581C
DETO582C
DETO583C
DETO584C
DETO585C
DETO586C
DETO587C
DETO588C
DETO589C
DETO590C
DETO591C
DETO592C
DETO593C
DETO594C
DETO595C
DETO596C
DETO597C
DETO598C
DETO599C
DETO600C
DETO601C
DETO602C
DETO603C
DETO604C
DETO605C
DETO606C
DETO607C
DETO608C
DETO609C
DETO610C
DETO611C
DETO612C
DETO613C
DETO614C
DETO615C
DETO616C
DETO617C
DETO618C
DETO619C
DETO620C
C NLECPA ARE USED TO CHECK FOR VISUAL SIGHTINGS
C FROM O600 TO 1800 FRS GAME TIME, SHIPS AND SURFACED
C SLEMARINES DETECT ALL UNITS WITHIN 10 MILES.
C AIRCRAFT DETECT ALL UNITS WITHIN 2C NAUT MILES.
C SUBMERGED SUBMARINES ARE NOT DETECTED. ALL
C DETECTIONS ARE STORED IN ARRAY RACAR, COL IC-20.
C RANGE = 10.C
IF (ITYPE .LT. 100) RANGE = 20.C
M = 0
UNIT = NROW(IFOE)
DC 50 N = 1, UNIT
IF (DATA(N,1),IFOE) = EQ. 0 GO TO 50
RTYPE =DATA(N,7),IFOE) -(1000*IFCE)
IF(CDATA(N,3,IFOE) GT. 0 AND NTYPE .LT. 20) GO TO 5C
IF (CPA(N,1) GT. RANGE) GC TO 5G
C DETECTION HAS OCCURRED
M = M+1
RADAR(N,19) = DATA(N,1,IFOE)
RADAR(N,21) = DATA(N,7,IFOE)
RADAR(N,12) = DATA(N,3,IFOE)
RADAR(N,13) = DATA(N,4,IFOE)
RADAR(N,14) = DATA(N,5,IFOE)
RADAR(N,15) = DATA(N,6,IFOE)
RADAR(N,17) = I
RADAR(N,16) = CPA(N,2)
RADAR(N,18) = CPA(N,1)
YPLOT(V + INDEX(10)) = DATA(N,5,IFOE)
YPLOT(V + INDEX(10)) = DATA(N,6,IFOE)
5C CONTINUE
MINDEX(10) = M + MINDEX(10)
IF (M .GT. 29) GC TO 99
WRITE(NWRITE,60) DATA(I,J,IP)
6C FCRMAT (/ ,/) VISUAL DETECTIONS BY UNIT ",I4,/,9X,
C TYPE*,1X,ALT.*,2X,COURSE*,1X,*X COORD *CCRD*,2X,*BEARING*,
C COMP*
570 II = M
WRITE (NWRITE,65) (RADAR(I,JJ),JJ = 11,17)
65 FCRMAT (",1X,51X,3X,2I6)
7C CONTINUE
7C FCRMAT (/ ,/) UNIPRE VERSION OF VISUAL SIGHTINGS, UNIT ",,I4,/,,
C TYPE*,4X,ALT.*,IX,COURSE*,1X,X COORD YCCRD*," BRC *
C,COMP CFA RNG*," ID*
DC 75 IT = 1,M
WRITE (6,71) (RADAR(I,JJ), JJ = 11,19)
71 FCRMAT (",3I6,4X,6I6)
7C CONTINUE
5C FEILRN
END
SUBROUTINE FSONAR(I,NPLAT)
INTEGER RADAR, SONAR, ROISK, BOISK, +FOE, DATA, ESMOE,T
DIMENSION PMAX (50,6,2), DICE(5), NROW(2), CPA(50,1),
C RADAR(50,2C), SONAR(50,20), DATA(50,20,2), XFLCT(50), VPLCT(50),
C +FOE(50,4), INTA(50,6), ESMOE(100,5), MINEX(10), R(4)
COMMON PMAX, DATA, RADAR, SCNAR, FCE, INTA, ICVEW, CPA,
C ICVE, NBLUE, NCLR, ID, TIME, EPS, ISEEO, ISEEO2, ISEEO3,
C DICE, NWRTIE, NMAX, ESMOE, IFOE, DATE, MTIME, MINDEP,R, XPLCT, YPLCT
M = 0
NLNIT = NROW(IFOE)
PLNIT = NRCW(ID)
DC 35 N = 1,NLNI
IF (DATA(N,1,IFOE) .LE. 0) GO TO TC 35
C FASSIVE DETECTION OF AN ACTIVE SCNAR
IF (DATA(N,1,IFOE) .LE. 0) GO TO TC 35
RANGE = 2 * DATA(N,1,IFOE) * PMAX(1,N,5)
IF (CPAN(1) .GT. RANGE) GC TC 35
A = P+1
TC 35 IA = 1,7
33 CONTINUE
SCNAR(M,18) = CPA(N,1)
SCNAR(M,19) = CPA(N,2)
SCNAR(M,17) = DATA(N,2,IFOE)
SCNAR(M,12) = DATA(N,7,IFOE)
SCNAR(M,2C) = CPA(N,1)*(DICE(N) -.5) + CPA(N,1)
25 CONTINUE
IF (M .EQ. 0) GO TO 50
C PRINT OUT ANY PASSIVE DETECTIONS OF ACTIVE SCNAR
WRITE (NWRITE,401) DATA(I,1,ID)
4C FCOMMON (7,7) DETECTIONS OF ACTIVE SONAR'S BY UNIT',14,
C '3X,' TYPE '2X,' BEAR'Neill '2X,' APPROX RANGE'
4C 43 IF = 1,M
WRITE (NWRITE,42) SONAR(11,13), SONAR(11,15), SCNAR(11,20)
42 FCOMMON ( '16,119,1111)
43 CCTINUE
WRITE (6,44) DATA(I,1,ID)
44 FCOMMON (7,7) UMPIRE DETECTIONS CF ACTIVE SCNAR'S, UNIT',15,
C '4X,' ID '4X,' DEPTH 'CSE 'X', '4X,' SPD 'RCF BRG'
4C 46 II = 1,M
WRITE(6,45) (SONAR(I,J), JJ=11,15)
45 FCOMMON ( '16,215,1X,715)
46 CCNTINUE
50 M = M
TC 6C N = 1,NLNI
IF (DATA(N,1,IFOE) .LE. 0) GO TO 6C
NTGT = DATA(N,7,IFOE) - (IFOE * ICGO)
IF (NTGT .GE. 200 .AND. NTGT .LT. 300) GO TO 60
IF (PLAT .NE. 1 .OR. DATA(I,3,IC) .EQ. 0) GO TO 53
IF (NTGT .GE. 100 .AND. DATA(I,3,IC) .EQ. 1) GC TO 52
IF (DATA(N,3,IFOE) .NE. DATA(I,3,IC)) GO TO 53

52 B2 = 2.0
C = 5
A = FLOAT ( DATA(N,2,IFOE) ) / 5.0 + 15.
CC TO 54

53 B2 = (FLCAT(DATA(N,2,IFOE)) - 20.0) / (-2.0)
IF (DATA(N,2,IFOE) .LE. 20) B2 = 0.0
IF (DATA(I,2,IC) .GT. 18) GC TO 60
C = .4
IF (DATA(N,2,IFOE) .GT. 15) C = .6
E = C * FLCAT(15 - DATA(N,2,IFOE)) + DATA(N,2,IFOE)

54 E = DATA(N,2,IFOE) + B2
IF (X .GT. 1.0) A = X
Y = SQRT(B2 * (1.0 - (X/A)**2))
IF (Y .GT. DATA(I,11,IC)) X = DATA(I,12,IC)
RANGE = Y * PRMAX(I,N,5)
IF (CPA(I,1) .GT. RANGE) GO TO 66
K = MM+1
CC 55 IK = 1.7
SCNAR(MM+10) = DATA(N,IK,IFOE)

55 CONTINUE
SCNAR(MM,18) = CPA(I,1)
SONAR(MM,15) = CPA(I,2)

66 CONTINUE
IF (M .EQ. MM) GO TO 99
WRITE(NWRITEGOAT) CATA(I,1,1,10)

50 FCMAT(1/1,11) 'PASSIVE DETECTIONS BY UNIT',I4,'/4X',
C 'COURSE',S,2X,'SPEED',S,3X,'DEPTH',S,3X,'RANGE',S,1X,'BEARING'
M1 = M+1
CC 93 II = M1,MM
WRITE(NWRITE,92) SONAR(II,14),SCNAR(II,12),SCNAR(II,13),
C SONAR(II,1),SONAR(II,19)

52 FCMAT(1,38,3X,216)
93 CONTINUE
WRITE(694) DATA(I,1,1,10)

54 FCMAT(1/1,11) 'umpire CCPY, PASSIVE SONAR DETECTIONS, UNIT ',I4,'/
C 'X',S,4X,'ID',S,8X,'CSE',S,4X,'X',S,4X,'Y',S,8X,'TYPE RAE ERE*'
D1 = 95 II = M1,MM
WRITE(695) (SONAR (II,1J), JJ = 11,19)

55 FCMAT(1,315,2X,615)
56 CONTINUE
59 RETURN
93 END
FORTRAN PROGRAM CHANGE

INTEGER DATA
DIMENSION DATA (5C, 2C)
NCOL = 15
READ = 5
WRITE = 6
DC 9 I = 1, 50
DO 8 J = 1, 20
DATA (I, J) = 0
8 CONTINUE
9 CONTINUE
10 WRITE (WRITR, 11)
11 FORMAT (* ENTER 1 FOR RED, 2 FOR BLUE, FORMAT 111.*/
READ (NREAD, 20) IDENT
2C FORMAT (111)
IF ((IDENT GT. 2) OR (IDENT LT. 1)) GO TO 10
ICISK = IDENT + 2
READ (ICISK, 30) ZTIME, NROW
3C FORMAT (F6. C, 13)
DC 32 I = 1, NROW
READ (ICISK, 31) (DATA(I, J), J = 1, NCOL)
31 FORMAT (1514)
32 CONTINUE
35 WRITE (WRITE, 40) ZTIME, NROW
4C FORMAT (* U1 IS , F6. C, ' Z TIME. YOU HAVE ',13, ' UNITS. */
C * ENTER THE ROW NUMBER OF THE UNIT YOU WISH TO CHANGE THE */
C * STATUS OF. 112. "0" WILL INDICATE NO CHANGES TO ANY UNIT. */
41 READ (NREAD, 42) N
42 FORMAT (112)
IF (N LE. C) GO TO 59
IF (N GT. 1) GO TO 39
45 WRITE (WRITE, 46) (DATA(N, I), I = 1, 4)
46 FORMAT (* CODE UNIT ID = ', I4, ’/
C : 2 SPEED IS ', I4, ’ KTS. ’/
C : 3 ALT/DEPTH = ', I4, ’ *10**3 ’/ 1 = SHALLOW, 2 = DEEP ’/
C : 4 COURSE IS ', I4, ’ T. ’/
C : WRITE (WRITE, 47) (DATA(N, I), I = 12, 15)
47 FORMAT (* 5 SURFACE SEARCH RADAR: ', I4, ’ 1 = CN ’/
C : 6 AIR SEARCH RADAR: ', I4, ’ C = EFF ’/
C : 7 SONAR: ', I4, ’ /
C : 8 HF COMMUNICATIONS: ', I4, ’ /
C : ENTER THE CODE NUMBER OF THE LIKE YOU WANT TO CHANGE,’/
C : III. "0" MEANS NO CHANGES TO THIS UNIT. */
48 READ (NREAD, 48) NCODE
49 FORMAT (111)
IF (NCODE .LE. 1 .OR. NCODE .GT. 6) GO TO 35
WRITE (WRITE,50) NCODE
50 FORMAT (* ENTER THE NEW VALUE FOR CODE NO. '12', 114', ') READ (NREAD,52) DATA (N,NCODE)
IF (NCODE .GT. 4) NCODE = NCODE + 7
C DATA (N,NCODE) = DATA(N,8)
IF (NCODE .EQ. 4 .AND. DATA(N,NCODE) .GE. 300) DATA (N,NCODE) = 0
C I TYPE = DATA(N,7) (IDENT * 100)
IF (NCODE .EQ. 3 .AND. DATA(N,NCODE) .GE. 300) DATA (N,NCODE) = 0
C DATA(N,NCODE) = 0
C TO 45
55 WRITE (WRITE,60)
60 FORMAT (*) IF YOU WISH TO BRING AN OUT OF PLAY UNIT '"', /
C INTO PLAY OR PUT A UNIT OUT OF PLAY ( I.E. SINK IT) '" , /
C ENTER THE ROW NUMBER OF THAT UNIT, 117. "C" MEANS '" , /
C NO UNITS TO PUT IN OR OUT OF PLAY '" , )
READ (NREAD,42) IDNO
IF (IDNO .EQ. 0) GO TO 90
IF (IDNO .GT. NRCW) GO TO 59
WRITE (WRITE,46) (DATA(ICNO,J),J =1,4)
WRITE (WRITE,66) ICNO
66 FORMAT (*) ENTER THE NEW UNIT ID, 114. "C" PUTS THE UNIT '" , /
C OUT OF PLAY. ANY OTHER VALUE BRINGS THE UNIT INTO '" , /
C PLAY WITH THAT ID NUMBER '" , )
READ (NREAD,60) DATA( IDNO,1)
68 FORMAT (114)
IF (DATA( IDNO,1) .NE. 0) GO TO 70
DATA( IDNO,2) = 0
DATA( IDNO,6) = 0
C TO 59
70 WRITE (WRITE,72) ICNO,DATA( ICAC,5), DATA( ICNO,6)
72 FORMAT (*) UNIT ,13 , IS NOW AT '" ,25', /
C ENTER FORMAT 1 FIRST THE NEW X CO-ORDINATE '" , /
C THEN, AFTER THE SECOND > APPEARS THE NEW Y CO-ORDINATE , '/
READ (NREAD,74) (DATA(IDNO,J),J =5,6)
74 FORMAT (114)
C TO 59
76 FORMAT (*)
READ DISK
WRITE (DISK,30) ZTIME,NROW
DC 92 I=1,NROW
WRITE (DISK,31) (DATA(I,J),J =1,NCOL)
52 CONTINUE
55 STOP
END
FORTRAN PROGRAM UPDATE

INTEGER RDATA, BDATA, BDISK, RDISK
DIMENSION RCATA(50,20), BDATA(50,20)
NCOL = 15
RCISK = 3
BEISK = 4
RNCW = 50
RDISK = 2
READ (RDISK, 10) ZTIME, NRED
10 FORMAT (F6.6,113)
DO 20 I = 1, NROW
DO 15 J = 1, NCOL
RCATA(I,J) = 0
BDATA(I,J) = 0
15 CONTINUE
DO 60 J = 1, NRED
20 CONTINUE
DO 60 J = 1, NRED
READ (RDISK, 40) (RDATA(I,J), J = 1, NCOL)
40 FORMAT (1514)
DO 60 CONTINUE
READ (BDISK, 77) ZTIME, NBLUE
77 FORMAT (F6.6,113)
DO 83 I = 1, NBLUE
READ (BDISK, 81) (BDATA(I,J), J = 1, NCOL)
81 FORMAT (1514)
CONTINUE
85 REWIND NDISK
WRITE (NDISK, 90) ZTIME, NREC, NBLUE
90 FORMAT (F6.6,213)
DO 92 I = 1, NRED
REWIND NDISK
WRITE (NDISK, 91) (RDATA(I,J), J = 1, NCOL)
91 FORMAT (1514)
CONTINUE
DC 93 I = 1, NBLUE
REWIND NDISK
DC 93 I = 1, NBLUE
REWIND BDISK
REWIND RDISK
955 STOP
END
BIBLIOGRAPHY


<table>
<thead>
<tr>
<th>No.</th>
<th>Copies</th>
<th>Initial Distribution List</th>
</tr>
</thead>
</table>
| 1.  | 2      | Defense Technical Information Center  
Cameron Station  
Alexandria, Virginia 22314 |
| 2.  | 2      | Library, Code 0142  
Naval Postgraduate School  
Monterey, California 93940 |
| 3.  | 10     | Department Chairman, Code 55  
Department of Operations Research  
Naval Postgraduate School  
Monterey, California 93940 |
| 4.  | 1      | Professor A. F. Andrus, Code 55As  
Department of Operations Research  
Naval Postgraduate School  
Monterey, California 93940 |
| 5.  | 1      | LT Ellen F. Roland, USN, Code 55Ro  
Department of Operations Research  
Naval Postgraduate School  
Monterey, California 93940 |
| 6.  | 1      | LT Kevin J. Kelley, USN  
211 Threadneedle  
San Antonio, Texas 78227 |
| 7.  | 1      | Professor J. K. Hartman, Code 55Hh  
Department of Operations Research  
Naval Postgraduate School  
Monterey, California 93940 |