FILE

INTERNAL DOCUMENT 276

I.O.S.

Programs for Acoustic Navigation on the BBC Master 128

by S.R.J. Williams

IOS Internal Document No. 276

1987

NATURAL ENVIRONMENT RESEARCH COUNCIL

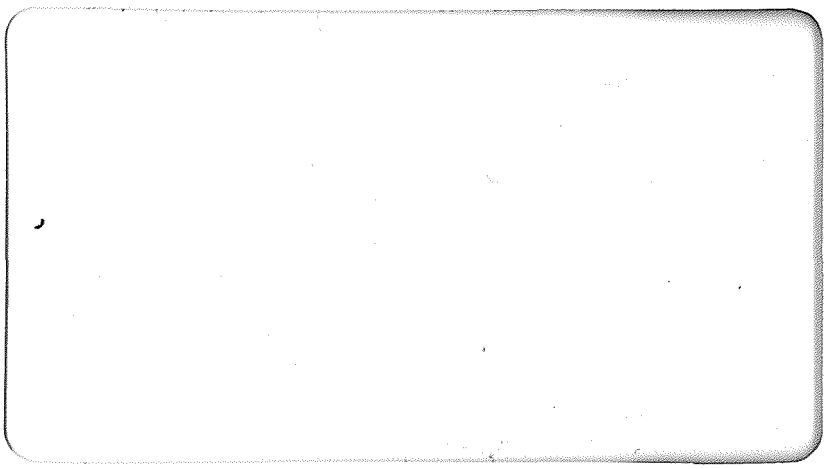
INSTITUTE OF OCEANOGRAPHIC SCIENCES

INSTITUTE OF OCEANOGRAPHIC SCIENCES

Wormley, Godalming,
Surrey GU8 5UB
(042-879-4141)

(Director: Dr. A. S. Laughton, FRS)

Bidston Observatory,
Birkenhead,
Merseyside L43 7RA
(051-653-8633)



Programs for Acoustic Navigation on the BBC Master 128

by S.R.J. Williams

IOS Internal Document No. 276

1987

CONTENTS

<Names in brackets indicate name of View text file or Autocad drawing file>

<T2 on ACNAV View disk>

CHAPTER 1 INTRODUCTION

This manual describes the software developed for use on a BBC Master computer to be used in conjunction with the IOS ACNAV system for acoustic navigation of ships and underwater devices. Three main programs NAVPLT, BEACON and NAVVY are provided for users of the IOS 5 kHz and 10 kHz deepwater acoustic navigation systems (Rouse, 1987). The programs utilise two-way travel time data which are read manually from graphic recorders. The programs offer many advantages over the manual calculator and compass technique (increased accuracy, storage of data, calculation of speed and heading, calculation of ranges to faint beacons, recalculation of fixes using updated beacon positions). In certain applications the manual technique may be adequate and it is valuable as an occasional check that correct data are being used in the programs.

The system allows the user to check that the fix is reasonable by displaying a plot of the fix geometry on the VDU screen. This is useful in cases where inaccurate or suspect ranges are used, as is often the case at long distances from the beacons or in bad weather. An alternative automatic fixing program without screen display would be faster but would not enable the user to identify bad ranges.

This manual is intended primarily as a guide to users who need understand little about the structure of individual programs. Additional material is provided for persons wishing to modify or improve the programs in the future.

The main program is NAVPLT by which travel times are converted into distances and the position of a ship or remote device is calculated. The raw data and results may then be stored on floppy disk for future reference. These activities may be performed online to produce a position fix within 1 minute of receiving the raw data values. On average though, two people may comfortably produce fixes every 5 minutes for extended periods (or every 10 minutes if a remote device is also being fixed). The user may accept the fix recommended by the program or alternatively he may select an alternative position if he suspects that some of the input data are unreliable.

The program BEACON is called from NAVPLT in order to create or edit the Beacon Data File (BDF). This file contains the latest estimates of beacon positions together with harmonic mean sound velocity, transponder propogation delays and the initial limits of the screen plot area. The BDF is accessed by NAVPLT during program initiation.

The program NAVVY is used from time to time as more data become available to recalculate the relative and absolute positions of fixed transponders (beacons) using reiterative procedures. The new values are placed in the BDF for use by NAVPLT. Recalculation increases the accuracy of subsequent fixes. In addition previous fixes stored on a disk may also be recalculated by NAVPLT with greater accuracy. Absolute positioning requires the input of geographical fixes obtained by satellite- or shore-based positioning systems and the accuracy varies according to the accuracy of the system used.

The error gradient method used by NAVVY for determining minimum error positions of the ship and beacons could be used for routine fixing if ranges are reliable but the process would be very slow in BBC BASIC.

Note that NAVPLT takes up a large amount of memory. The program BAS128 on the program disk is run before loading NAVPLT in order to make full use of the Master's available memory. This does have some unexpected effects such as the corruption of some variables when the program gets very large. These effects are seldom obvious so take care if rewriting!

<T3 on ACNAV View disk>

CHAPTER 2 OPERATION AND USE

This section describes how the programs are used in an ACNAV survey.

2.1 SETTING UP AND DEPLOYMENT

1) Refer to Rouse (1987) for details of deploying the beacons and setting up the laboratory equipment.

2) Determine the depth:sound velocity profile for the area by XBT, CTD or Matthews/Carter Tables and calculate the harmonic mean velocity (see Rouse 1987).

3) Make notes of individual transponder propagation delay times (typically c. 0.025 s), and elevation of transponder above anchor weight (typically 100 or 200 m).

4) Designate individual beacons by colours RED, GREEN, BLUE, ORANGE, BROWN, BLACK.

5) Obtain best possible estimate of deployment position as possible from normal navigation system (or from ACNAV if 2 or more beacons are in range).

6) Set up the BBC microcomputer system (as described in Chapter 3).

7) Load and run NAVPLT (as described in Chapter 4).

8) Create a new BDF using the information referred to above.

9) With the ship hove to, as soon as the transponder has hit the seabed the range will stop increasing. Ranges from the transponder can now be used for navigation.

10) As soon as one or more beacons are in place acoustic navigation may commence.

2.2 DETERMINING THE BEACON NET

The positions of the beacons relative to one another may be determined by 2 methods:

2.2.1 Firstly the ship may do a BASELINE CROSSING SURVEY (Appendix 4). This involves steaming in a straight line, at constant speed across the baseline which connects each pair of beacons. Preferably the baselines

are crossed at ninety degrees, equidistant from the beacons. Horizontal ranges at the sea surface are calculated using NAVPLT at short, regular time intervals (typically every 1 minute at 4 knots, i.e. at 120 m intervals). For each pair of beacons the sum of the 2 ranges is plotted against time on graph paper. The minimum summed range is then determined by interpolating a hyperbola through the points. The relative positions of the beacons are then determined by constructing triangles from the baselines.

2.2.2 The second method is to use the program NAVVY. This program takes a group of fixes determined using the latest estimated beacon positions. A minimum number of fixes are required depending on the number of beacons. The program adjusts the relative beacon positions until the range errors are minimised. With the new set of beacon positions the fixes are then recalculated and fixes with large errors are discarded. The whole sequence is repeated until no further reduction in errors takes place. Better results are obtained if there is a wide geographical spread of fixes but good results are obtained with a few fixes in the same area.

The 2 methods may be used seperately or together but repeated use of the computer method will produce increasingly better results as more and more good fixes are obtained.

2.3 DATA RECORDING

It is advisable to keep a record of all ACNAV data. Logsheets (Figure 5) are provided for the watchkeeper to record raw and calculated ranges and calculated positions for every fix. NAVPLT can provide an online print-out of all the relevant information.

2.4 SAVING DATA TO DISK

Better still the range and position data may be stored in a floppy disk file for later access. This option is provided by NAVPLT (Section 4.2.8). If fix data are to be saved on disk then a Fix Data File (FDF) must be created. These may be created during an ACNAV session but the process is fairly time-consuming and it is recommended that FDF's are created beforehand. Create files with sequential names (e.g. D163A01, D163A02.....) and fill files completely before starting a new one. Each file may contain up to 100 fixes and 7 files will fit on one side of an 80 track disk. When creating a file you should provide the name of the file which will contain the preceeding fixes

On each data disk there must also be a BDF and a 'trans' file (Appendix 5). The latter is used for temporary storage of variables during program execution; it can be created from NAVPLT (Section 4.2.10).

2.5 PLOTTING

Normally fixes will be plotted by hand during the survey. It is best to use a base grid with a seperate permatrace overlay for each ACNAV session. NAVPLT provides fix coordinates based either on the local metric grid or on Latitudes and Longitudes (in degrees, minutes and decimal minutes). To date the best choice of base grid has been found to be a 1:50,000 scale chart with grid lines every 1 minute of latitude and longitude. (This can be produced by the ABC computer system). Fixes can then be plotted by hand without need of a ruler or dividers. Establish a set of symbols for different fix types (e.g. ship, remote, ACNAV/SATNAV/GPS/LORAN C.....) at the beginning of the survey. Plot beacon positions on the chart and use a beam compass to check the fixes from time to time.

As yet there is no facility for producing hard plots by computer (see 7.2).

2.6 REPROCESSING

From time to time during the survey the beacon positions may be reestimated using NAVVY and the BDF may then be updated accordingly using NAVPLT or BEACON. (This must be done when no ACNAV processing is taking place). It is then possible to go back and recompute all previous fixes using the new beacon positions by using NAVPLT's reprocessing option (Section 4.2.9). To avoid confusion though it is generally better to leave all reprocessing until the end of the survey.

<T4 on ACNAV View disk>

CHAPTER 3 COMPUTER HARDWARE

The following computer equipment is required:

1) A BBC Master 128K microcomputer (smaller BBC's have insufficient memory).

2) A 5.25" floppy disk drive unit (40 and or 80 tracks, single or preferably, double-sided).

3) A supply of floppy disks. (These should have been formatted using the FORMAT command on the BBC).

4) The 2 ACNAV program disks, one containing BOOT!, BAS128, NAVPLT, BEACON and the second with NAVVY.

The following items are optional but recommended:

5) An 80 track BBC compatible printer e.g. EPSON FX-80 with a box of listing paper and spare ribbon.

6) A supply of ACNAV data forms for recording data.

7) A copy of this manual (the text is stored on a VIEW disk and the drawings are stored on an AUTOCAD disk).

8) Manuals for the equipment.

<T5 on ACNAV View disk>

CHAPTER 4 NAVPLT

4.1 INTRODUCTION

NAVPLT is designed to be useable by any numerate person with a little practice. Apart from typing in data values and filenames the user is only required to select from a number of options displayed on the screen in the form of menus or simple alternatives. Most user errors are trapped without causing program failure. The program is designed so that fatal errors cause minimal disruption and restarting is straightforward using the function keys. Not all errors are predictable but most unexplained errors will be a result of incorrect datafile handling.

The basic purpose of NAVPLT is to calculate the coordinates of position fixes given a number of ranges from the ACNAV system. The ranges are entered at the keyboard for each fix. There are several ways of calculating a fix, depending on the number of ranges available. The basic methods are called the 'Cokhat', 'Matrix' and 'Manual' methods and the differences are described subsequently (Section 4.2.7). NAVPLT automatically recommends a particular method in each fix situation.

The user may accept the recommended fix or he may decide that certain ranges are less reliable than others and choose an alternative position. To assist this decision NAVPLT displays a screen plot of the area of interest. Upon the plot may be displayed any or all of: local kilometer grid, beacons, earlier ship positions, earlier remote positions, range circles, and circle intersection point code numbers. The area of the plot may be redefined by the user.

A large part of the program is devoted to data filing operations. These operations include saving data to disk, reading in old fix data, maintenance of the 'trans' variable file, creation of new files, reading disk catalogues and accessing the BDF file.

The following sections describe the detailed operating instructions for NAVPLT. After starting-up all requests for user input are highlighted in BLUE with an optional audible prompt. At any time after the start-up sequence the program can be stopped by pressing <escape> and then restarted by hitting <f3> for the main menu.

4.2 DETAILED OPERATING INSTRUCTIONS

The basic sequence is illustrated in the flow diagram (Figure 1). All prompts for user action by the operator are highlighted in blue with an optional audible 'bleep'. Except in critical places the usual response to program prompts is to hit the return key <RTN>. If done by mistake the results are not usually fatal. For certain important decisions the user must hit either <Y> or <N>. All data entries must be followed by a <RTN>, this enables typing mistakes to be corrected before entry.

4.2.1. LOADING

1) Switch disk drives to 40/80 track as necessary.

2) Insert program disk in Drive 0.

3) Hold <SHIFT> and press <BREAK>. Program loading takes 1 minute.

4.2.2 START-UP

1) The Title page appears.

2) Insert Data Disk in Drive 1 and type <1> <2> or <3>. The data disk is then catalogued upon the screen. Check that the file 'trans', a beacon datafile (BDF) and sufficient empty fix datafiles are present (labelled 'D163A01' or similar, where D163 is a cruise identifier, A is a disk identifier, 01 is the filenumber on the disk). You will need to keep a written record of which files have what data and which are empty.

3) Should you wish to CREATE any new files enter <Y> and follow the prompts (see Section 4.2.10).
You only need to create a new 'trans' file if it does not already appear on data disk catalogue.
You only need to create a new BDF if one does not already exist on data disk catalogue.
You will need to have sufficient files available to hold fix data from the whole of the following navigation session (100 fixes per file, 7 files on each side of a disk =1400 fixes per disk =116 hours per disk at a 5 minute fix interval). Start a new file at the start of each new navigation session unless the sessions are brief. The procedure for creating new FDF's is described in Section 4.2.10.
Usually you will not need to create files so press <RTN>.

4) The current BDF name is displayed. Type <RTN> to accept it or enter <"newname"> where "newname" is the name of a preferred, existing alternative BDF on disk.

The BDF will be read and you can choose whether to list the data to the screen and printer. It is a good idea to do so at the beginning of a new ACNAV session. Then examine the data and if it is correct type <RTN> or <N> to call up BEACON and edit the BDF.

5) "Do you want to reset switches RTN/N"
Switches (Section 4.2.16) include:-
* Diagnostic printouts of variables at certain places in the program. This is only used for development or debugging.
* Sound; the bleep prompt can be switched on or off.
* Print out: If you require print out then type <Y>. Connect the printer and ensure it is online then press <RTN>.
* Remote transponder: if you might want to fix a remote during the session type <Y> and then for each fix after entering ship data you will be asked whether you wish to enter remote data.

After setting the switches the values are recorded to 'trans' on disk. Then the Main menu is displayed.

4.2.3 MAIN MENU

The main menu may be called by hitting <f3> when the program has stopped. The main menu options are:-
* Enter data * Plot * Fix * Display other options
* Calculate * Reprocess data * Renominate FDF
* Disk command * Access BDF * Create new files
* Plot FDF data on screen * Save data to FDF
* Inspect FDF * Reset switches * Stop

MAIN MENU OPTIONS

<RTN> or <1> ENTER DATA: This is the usual choice (See Section 4.2.4).

<2> PLOT: Plots data on screen (See Section 4.2.6).

<3> MAKE A FIX: Decide on position of ship/remote (See Section 4.2.7).

<4> OTHER OPTIONS: Prints extended menu.

<5> CALCULATE: Calculates circle intersection coordinates, autofix and matrix fix of ship/remote (See Section 4.2.5).

<6> REPROCESS DATA: User selects data from disk to be reprocessed (See Section 4.2.9).

<7> RESELECT FDF DATAFILE: User nominates a different FDF to receive saved data.

<8> COMMAND DISK: User can issue an MDOS command e.g. CAT, DR.0 etc.

<9> ACCESS BEACON DATA: Program reads in data from BDF file.

<10> CREATE NEW FILES: Used to create new, empty FDF, trans or BDF files on data disk.

<11> SAVE TO DISK: Save current fix data to FDF.

<12> INSPECT DATAFILE: Allows user to inspect the header and fix data of any datafile on disk.

<13> RESET SWITCHES: Switches for diagnostics, sound, printer and remote can be reset.

<14> STOP: Stops the program.

4.2.4 DATA ENTRY (Main Menu option <RTN>)

1) Enter Julian day number and GMT in DDD, HHMM format e.g. '365,1233 '. Values are displayed in red banner on LHS of screen until superceded by new values. You may type <RTN> for day number if it has not changed since the last fix in the current session.

Banners reading 'SHIP' and 'DATA not yet fixed' are displayed at the top of the graphics screen.

2) * For each beacon enter SHIP:BEACON ranges either:-
a) in seconds TWT as read from EPC or
b) in metres slant range read from MUFAX.
* Entering 'X' at any time will restart data entry from 1) if you have made a mistake.
* For each beacon the minimum permissible ranges in m. or s. are indicated. Smaller ranges will not be accepted as they imply either that the ship is submerged or that the beacon has ascended.
* If no range is available press <RTN>.
* The time in s., the slant range in m. and the calculated horizontal range in m. are displayed (and printed out if applicable). The latter can be used to plot arcs manually on a chart, if required. All values are corrected for the beacon delay time.

3) If the remote fix option was switched on in the Start-Up sequence (4.2.2) now choose whether you wish to fix the remote at this particular fix time <Y> or <N>. Otherwise the program proceeds to 7) below. Similarly, if you press <N> the program goes to 7).

4) * For each 5 kHz beacon, enter REMOTE:BEACON:SHIP ranges as measured from the EPC (including beacon delay time) in seconds 1WT (one way time).
* Minimum times (i.e. the 1WT from the beacon to the ship) are indicated, smaller values will be rejected.
* If no range is available (as for the 10 kHz beacons) press <RTN>.
* If no range was available from the ship to a particular beacon then no remote range will be requested for that same beacon because the remote:beacon range could not be calculated. In rare instances a remote:beacon:ship range may be available when the corresponding ship:beacon range is not. In such circumstances it would be possible to fix the ship and derive a predicted ship:beacon range and then redo the whole fix by reentering all the range data including the predicted range. (PROGRAMMERS NOTE This would of course be very slow and it might be worth modifying NAVPLT in the future so that remote data is not entered until the ship position has been fixed).

5) Enter depth in m. of the remote below the sea surface. This should be determined, if possible, by measuring the remote height above seabed and subtracting the value from the total water depth or from a pressure determination at the remote. If the value is unknown, guess it, using calculated depths of previous fixes and the amount of wire out as a guide.

The accuracy of the remote cokhat fix depends on the accuracy of the calculated horizontal BEACON:REMOTE ranges which themselves depend on the accuracy of the remote depth estimate. If the guess is wild, don't despair as the Matrix fix will function given 3 good REMOTE:BEACON:SHIP ranges and a good SHIP:REMOTE range.

(PROGRAMMERS NOTE It would be possible to reiterate the cokhat fix using different remote depths and find the depth which gives the smallest error but this would be too slow for online navigation using BBC BASIC).

If the depth entered is such that the depth difference between the remote and any beacon is less than the calculated range to that beacon the program will inform you and give you 2 options:-

<T> re-enter all the REMOTE:BEACON:SHIP l WT's. Before re-entering each range you will be told the minimum lWT based on the proposed remote depth.

or <D> enter a larger remote depth, in which case you will be told the minimum acceptable depth considering the range to the beacon in question.

6) * Next you must enter the range in s. lWT from REMOTE:SHIP as read from the EPC (no delay).

* If the implied range is less than the estimated remote depth (see above) this implies a submerged ship or excessive remote depth estimate. The program would then ask you to choose:-

either <RTN> to reenter a larger REMOTE:SHIP lWT value,
or <Z> to go back to 5) and enter a smaller remote depth estimate.

You must choose to reenter whichever value was the least reliable.

* When all is satisfactory the horizontal ranges from each beacon and the ship to the remote will be printed out.

7) * When all data have been entered, choose:-

<X> to reenter data from l) if you have made mistakes,

<RTN> to finish data input and automatically call up routines for calculating, plotting, fixing and saving the data (4.2.5 to 4.2.8).

<A> to automatically run through the sequence of calculation, plotting, fixing and saving using default settings and the recommended fix type.

<S> to save raw unfixed data to disk and stop.

<T6 on ACNAV View disk>

4.2.5 CALCULATIONS (Main Menu Option <5>)

Called automatically following data input (Section 4.2.4).

1) Normally all calculation is carried out automatically but if there are more ranges available than are necessary for the matrix fix (3 for ship, 3 + 1 ship:remote for remote) you will be asked which ranges to use (See 4)) because the alternative of automatic reiterative selection is too slow.

2) (PROCcalc) Program automatically calculates coordinates of the intersection points (IPs) of the beacon:ship range circles projected on the horizontal sea-surface.
 * For remote data the range circles are projected at the estimated remote depth before IP calculation.

3) (PROCcokhat) Program selects the 3 IPs lying closest to one another and takes their centroid as the XY position for the ship autofix.
 * The Std. Devn. of the autofix indicates the goodness of fit of the fix.
 * The numbers of the 3 chosen IPs are displayed.
 * If applicable the same is done for the remote data.

4) (PROCMATFIX) Program calculates a matrix solution for the ship XY position by assuming the ship is on the sea surface. 3 beacon ranges are required for this.
 * If more than 3 ranges are available the program asks you which 3 to use. You may then plot the range circles to help you decide.
 * (PROGRAMMERS NOTE It would be possible for the program to automatically compare all possible combinations of 3 ranges and select that with the minimum error but the code has yet to be written. Also the time factor may be excessive: x4 for 4 ranges, x10 for 5 ranges).
 * The fix error is taken to be the average error of the 3 ranges.
 * If remote data are available a matrix fix is calculated using 3 beacon ranges and the 1 ship range. If >4 ranges are available user must select which 4 to use, the ship:remote range may be excluded if the range or the ship position are not considered very reliable.
 * The program normally cannot utilise remote ranges from a particular beacon without having been given ship ranges from that beacon (but see Section 4.2.4(4) for a way to manage this).

4.2.6 PLOTTING (Menu Option <2>) (PROCplot).

Called automatically after data input and also from various procedures.

1) PLOT MENU:- Select Plotting Window.
   * If there are no current fix data to plot select:-
   EITHER:- <M> to go back to main menu
   OR <RTN> to proceed with plot, for example if you wish to plot old FDF fixes.

2) <RTN> Selects current window which is:-
   EITHER a) the initial window as defined by the BDF,
   OR b) the latest window defined in the current session.

3) <I> Selects default window which is:-
   EITHER a) as defined by the BDF (which should cover a wide area including all the beacon locations),
   OR b) as previously redefined by option <XN> (See 6)).

4) <N> To define a new current window. You must enter new values of xmin,xmax,ymin and ymax in local Km coordinates. Refer to previous plots to select the desired window which will usually include the previous and current fix positions and the expected positions of the next few fixes.

5) <A> Automatically scaled to cover cokhat 'triangle'. This plots a small window centred on the automatic cokhat fix enabling you to distinguish the seperate IPs when otherwise they may be confused. The width of the box is scaled as a function of the standard deviation of the fix, minimum width: 2.2 km (to show grid lines).

6) <XN> To change default window.
You must enter xmin, xmax, ymin & ymax as in 4) above.

7) <MM> To escape to Main Menu.

This terminates any preset program flow and stops the program. Then use <f3> key to call main menu.

8) <X> To exit.

Exit plotting procedure and return to the procedure where you were previously (POST FIX menu, Matrix Fix, Main Menu.....).

9) Having selected the window, a grid is drawn labelled in local Km coordinates relative to the local grid origin

(see Figure 3).

Beacon positions are indicated by white '*' s which are numbered. If current fix positions of the ship or remote are available they are indicated by a white 'S' or 'R'.

10) You will be asked how many previous fixes you want plotted. Give any number so long as the earliest fixes are on an FDF on the same drive. Fixes are read from the current FDF and, if necessary, from any previous FDF on the same side of the disk.

If the remote option was selected you must specify whether to plot:-
ship fixes <S>, remote fixes <R> or both <RTN>.

Old fixes are read from the disk, the most recent first and plotted as red 's' or 'r' symbols, linked in time sequence by red lines.

11) If remote data are available select which current data to plot:- ship <RTN> or remote <R>.

A red banner indicates the choice selected and range circles are plotted in white at the sea surface (for ship) or at the remote depth (for remote). Circle IPs are indicated by numbered blue '+' s.

12) The current MATRIX fix position (red cross) and COKHAT fix position (blue cross) are displayed.

13) Finally, select:-
<RTN> to finish plotting (and, in the normal sequence, proceed to fixing)
OR <P> to return to plot menu (See 1)).

4.2.7 POSITION FIXING (Menu Option <3>) (PROCfix)

1) If remote data are available and the ship is fixed and remote IP calculations have been performed you must select:-

<RTN> to fix ship OR <R> to fix remote.

2) Do you wish to plot data?:-

<RTN> if you do (4.2.6) OR <X> if you do not.

3) The 3 chosen IPs for the automatic COKHAT fix are listed e.g. 061003= IPs 6, 10 & 3. The automatic fix method which gave the least acceptable error is highlighted in RED. This will be MATRIX or COKHAT or, if these both gave unacceptable errors MANUAL will be recommended.

4) FIX TYPE MENU

Choose:-

<RTN> To choose the recommended fix method as highlighted. Then proceed as for 1, 2 or 3 as applicable below.

<1> COKHAT: The fix is centered at the centroid of the closest 3 IPs as shown by the BLUE cross. The Z value =0 (for ship) OR =estimated remote depth (for remote).

<2> MANUAL: You must enter between 0 and 4 IP numbers, repeating particular IPs if required to give them extra weight. The fix is placed at the centroid. The Z value is as for <1> above.

<3> MATRIX: The fix is as displayed by the RED cross. The Z value =0 (for ship) OR as determined by the matrix method (for remote).

<4> EXIT: If you are not ready to fix, escape to the MAIN MENU.

5) After selecting <1>, <2> or <3>, the XYZ coordinates of the fix are displayed and printed out with the error, and the longitude and latitude. Simultaneously the fix position is marked by a white dotted line cross and a banner is displayed indicating that the ship/remote has been fixed.

6) Choose either:

<RTN> to COMPARE/PREDICT
With the chosen fix position, this option calculates the distance, bearing and speed of the ship/remote from the last fix saved to disk. This information may allow user to reject the fix if values are unreasonable. If the FDF is "undefnd" you will be asked to name an FDF file on the disk or press <RTN> to skip the comparison. Predicted ranges to all beacons from the chosen fix are then displayed and printed out.

or: <N> to skip and proceed to 7).

7) Choose either:

<RTN> to accept the fix and proceed to 8) below.

or <X> to redo the fix in which case the cross is obliterated, the 'fixed' banner is removed and you return to the FIX TYPE Menu 4).

8) The POST FIX MENU is displayed.

(N.B. If remote data are available and unfixed the alternative response indicated by a '*' is displayed.)

You must select from:-

<R> or *<RTN> FIX REMOTE
Calculates the remote position using the current ship position then goes to FIX TYPE MENU 4).

<RTN> or *<S> SAVE TO DISK
* If all is well you will proceed to the saving routine (4.2.8).
* If the ship or remote have not been fixed you will be warned and must choose either:-
<RTN> to go back to FIX TYPE Menu 4) and fix the missing device OR

<C> to carry on and save to disk (4.2.8) without fixing the other device (e.g. when remote data are too poor to warrant fixing).

If <RTN> chosen, you must then select either:
<N> to retain current plot,
or <RTN> to replot.

<F> REDO WHOLE FIX
If you are unhappy about the chosen fix you can opt to redo it. The program stops and the Fix Procedure may be called with key <f6>. The ship and the remote (if

applicable) will both be refixed. If you have fixed the remote and then decide to refix the ship you must choose this option as the remote position is usually dependent on the ship position and must therefore be refixed also.

<C> COMPARE/PREDICT
As in 6) above, but if remote data are available you can choose which device to compare and predict.

<M> MAIN MENU
If you are completely confused go back to main menu and start again from wherever you wish. Program stops. Hit key <f3>.

Normally the fix procedure ends with a call to the save procedure (next section).

4.2.8 SAVING TO DISK (PROCsave)

Called from POST FIX Menu, MAIN Menu & KEY <f7>.

1) FDF Name is displayed. You must select from:-

<RTN> to accept nominated disk datafile and write fix data to it. If all is well data is written, Read-After-Write check is performed, fix-type (Auto,Manual or Matrix) and code are displayed and printed for ship and a 'Data Saved' banner is displayed. Program stops and prompts you to select Main Menu with key <f3> for more data input or whatever.

<NEW> to create a new FDF file. The procedure is described in Section 4.2.10. It is a slow process best done outside an active ACNAV session.

<OLD> to select a (non-full) FDF file other than the nominated one. You must enter the 7 character filename. If the file does not exist on disk an error will occur.

<CAT> to catalog the data disk.

<HDR> to inspect a particular FDF on the data disk.

N.B. If errors occur due to your bad file designation then escape, to the main menu and choose the SAVE option or stop program with <escape> key and hit the <f7> SAVE key. This time specify a different diskfilename, or insert a different pre-formatted disk in drive 1.

2) If the selected FDF is full you must choose:
<RTN> to create a new file as in 1) (NEW)
or <'filename'> to nominate another pre-existing file as in 1) (OLD).

You will then be asked to confirm the drive No. <RTN> or enter an alternative. The normal program sequence following data input terminates here and you must hit <f3> to call up the main menu.

4.2.9 REPROCESS DATA (Main Menu Option 6)

You nominate an FDF file to be read for the input of raw ranges and fix data (ensure that the correct BDF data has been read in). User specifies an individual fix time or a time sequence. Data may be recalculated using updated beacon positions or simply by choosing alternative fix positions with the advantage of hindsight. In the former case the process may be automated so that the same relative fix selections (e.g. Matrix or IP centroid, as indicated by the SCODE and RCODE variables stored on in

the original FDF) are chosen.

4.2.10 CREATE NEW FILES (Option 10, key <f8>)

Used to create new, empty FDF, trans or BDF files on data disk. FDF files are best created outside an ACNAV session because the process is rather time-consuming. 100 fixes are written to each FDF file. Each fix may include remote data.

You enter the new filename (7 characters), beware giving the name of a pre-existing file as this would be overwritten. You then enter the name of the file (if one is present on the same side of the disk) with the immediately preceeding (in time) fix data. The new file will be created (if there is sufficient room on the disk) and the fix data will be written to it.

Insert the data disk and give the drive number. The disk catalogue is displayed.

You are asked whether you wish to create files:-
EITHER <RTN> to end procedure
OR <Y> to proceed and create files.

Next:-
EITHER <TRANS> to create a new 'trans' file (only necessary if one does not appear on catalog).
OR <RTN> to skip.

Next:-
EITHER <BEEK> to create or edit a BDF file. Exits program and chains BEACON from program disk (see Chapter 5). *BEWARE* Any current fix data in program memory will be lost.
OR <RTN> to skip.

Next:-
EITHER <RTN> to skip
OR <FIX> to create a new FDF file.

Enter the new FDF name (7 characters).

The current previous FDF name is displayed. EITHER <RTN> to accept current name OR 'newname' (7 characters) where newname is the name of the FDF with the preceeding fixes.

Then the new file is created for 100 fixes. This takes about a minute. If there is no room on the disk an error will occur. Escape and try again from the main menu with another formatted disk.

If the file is created successfully the FDF inspection procedure is called up (4.2.15).

* Finally you are asked whether you wish to create more files:- EITHER <Y> to repeat the whole procedure OR <RTN> to finish.

4.2.11 RESELECT FDF DATAFILE (Option 7)

User nominates a different FDF to receive saved data.

* The current FDF is displayed:-
  EITHER <RTN> to accept this
  OR type alternative name (7 characters).

* The new name is then saved to the 'trans' file.

4.2.12 COMMAND DISK (Option 8)

You can send an MDOS command e.g. CAT, DR. 0 or 1, DELE., etc. without a preceeding '*'. N.B. Certain commands cause the program to be erased, these include: BACKUP, FORM40/80, COMPACT, COPY and so they should not be used during a navigation session.

4.2.13 ACCESS BEACON DATA (Option 9)

*The BDF name is displayed:-
EITHER <RTN> to accept it
OR type the name of an existing alternative.

*Enter the drive number.

*Do you wish to list the BDF data?:-
EITHER <RTN> to avoid listing
OR <Y> to display data.
* The data are displayed:-
EITHER <RTN> if data are OK
OR <N> if they are not and you wish to edit data or create a new BDF file. The program asks you to put the program disk in drive 0 and hit <RTN>. NAVPLT is exited and BEACON is chained (Chapter 5).

4.2.14 SAVE TO DISK (Option 12, key <f7>)

In the event of program failure (for example due to confusion in diskfile allocation) if raw or fix data are in memory they can be saved to the FDF (see 4.2.8).

If there are no current raw data the procedure finishes.

4.2.15 INSPECT DATAFILE (Option 13)

* This allows you to inspect the header and fix data of any FDF file on disk. The current FDF name is displayed:-
  EITHER <RTN> to accept
  OR type alternative name (7 characters).

* Enter the drive No. when asked.
  The header data are read and displayed.

* Do you wish to display all the fix data?:-
  EITHER <RTN> to skip this
  OR <Y> to display data.

4.2.16 RESET SWITCHES (Option 14)

Switches for diagnostics, sound, printer and remote can be reset.

* Do you wish to reset switches?:-
  EITHER <RTN> if you do
  OR <N> to finish.

* Do you wish to enable diagnostics?:-
  EITHER <RTN> if you do not
  OR <Y> to print out variables at various stages in the program for debugging.

* Do you wish to have audible 'BLEEPS' ?:-
  EITHER <Y> if you do
  OR <RTN> if you do not.

* Do you require print-out?:-
  EITHER <RTN> if you do not
  OR <Y> if you do. Ensure printer is connected and on-line then hit <RTN>. Printer responds 'READY'.

* Do you wish to have the option to fix a remote transponder?:-
  EITHER <RTN> if you do not
  OR <Y> if you do. Then for each fix, after entering ship data you will have to choose whether to enter remote data.

* Finally the values of all the switch parameters are stored in the 'trans' file.

4.2.17 STOP (Option 15)

* Stops the program.

<Beacon on ACNAV View disk>

CHAPTER 5 PROGRAM BEACON

5.1 INTRODUCTION

This program is found on the same disk as NAVPLT. It may be run on its own or called up from NAVPLT. In the latter case once BEACON has served its purpose it will be necessary to reload NAVPLT to continue and this takes 1 minute. This is rather long to wait in an active navigation session. Ideally the functions of BEACON would be incorporated in NAVPLT itself but, as mentioned previously, NAVPLT presently occupies most of the available memory.

The function of BEACON is to write or edit the Beacon Data File (BDF). The BDF is normally written on the same disk channel as the Fix Data Files (FDF's). The structure of the BDF is shown in Appendix 7. The header section contains information on: the BDF filename (6 characters), the effective sound velocity, the disk drive number, the number of beacons, the latitude and longitude of the origin of the local metric grid, and the initial limits for NAVPLT screen plots in local metres. After this come the parameters of the first beacon in the order of colour, transmit/receive delay time in ms, and the x, y and z coordinates in metres, relative to the local grid origin. (Note that the origin is situated at mean sea level, x increases towards the east, y towards the north and z vertically upwards, hence all underwater positions have negative z coordinates).

5.2 OPERATION

The operation sequence of BEACON is displayed as a flow diagram in Figure 6.

Initially the user is asked to enter the drive number of the BDF. Then he must give either the filename of an existing BDF or choose to create a new file. With the former option the BDF data is read from the disk and displayed on the screen and then a menu is displayed. With the latter option the editing procedure is initiated.

BEACON MENU

The menu presents the following alternative choices:

i) READ to read in all the data from another BDF and overwrite all data currently in memory.

ii) EDIT to display, one by one, all the parameters from the header block. After each parameter is displayed the user either accepts it or types an alternative value or character string. When the header block has been completed all the current data is listed. Then the user is asked to select a beacon to edit its parameters or else to return to the main menu. After editing the parameters for the chosen beacon the same choice is given between editing another beacon and returning to the menu.

iii) WRITE to write the current BDF data to the BDF on disk, overwriting any exisitng file with the same BDF name. After writing, the data is read in again to check the file was written correctly and then the data are displayed on the screen before returning to the menu.

iv) RETURN to NAVPLT The user is prompted to ensure that the NAVPLT program disk is inserted in Drive 0. The computer then loads the program NAVPLT which takes about 1 minute.

<NAVVY1 on ACNAV View disk>

CHAPTER 6 PROGRAM NAVVY

6.1 INTRODUCTION

This program was developed from the program NAV5 by I. Rouse (IOS). NAV5 performed the calculations required to reiteratively fix a set of transponders, given a set of ship fix positions which had to be typed in. NAVVY uses the same calculations but the program is more sophisticated in that the basic data set can be edited and saved in a disk file. Also the program is written in the same style as NAVPLT and BEACON with many common variables and arrays.

6.2 OPERATING INSTRUCTIONS

The basic flow diagram is illustrated in Figure 7. The program sequence is as follows.

* Constants are assigned default values: local metric grid origin X & Y coordinates; No. of beacons, mean harmonic sound velocity and transponder receive/transmit delay time.

* PROCdatip is called. The user decides whether the input ship positions are adjustable or fixed. Data are read from a previously-created diskfile or entered at the keyboard (the User chooses).

* If the user chooses to enter all new data then PROCdatedit is called. This option is chosen when absolute geographical positions are provided by an independent navigation system (e.g. Transit satellites, Loran C, GPS). The resulting beacon net will then be fixed absolutely in space but the quality of fit of the calculated beacon positions to the measured ship positions and ranges will be less good than if the ship positions are considered adjustable.

* PROCdatedit is then called. The user may skip this editing function. Otherwise the current values of all variables are displayed in turn and the user decides whether to accept them or enter new values.

The sequence is: Number of beacons; X,Y,Z coordinates and Lat. and Long. of each beacon in turn (new positions can be entered either as X,Y or Lat and Long but z must be entered in m.)

* Next the data are listed (by PROCdatlist), firstly the beacon positions are displayed then the fixes in the

format: number, day, time, ranges to transponders. Ranges may be input in one of 3 alternative forms: i) 2WT in s. including delay (as read from the EPC); ii) calculated slant range in m or iii) calculated horizontal range in m. ii) and iii) are derived from NAVPLT output. (N.B. no allowance for refraction is made; see 7.1).

* PROCdatop is called next. The user decides whether to save the edited data to a diskfile.

* PROCallships is called next. The user selects the iteration step size G; the maximum number of iterations for each ship position and the maximum acceptable error. The values of these variables affect the program speed and the accuracy of the results. Default values are recommended but other values may be chosen once experience is gained.

The procedure considers each ship fix in turn. PROCshipfix is called for each fix and then the fix is either accepted or rejected. The number of acceptable fixes is totalled.

* PROCshipfix considers a single ship position using the current beacon positions. Firstly the deviation between the predicted and the measured range are summed for all available ranges. If the ship positions are not to be adjusted then the procedure ends here.

Otherwise the ship position is adjusted by an arbitrary small amount and the new error is calculated. The ship position is then adjusted by an amount which corresponds to the gradient of the error and the new error is calculated.

This procedure is repeated until either the number of iterations reaches the maximum allowed or the position adjustment between iterations decreases below 0.1 m in both X and Y. The program then returns to PROCallships as described above.

* If the number of acceptable fixes is too small the program returns to the beginning of data input (see PROCdatip above). Otherwise PROCiternet is called. This procedure uses the current set of acceptable ship positions. It reiteratively calls the procedure PROCtranfix, progresively improving the fit (reducing the error) of the beacon net to the ship fixes until either the adjustments of the X,Y and Z coordinate adjustments between iterations of all beacons all decrease below 0.1 m or the maximum number of iterations is reached.
  Then all the ship fixes are reconsidered in turn by calling PROCshipfix and calculating their individual errors (and adjusting their positions if allowed).

* PROCtranfix uses all the current accpetable ship positions to adjust each beacon position in turn reiteratively using the same error gradient method described for PROCshipfix. Note that the measured baseline distances are not used to constrain the positions. The predicted baseline distances are calculated at the end of the procedure for comparison with the measured values.

* Then the data is listed on the screen (PROCdatlist) and the options are given to reedit the data and resave it all to the disk (PROCdatedit and PROCdatop). The program then stops.

* Other procedures include:-

PROCdegmet which converts positions from Lat & Long to local metric coordinates;

PROCmetdeg which carries out the reverse process.

PROCshipfix2 is a spare procedure which chooses the best ship position in a different way to PROCshipfix. It does this by setting up a grid of points 4 km square at 50 m intervals and moving the ship position to every point and calculating the residual error and finally selecting the position with the minum error. It is very slow and not as accurate but it may be useful to check that the gradient method does not get stuck in a local secondary minimum.

<T9 on ACNAV View disk>

CHAPTER 7 FUTURE IMPROVEMENTS

7.1 RAY REFRACTION.

At present NAVPLT assumes that all acoustic rays travel along straight lines. It assumes that the sonic velocity of the water is constant and the value used is the harmonic mean sound velocity which can be derived by integrating the velocity-depth profile (see Matthews/ Carter tables).

Such assumptions produce reasonable results for ship navigation at ranges of up to 20 km from near-seabed beacons in water depths of 3 km (Rouse, 1987). At greater ranges the errors induced by raypath refraction become more appreciable (several 10's of metres). Also, the accuracy of positioning of remote devices may be degraded when the remote is at shallow depths and there is a strong velocity gradient.

It is possible to avoid these errors by taking account of refraction. The simplest way to do this is to model the path of the rays through a known velocity-depth profile assuming a horizontally-layered water column and dividing this up into layers of constant velocity gradient. The only known way to do this is to reiteratively adjust the angle at which a ray leaves the transmitter until its path reaches the known depth of the receiver at the indicated travel time. This process is however rather slow (c. 11 s per range even on an IBM PC). Furthermore it does not completely solve the problem when a remote receiver is at an unknown depth.

An alternative method is described by Rouse (1987) as proposed by Daintith (1970) which is to apply a correction factor CF to the mean sonic velocity based on the depths of the source and receiver and the normalised variance of the velocity profile between the two. For ship navigation such a correction could be incorporated into the programs fairly simply by extending the program BEACON to:-

i) read from the keyboard the known velocity-depth profile;
ii) calculate CF for each beacon depth;
iii) store the correction factors in the BDF for reading into NAVPLT to be used in the travel time to distance conversions.

For a remote transponder however the procedure would not be so simple since the remote depth can vary. Three alternatives present themselves. In each one CFs for each

beacon must be determined for the estimated remote depth and then an initial fix position is determined. If the calculated depth is very different from the estimated depth then new CF factors must be determined and the fix recalculated. This process may be repeated until convergence occurs. The 3 alternatives are:-

i) The entire velocity-depth profile data could be stored in the BDF using BEACON for keyboard entry and then a new procedure in NAVPLT could calculate the value of CF according to the estimated remote depth.

ii) The correction factors could be calculated over the whole depth range at regular intervals in BEACON and stored in NAVPLT as a look up table. At 100m intervals in 6000m water depth this would produce 60 values per beacon. Remember that values for the ship:remote path would also be needed. It is doubtful whether sufficient room is available in memory for such a volume of data when NAVPLT is loaded. It would therefore be necessary to store the values in a diskfile, the BDF perhaps, or a random access file for speedier access.

iii) Alternatively if the remote is likely to be used within a fairly narrow depth range the correction factors need only be worked out for the middle of the range. Note that a seperate correction factor for the ship:remote path will be required.

7.2 PLOTTING.

At the moment plotting is only possible on the screen as no hard plotter was available during program development. It would not be difficult however to use the procedure PROCreadFfix to read in old fixes from an FDF and plot these out.

Provision for plotting range circles, beacon positions etc. during an ACNAV session could be made by defining a flag, set by the user, to produce hard copy plot via statements of the form: IF PLOTFLAG=1 THEN PLOT..... .

7.3 MAINFRAME LINK.

Another possibility that may be considered is to set up a data link between the BBC and the ship's mainframe ABC computer system. A short BBC program using the procedure PROCreadlfix could read the fix data in from the FDF and send it to the mainframe. It would then be possible to log and plot ACNAV data in the same way as other data logged by the ABC system on RVS ships. The main advantage would be automatic large-scale plotting and comparison with other navigation methods.

Slightly more complicated would be the transmission of data to the ABC whilst online. This would be done once each fix has been saved to floppy disk. Advantages are online plotting of ship position on the ABC-driven VDU used by the helmsman.

These improvements would most easilly be accomplished in collaboration with an RVS computer engineer on ship. Note that converting the program to run on the ABC would not be worthwhile because the ABC system is too slow. Conversion to an IBM PC FORTRAN program would speed up operation but the work involved would be at least 2 man months. I. Rouse (personal communication) is evaluating a method of using the Lowenstein gradient method to calculate online fixes automatically. A fast computer is required for this.

<T10 on ACNAV View disk>

CHAPTER 8 PROGRAMMING HINTS

These brief notes are intended for anybody wishing to develop the ACNAV programs.

* Be very sure that the new code will be a significant improvement as it is easy to spend a lot of time editing for little reward.

* When editing NAVPLT save the program frequently to disk. Sometimes during extended edits the computer comes up with 'Bad program' and the entire program is lost from memory.

* Beware renumbering NAVPLT, it may grow so large that renumbering fails.

* Try to have just one ENDPROC statement per procedure to facilitate debugging and path tracing.

* Keep the flow diagrams, variable lists and file structure maps updated. This facilitates future editing and minimises ungainly structuring and convoluted paths.

* If in doubt keep things simple.

* KEEP BACKUP COPIES OF ALL PROGRAMS AND DATAFILES.

<Tll on ACNAV View disk>

REFERENCES

DAINTITH M.J., 1970. A simple method of correction for refraction in underwater navigation systems. British Acoustical Society, London 6 pp.

MATTHEWS D.J. Tables of the velocity of sound in pore water and in sea water. Admiralty Hydrographic Department, London.

PHILLIPS G.R.J., 1980. The IOS Acoustic Command and Monitoring System. Part 1. Operating Principles. Institute of Oceanographic Sciences Report No. 96, 9pp. Unpublished Manuscript.

ROUSE I., 1987, The IOS 5kHz acoustic navigation system. Unpublished IOS Internal Report.

<Alproc on ACNAV View disk>

APPENDIX 1 KEY TO ABBREVIATIONS USED.

ABC Mainframe computer system used on NERC ships.

ACNAV Acoustic Navigation System.

BDF Beacon datafile. Created using program BEACON.

BEACON Program for creating and editing BDF's.

EPC Analogue graphic recorder used to measure acoustic travel times.

FDF Fix datafile. Used to store raw and processed fix data.

GPS Satellite-based Global Positioning System.

IP Intersection Point of two range circles on a horizontal plane.

NAVVY Program for determining the relative geometry and absolute position of an array of fixed transponders (beacons).

NAVPLT Main program for processing ACNAV data.

RVS Research Vessel Services, a branch of NERC.

trans 'Housekeeping' disk file used for storage of current parameter values.

<A2proc on ACNAV view disk>

APPENDIX 2 NAVPLT PROCEDURES AND FUNCTIONS

PART 1: List of Procedures.

PART 2: Detailed list of procedures.

The procedures are listed in alphabetical order, each with their first and last line numbers, a brief description of their purpose and a list of other procedures and functions which it calls directly. Any arguments required by the procedure are listed in brackets. The prefix PROC- is omitted from procedure names. Also listed here are functions and programs which are indicated by a '*'.

ban(x0,x9,y0,y9,ban$,NCOL) 11380-11420
Draws an NCOL coloured box on screen with text given by ban$.
Calls: None.

*BEACON (A Seperate program)
User creates or edits a BDF file.
Calls: various procedures and NAVPLT

beepin 3260-3640
Reads data from BDF. Accesses BEACON for BDF editing.
Calls: qon; qof; X; diskIOerror; metdeg; error; BEACON

calc 9560-10220
Calls Procs for calculating circle IPs, cokhat fix, matrix fix, chooses best result.
Calls: ints; cokhat; MATFIX.

circ(XC,YC,R) 14320-14380
Screen plots a circle.
Calls: scal.

cokhat 15320-15860
Selects 3 co-planar IPs with smallest deviation from centroid. Calculates centroid and deviation.
Calls: None.

comp 7800-8520
Calcs heading & speed from last fix, predicts ranges.
Calls: ban.

datdri 11460-11520
Sets drive string for writing to disk e.g. ":1.".
Calls: qon; qof.

datin 4020-4540
User enters raw ship data for a new fix.
Calls: qon; qof; relabel; stom; reset; X; remdatip;

datwrit 12440-12900
Checks nominated FDF or asks for alternative then writes data for 1 fix to it. Does a "parity" check. Calls Read-after-Write check.
Calls: diskIOerror; datdri; qon; qof; new; error; rarchek.

degmet (xd,yd) 18720-18780
Converts from Lat and Long to local metric coords.
Calls: None.

diskIOerror 11700-11760
Traps disk errors. Disables 'escape's while writing to disk which could corrupt diskfiles.
Calls: qon; qof; error.

error 11560-11660
Traps non-disk errors, reports, alarms and may call execution stop.
Calls: stop.

filemake 11800-12000
Catalogs data disk, options to call file creation.
Calls: qon; qof; datdri; newtrans; BEACON; new; hdrd.

fix 5620-7760
Calls Screen Plot, Pre-Fix Menu, Fix-Type Menu, calls compare & predict,Post-Fix Menu, calls Save routine.
Calls: qon; qof; plot; interp; X; scal; latlon; ban; comp; calc; save.

*FNDP(X) 17000-17040
Truncates to 2 decimal places.
Calls: None.

grid (XDIV,YDIV) 14620-14780
Draws & annotates Km grid lines for screen plot.
Calls: scal.

hdrd 8880-9160
Reads header of FDF and lists to screen or printer.
Calls: qon; qof; X; diskIOerror; readlfix; error.

hrdplt 15220-15280
For plotting to plotter. Not operational yet.
Calls: qon; qof.

interp 18800-19020
Interprets CODE form old format FDF to determine fixtype.
Calls: None.

ints(XA,YA,RA,XB,YB,RB) 10260-10560
Calculates coords of the 2 IPs of two circles of specified centres and radii.
Calls: short; long; root.

latlon 5360-5580
Converts metric coords to Lat & Lon and displays them.
Calls: None.

long 10680-10760
Calculates delta coords of 2 coincident IPs for non-overlapping circles.
Calls: None.

MATFIX 17100-17660
Calculates matrix solution for fix position.
Calls: matset.

matset 17700-17980
Selects ranges for matrix fix.
Calls: qon; qof; plot.

menu 2760-3140
Main menu.
Calls:-beepin; reset-datin-calc-fix; plot; fix; calc; rename; hdrd-transin; qof; beepin; new; read; switch; save; reproce; filemake; qon.

metdeg (XP,YP) 16880-16960
Converts local metric coords to Long. and Lat.
Calls: None.

new 12040-12320
Creates a new, empty FDF file on data disk.
Calls: qon; qof; X; diskIOerror; datdri; error.

newtrans 16660-16840
Creates an empty 'trans' file.
Calls: diskIOerror; datdri; error; transin.

old 12360-12400
Nominates an existing FDF to recieve saved fix data.
Calls: qon; qof; X.

olfix 14820-15180
Reads from FDF and screen plots a number of old fixes.
Calls: qon; qof; olread; scal;dsikIOerror; datdri; prevfile; degmet; error.

plot 13200-14280
Plot window menu. Plots screen grid and beacons and fixed ship/remote. Plots range circles, IPs, matrix and cokhat fixes.
Calls: qon; qof; stop; ban; set; grid; scal; olfix; circ.

prevfile 16300-16360
Opens the FDF which preceedes current FDF for reading.
Calls: None.

qof 2720
Undoes blue highlight.
Calls: None.

qon 2680
Produces blue highlight and a 'beep' for user attention.
Calls: None.

rarchek 12940-13080
Performs a Read-after-Write check on FDF data for last fix.
Calls: diskIOerror; datdri; error.

readFfix 19400-19600
Reads data for a single fix into non-current fix variables.
Calls: degmet.

readlfix(ICHN,NFIX) 9200-9520
Reads data for one old fix from FDF file into current fix variables.
Calls: metdeg.

relabel 18580-18680
Sets screen flags to default settings.
Calls: ban.

remdatip 4580-5260
User enters raw range data for remote device.
Calls: qon; qof; X; stom.

renam 3180-3220
User nominates a different FDF file for saved data.
Calls: qon; qof; transout.

reproce 18080-18540
Reads fix data from old FDF, reprocesses and writes to new FDF.
Calls: hdrd; qon; qof; stop; reset; readlfix; save; relabel; calc; fix; X; interp; save.

reset 3680-3980
Resets data & control variables to null/default values.
Calls: none.

root 10600-10640
Calculates delta coords of 2 IPs of 2 intersecting circles.
Calls: None.

save 8620-8840
Confirms or reselects FDF. Calls write procedure.
Calls: qon; qof; new; old; datwrit; transout; ban.

scal(X,Y) 14540-14580
Produces scaled plot coords XP & YP.
Calls: None.

set(YMIN,YMAX,XMIN,XMAX) 14420-14500
Defines scaling factors for screen plot.
Calls: None.

short 10800-10840
Calculates delta coords of 2 IPs of 2 completely overlapping circles.
Calls: None.

start 2120-2640
Sets windows, key functions, arrays. Called once.
Calls: error; ban; qon; qof; transin.

stom 5300-5320
Converts slant 2WT in s. to horiz. range in m. at Z=ZZ
Calls: None.

stop 19040-19100
Stops execution and tells user how to restart.
Calls: qon; qof.

switch 16400-16620
User selects options for diagnostics, sound, print-out and remote device.
Calls: qon; qof; transin; transout.

transin 10880-11120
Reads data from 'trans' datafile.
Calls: diskIOerror; datdri; error.

transout 11160-11340
Writes data to 'trans' datafile.
Calls: diskIOerror; datdri; error.

X 18020-18040
Audiovisual alarm to wake user up.
Calls: qof.

<app3 on ACNAV Viewdisk>

APPENDIX 3 NAVPLT VARIABLES AND ARRAYS.

Variables and arrays in NAVPLT are listed. The order is generally alphabetical but similar variables such as XMEAN, YMEAN and ZMEAN are listed together under XMEAN for example. The procedures in which the variables occur are listed in brackets. Frequently occurring variables such as loop indices I, J etc. are omitted.
N.B. The character 'Δ' means 'raised to the power of'.

@% (X) BBC BASIC print format parameter.

A (reset, cokhat)
Flag used to signify whether PROCauto has been called.

A1, A2 (MATFIX, matset)
A1=2*(X2-X1), A2=2*(X3-X1)

ALFA (ints, root, MATFIX)
=arccos(ARG); OR =(R1Δ2 - R2Δ2)+(X2Δ2+Y2Δ2+Z2Δ2)
-(X1Δ2+Y1Δ2+Z1Δ2) in
MATFIX.

ARG (ints)
=(RAΔ2 +CΔ2 -RBΔ2)/(2*RA*C)

AUTPROC$ (start, reset, datin, fix, comp, save, plot, olfix, cokhat, matset, reproce)
Flag indicating whether automatic reprocessing is to take place or not.

B$ (fix)
String indicating best fix method.

B1, B2 (MATFIX, matset)
B1=2*(Y2-Y1), B2=2*(Y3-Y1)

BAN$ (start)
String variable to be displayed as banner on graphics screen.

BCN1, BCN2, BCN3, BCN4 (MATFIX, matset)
BCN1=BCNO(1) etc.

BCN5, BCN6, BCN7 (matset)
See BCN1

BCNO() (start matset)
Numbers of beacons acceptable for matrix fixing.

BECOL$() (start, beepin, datin, remdatip, comp)
String array containing beacon colour string.

BEEP$ (start, beepin, transin, transout)
Flag indicating whether audible 'beep' is to be sounded.

BEEPIN$ (start, beepin)
Flag indicating whether BDF data has been read in yet.

BETA (MATFIX)
Algebraic shorthand variable.

BN (matset)
Loop index.

bn$ (various: comp)
Banner string for screen display by PROCban

BRNG (comp)
Bearing in degrees of device track from previous fix to current fix.

C (ints, long, short)
=SQR(DXΔ2 +DYΔ2)

C$ (fix)
String representation of IP number with leading '0' if IP <10.

C1, C2, C3 (MATFIX)
Algebraic shorthand variables.

CHCODE (fix, interp)
Numeric representation of CHCODE$.

CHCODE$ (fix)
String code representing the 3 IP's used for cokhat fix e.g. '010511'.

CHEK (start, calc, transin, transout, rarchek, newtrans)
Flag to enable/disable diagnostic printout. Replaced by CHEK$.

CHEK$ (switch)
Flag to enable/disable diagnostic printout.

CLR$ (start)
Name of procedure from which current procedure was called. Used for path tracing.

CODE (reset, fix, interp)
Numeric code indicating fix type. See Appendix 10.

CODE$ (fix, interp)
String descriptor of fix type. See Appendix 10.

D% (start, save, datdri, filemake, filemake, datwrit)
Stores current disk drive No. for data output.

D1, D2, D3 (MATFIX)
Algebraic shorthand variables.

DATIN$ (start, reset, datin, fix, save, calc, plot, reproce)
Flag indicating whether raw fix data has been entered.

DAY$ (readlfix, relabel)
String representing Julian day number.

DAY0 (hdrd, datwrit, hrdplt)
Julian day number of first fix on FDF.

DAY9 (hrdplt) *Replace by DAYEND*
Julian day number of last fix on FDF.

DAYEND (hdrd, datwrit)
Julian day number of last fix on FDF.

DDX (plot)
Distance in Km from last ship fix to boundaries of new screen plot area.

DET (MATFIX)
Determinant of matrix.

DH (comp)
Horizontal distance between current and previous fix
=SQR(DH2)

DH2 (comp)
=DXΔ2 + DYΔ2. See DH.

DHRB (comp)
Horizontal distance from remote to beacon.

DHSB (comp)
Horizontal distance from ship to beacon.

DHSR (comp)
Horizontal distance from ship to remote.

DR$ (start, comp, transin, transout, datdri, new, datwrit, rarchek, olfix, prevfile, newtrans)
String representation of disk drive No. for data output.

DRINO (beepin)
Disk drive No. for data O/P.

DRINO$ (filemake)
String representation of DRINO.

DRIP$ (beepin, hdrd, readlfix)
String representation of disk drive No. for data I/P.

DSRB (comp)
Slant distance from remote to beacon in m.

DSSB (comp)
Slant distance from ship to beacon in m.

DSSR (comp)
Slant distance from ship to remote in m.

DTIME (comp)
Time difference between current and previous fixes in minutes.

DTYPE$ (hdrd, datwrit)
Obsolete variable indicating data type (raw or processed).

DUM (start, fix, reproce)
Variable for storing menu selection number from keyboard.

DUM$ (menu, beepin, datin, remdatip, fix, save, hdrd, filemake, datwrit, plot, switch)
String equivalent of DUM.

DX (comp, ints)
Difference in X coordinates in metres.

DX1, DX2 (ints, root, long, short)
Differences in X coordinates for a pair of IP's from common point.

DY (comp, ints)
Y-axis equivalent of DX.

DY1, DY2 (ints, root, long, short)
As for DX1, DX2 but along Y axis.

DYTM% (readFfix)
Julian day number and time: e.g. 3651632.

DZ (comp)
Difference in Z coordinates in metres.

ENDPTR (comp, hdrd, datwrit, olfix, prevfile, readFfix)
Number of the last byte in a random access file occupied by real data.

ERL (error)
Error line number.

ERR (error)
BBC BASIC Error code number.

FDAY (comp, readFfix)
Julian day number of temporary fix.

FFIX (olfix, readFfix)
Number of temporary fix in FDF.

FILEN (start, filemake, new, datwrit)
Number of fix spaces in FDF file.

FRXDEG, FRYDEG (readFfix)
As XDEG & YDEG but for temporary fix.

FRXMEAN, FRYMEAN, FRZMEAN (comp, olfix, readFfix)
As RXMEAN, RYMENA & RZMEAN but for temporary fix.

FTIME (comp, readFfix)
Time in HHMM format for temporary fix.

FXDEG, FYDEG, FZDEG (readFfix)
As XDEG, YDEG, ZDEG but for temporary fix.

FXMEAN, FYMEAN, FZMEAN (comp, olfix, readFfix)
As XMEAN, YMEAN & ZMEAN but for temporary fix.

FXTYP (reset, fix, interp)
Numeric code for fix type.

FXTYP$ (fix)
String description of fix type.

GAM (ints, root, long, short)
=GAM0 + N*PI where N depends on signs of DX & DY.

GAM0 (ints)
=arctan (DY/DX) or =PI/2 if DX=0.

GAMA (MATFIX)
Algebraic shorthand variable.

HDLEN (hdrd, readlfix, new, datwrit)
Number of bytes in FDF header section. Currently = 3*9 + 7*6, i.e. 3 strings with 7 characters plus 7 numeric values. See Appendix 6.

HM0 (comp)
=INT(OLTIME/100)

HMN (comp)
=INT(NTIME/100)

ICHN (beepin, hdrd, readlfix, transin, reproce)
Number of channel for input of data from disk

IPCALC$ (plot)
Flag indicating whether IP's have been calculated yet for current data.

IPFILE$ (hdrd, readlfix, reproce)
Name of input FDF file.

IX, IY (fix)
Shorthand for XI(VI(I%)), YI(VI(I%)): X & Y coordinates of IP number VI(I%).

IX%, IY% (grid)
Integer kilometer values for labelling plot axes.

MCODE (fix, matset, interp)
Numeric code indicating which ranges were used in Matrix fix.

MCODE$ (MATFIX, matset)
String version of MCODE.

MDUM (menu)
Parameter used for keyboard selection of menu item.

MEANX, MEANY (latlon, fix, plot)
The metric X & Y coordinates produced by the chosen fix method.

MEANZ (fix)
See MEANX.

MMSD (cokhat)
Mean of roots of squares of deviations of 3 ccokhat IP's from mean position.

MN (comp)
Minutes of current fix time.

MNNRANS (MATFIX)
Minimum number of ranges required for a matrix fix. =3.

MNNRNS (matset)
Minimum number of ranges required to matrix fix particular device ship=3, remote =4.

MNRS (start, reset, remdatip, calc, plot, matset)
Maximum allowable number of transponder ranges per device.

MO (comp)
Minutes of previous fix time.

MULT (latlon, metdeg)
Multiplication factor for converting degrees longitude into distances in km, based on cos of latitude =SGN(YDEG) * COS(RAD(YDEG)).

NBCNS (start, beepin, datin, remdatip, comp, calc, transin, transout, plot)
Number of existing beacons, from BDF.

NCOL (start, ban, plot)
Numeric colour code as defined in BBC BASIC.

NDAY (start, datin, comp, readlfix, transin, transout, datwrit)
Julian day number.

NDAY$ (datin)
String version of NDAY.

NF (olfix)
Loop index of fix number, numbered backwards from current fix.

NFBEOF (comp, olfix, prevfile, readFfix)
Number of fixes before end of FDF file from current pointer position.

NFIX (readlfix, reproce)
Number of fix in FDF.

NFIX1, NFIX9 (reproce)
Numbers in FDF of first and last fixes of a batch to be reprocessed.

NFIXES (comp, hdrd, new, datwrit, olfix)
Number of fixes in FDF.

NGIPS (calc, plot, cokhat)
Number of good IP's in current fix geometry.

NGRANS (fix, calc, cokhat)
Number of good ranges for current fix.

NH (comp)
Hour of current fix time.

NI (reset, fix)
Counter for number of valid IP's.

NI% (plot)
Loop index representing number of IP.

NI2% (plot)
Number of second of pair of IP's, NI% being the first. =NI%+1.

NINT1, NINT2, NINT3 (cokhat)
The 3 IP's used to calculate cokhat fix of either device.

NM (comp)
Minute of current fix time.

NOPFIX (datwrit)
Number of new fix written to FDF. =old NFIXES +1.

NPF (olfix, hrdplt)
Number of previous fixes required to be plot.

NRANS (start, datin, remdatip, transin, transout, cokhat, MATFIX, matset)
Number of ranges input from keyboard, or from FDF if reprocessing.

NRNGS (calc)
Number of possible ranges, for ship=NBCNS, for remote =NBCNS+1.

NT (comp)
Current fix time in minutes since start of year.

NTIME (start, datin, comp, readlfix, transin, transout, datwrit)
Time of current fix in HHMM format.

NTIME$ (datin, readlfix, relabel)
String version of NTIME.

OCHN (comp, transout, new, datwrit, rarchek, olfix, prevfile, newtrans, readFfix)
Number of output channel for writing to FDF.

OF$ (start, menu, comp, save, hdrd, transin, transout, new, old, datwrit, rarchek, olfix, reproce)
Name of FDF file for data output.

OH (comp)
Hour of previous fix time.

OLDAY, OLTIME (start, comp, transin, transout)
Julian day number and time in HHMM format of previous fix.

OM (comp)
Minute of previous fix time.

OSOR$ (olfix)
Flag indicating whether ship data or remote data are selected for previous fix.

OT (comp)
Previous fix time in minutes since start of year.

OXMEAN, OYMEAN, OZMEAN (start, comp, transin, transout)
TO BE REPLACED BY FXMEAN, FYMEAN & FZMEAN in all PROCS

P% (fix)
Loop index.

PIPS (calc)
Number of possible IP's, based on number of good ranges.

PNTR (rarchek, readFfix)
Number of byte at which file pointer rests in random access file.

PREVOF$ (start, hdrd, transin, transout, filemake, new, datwrit)
Name of FDF preceeding current FDF.

PRNTR (start, beepin, datin, remdatip, latlon, fix, comp, save, hdrd, readlfix, calc, transin, transout, rarchek, cokhat, switch, newtrans, MATFIX, interp)
Flag (=1 or 0) indicating whether print output is sent to the printer aswell as to the screen.

PTRINT (fix, comp, hdrd, readlfix, new, datwrit, olfix, prevfile, readFfix)
Number of bytes occupied by data for each fix in an FDF, the interval which the file pointer must move between equivalent positions in successive fixes>

Q$ (remdatip, fix, hdrd, diskIOerror, new, datwrit, ACPLOT, switch, matset, reproce)
Dummy variable used for keyboard selection of options.

QQ$ (hdrd)
As Q$.

QQREM (start, datin, transout, olfix, switch, newtrans, reproce, relabel)
Flag indicating whether remote option is available in current program configuration. Reset in PROCswitch.

QREM$ (start, datin, remdatip, fix, save, calc, ACPLOT, switch, reproce)
Flag indicating whether remote option was selected for current fix.

R (circ)
Radius of range circle in horizontal metres.

R(I) (start, reset, datin, remdatip, readlfix, calc, transin, transout, datwrit, plot, MATFIX, matset)
I= 1 to 6: Ranges between ship and beacons in metres or seconds.
I= 7 : Range between ship and remote.

R1, R2, R3, R4 (MATFIX)
Algebraic shorthand variables.

RA, RB (ints, root, long, short)
Horizontal range circle radii from the 2 beacons under consideration.

RBDSlT (remdatip)
One way acoustic travel time from remote to beacon to ship, including beacon delay.

RBDSlT() (start, remdatip)
Array of RBDSlT values for each beacon.

RBSTFX$ (fix, calc)
String descriptor of recommended best fix method for remote.

RCODE (start, reset, fix, save, readlfix, transin, transout, datwrit, plot)
Code indicating remote fix type.

RCODE$ (readlfix, matset)
Descriptor of remote fix type.

RDl, RD2, RD3, RD4 (MATFIX)
Algebraic shorthand variables.

RDET (MATFIX)
Determinant of matrix for remote fix.

REPROC$ (start, fix, readlfix)
Flag indicating whether current data is to be reprocessed.

RFIX$ (reset, fix)
Flag indicating whether remote has been fixed yet.

RFXTYP (reset, fix, save, readlfix, datwrit)
Code indicating remote fix type.

RFXTYP$ (reset, fix, save)
Descriptor of remote fix type.

RINTl, RINT2, RINT3 (start, reset, fix, transin, transout, cokhat)
The three IP's chosen to fix the remote by cokhat method.

RIP() (start) *PRESENTLY REDUNDANT*
Array containg numbers of remote IP's.

RIPCAL$ (reset, fix, calc, plot)
Flag indicating whether remote IP's have been calculated yet.

RM (datin)
Range in metres.

RNGIPS (calc, plot)
Number of good IP's for remote data.

RNRANS (remdatip)
Number of ranges for remote fix.

ROXMEAN, ROYMEAN, ROZMEAN (start, comp, transin, transout)
X, Y & Z metric coordinates for previous remote fix.

RR (matset)
Algebraic shorthand variable.

RR(I) (start, reset, remdatip, readlfix, calc, transin, transout, datwrit,plot, MATFIX, matset)
I= 1 to 6: Remote range to beacon in metres or seconds.
I=7 : Remote range to ship.

RS (datin, remdatip, stom)
Input range, units either seconds or metres.

RS$ (datin, remdatip)
Keyboard input parameter in response to prompt for range input.

RSDAUT (calc)
Standard deviation of automatic fix for remote.

RSDCOK (calc, plot, cokhat)
Standard deviation of cokhat fix for remote.

RSDMAT (fix, calc, MATFIX) Standard deviation of matrix fix for remote.

RXAUT, RYAUT, RZAUT (calc)
X, Y & Z coordinates of automatic remote fix.

RXCOK, RYCOK (calc, plot, cokhat)
X & Y coordinates of cokhat remote fix.

RXDEG, RYDEG (reset, fix, readlfix, datwrit)
Longitude and latitude of remote fix in degrees.

RXI(), RYI() (start, reset, fix, calc, plot, cokhat)
X & Y coordinates of remote IP.

RXMAT, RYMAT, RZMAT (fix, calc, plot, MATFIX)
X, Y & Z coordinates of remote matrix fix.

RXMEAN, RYMEAN (start, reset, fix, comp, save, transin, transout, olfix)
X & Y coordinates of remote fix.

RXSD, RYSD, RZSD (reset, fix, save, readlfix, transin, datwrit)
X, Y & Z components of standard deviation of remote fix.

RZ (datin, remdatip, stom)
Horizontal range in metres.

RZCOK (fix, calc, cokhat)
Z coordinate of cokhat remote fix (=RZMEAN if auto reprocessing, =ZRIP if normal online processing).

RZMEAN (reset, fix, comp, save, readlfix, datwrit, cokhat, reproce)
Z coordinate of remote fix.

SBSTFX$ (fix, calc)
Descriptor of best fix method selected automatically for ship.

SCODE (start, reset, fix, readlfix, transin, transout, datwrit, plot)
Code indicating ship fix type.

SCODE$ (readlfix, matset)
Descriptor of ship fix type.

SDCOK (cokhat)
Standard deviation of cokhat fix.

SDX, SDY, SDZ (fix)
X, Y and Z components of standard deviation of fix.

SECMET$ (datin)
Flag indicating whether range data received from keyboard is in metres or in seconds.

SEP$ (plot)
Spacing string.

SFIX$ (reset, fix)
Flag indicating whether ship position has been fixed yet.

SFXTYP (reset, fix, save, readlfix, datwrit)
Code indicating ship fix type.

SFXTYP$ (reset, fix, save)
Descriptor of ship fix type.

SHINT1, SHINT2, SHINT3 (start, reset, fix, transin, transout, cokhat)
The three IP's chosen to fix the remote by cokhat method.

SIPCAL$ (reset, calc, plot)
Flag indicating whether ship IP's have been calculated yet.

SND (start and various)
Flag indicating whether beeps are to be made.

SNGIPS (calc, plot)
Number of good IPs for ship fix.

SOR$ (reset, fix, comp, calc, plot, olfix, cokhat, matset)
Flag indicating whether subsequent operations apply to ship data or to remote data.

SORR$ (calc, cokhat, MATFIX, matset)
Similar to SOR$.

SSDAUT (calc, plot)
Standard deviation of automatically-selected ship fix.

SSDCOK (calc, plot, cokhat)
Standard deviation of cokhat SHIP fix.

SSDMAT (fix, calc, MATFIX)
Standard deviation of matrix ship fix.

SUM2X, SUM2Y (fix)
Sum of squares of X and Y differences from means.

SUMX, SUMY (fix)
Sum of X and Y differences from means.

SVEL (beepin, datin, remdatip, stom)
Harmonic mean sound velocity for working area, calculated from Carter's tables and stored in BDF.

SXAUT, SYAUT, SZAUT (calc)
X, Y & Z coordinates of automatically-selected ship fix.

SXCOK, SYCOK (calc, plot, cokhat)
X & Y coordinates of cokhat ship fix.

SXDEG, SYDEG (reset, fix, readlfix, datwrit)
Longitude and latitude of ship fix in degrees.

SXMAT, SYMAT (fix, calc, plot, MATFIX) X & Y coordinates of matrix ship fix.

SYM$ (plot)
Symbol used to depict object on screen plot.

SZCOK (calc, cokhat)
Z coordinate of cokhat ship fix (=0).

SZMAT (fix, calc, MATFIX)
Z coordinate of matrix ship fix.

T1RBS (comp)
One-way travel time from remote to beacon to ship, in s.

T1SB, T2SB (comp)
One and two-way travel times between ship and beacon.

TDEL (datin)
Beacon receive-transmit delay time in s.

TDEL() (start, beepin, datin, remdatip, comp)
Beacon receive-transmit delay time in s.

TEX$ (menu)
First 2 characters of OSCLI command, used to reject commands: BACKUP, COPY, FORMAT or COMPACT which would corrupt the program.

TITLE$ (beepin)
Name of BDF file.

TM0 (hdrd, datwrit, hrdplt)
Time of first fix on FDF.

TM9 (hrdplt)
Time of last fix on FDF.

TMEND (hdrd, datwrit)
Time of last fix on FDF.

VARX, VARY (fix)
Variances of X and Y.

VI() (reset, start, fix)
Numbers of Valid Intersection points.

X,Y (scal)
Metric coordinates sent to PROCscal to be conerted to screen coordinates.

X(I), Y(I), Z(I) (start, beepin, reset, fix, comp, save, calc, transin, transout, plot, MATFIX, matset)
X, Y and Z coordinates of: (I=1 to 6) beacons; (I=7) ship position; (I=8) remote position; (I=9, 10) standard deviations of ship and remote fixes; (I=11, 12) previous ship and remote positions; (I=13) initial plot limits from BDF: xmin, ymin, null; (I=14) initial plot limits from BDF: xmax, ymax, null;

x0, x2, x9, y0, y9 (COMP and other procedures)
Screen coordinates for corners of banners.

X0, Y0 (start, transin, transout, set, scal)
Screen coordinates of plot origin. Both =0.

X1, Y1, X2, Y2 (plot, MATFIX, matset)
Algebraic shorthand variables.

X1$, X2$, X3$ (rarchek)
Dummy variables for reading and checking FDF data.

X3, Y3 (MATFIX, matset)
Algebraic shorthand variables.

X4, Y4 (MATFIX)
Algebraic shorthand variables.

X9, Y9 (start, transin, transout, set)
Screen coordinates of ends of plot axes.

XA, YA, XB, YB (ints)
Metric X and Y coordinates of pair of IP's.

XC, YC (circ)
Centre of circle in screen coordinates.

XCOK, YCOK, ZCOK (cokhat)
Metric coordinates of cokhat fix (either device).

XDEG, YDEG (beepin, reset, latlon, readlfix, metdeg)
Device longitude and latitude in degrees.

XDEG0, YDEG0 (beepin, latlon, metdeg, degmet)
Local metric grid origin longitude and latitude in degrees.

XDIV, YDIV (grid)
Number of subdivisions of X and Y axes for gridding.

XI, YI (cokhat)
Metric coordinates of first cokhat IP. XI(), YI()
(start, reset, fix, calc, plot, cokhat)

XI1, YI1, XI2, YI2 (calc, ints)
X & Y Coordinates of 2 IP's for 2 circles.

XJ, YJ, XK, YK (cokhat)
Metric coordinates of second and third cokhat IP's.

XLEN, YLEN (start, transin, transout, set)
Lenths of X and Y plot axes in screen coordinates, =X9-X0 & Y9-Y0.

XMAX, YMAX (start, beepin, fix, transin, transout, plot, set, grid)
Limits of screen plot in metric coordinates.

XMEAN, YMEAN (start, reset, fix, comp, save, transin, transout, olfix)
X & Y coordinates averaged from 2 or more IP's.

XMIN, YMIN (start, beepin, fix, transin, transout, plot, set, scal, grid)
Limits of screen plot in metric coordinates.

XP, YP (fix, plot, circ, scal, grid, olfix, metdeg)
Screen coordinates for plotting.

XP1, YP1 ...to... XP4, YP4 (fix)
Screen coordinates used for plotting fix crosses.

XR, YR (olfix)
Screen coordinates of previous remote fix when plotting a series of fixes.

XRAN, YRAN (start, transin, transout, set, grid)
Lengths of plot axes in metric coordinates, =XMAX-XMIN, YMAX-YMIN.

XS, YS (olfix)
Screen coordinates of previous ship fix when plotting a series of fixes.

XSCA, YSCA (start, transin, transout, set, scal)
Scaling factors for converting from metric coordinates to screen coordinates, =XLEN/XRAN, YLEN/YRAN.

XSD, YSD, ZSD (reset, save, readlfix, transin, datwrit)
Standard deviations of device fix in X, Y and Z.

XX, YY (rarchek, plot, X)
Multi-purpose variables.

XX1 to XX7 (rarchek)
Dummy variables for reading and checking FDF data.

XXMN, YYMN (cokhat)
Means of X & Y coordinates of 3 cokhat IP's.

****
** For variables beginning with Y... see **
** corresponding beginning with X... **
****

Z(I) (start, beepin, reset, datin, remdatip, fix, comp, save, calc, transin, transout, filemake, datwrit, newtrans, MATFIX, reproce)
See X(I).

Z1, Z2, Z3 (MATFIX, matset)
See X1.

Z4 (MATFIX)
See X1.

ZMEAN (reset, fix, comp, save, readlfix, datwrit)
See XMEAN.

ZR (reset, datin, remdatip, MATFIX)
Z coordinate of remote measured negatively from the sea-surface.

ZR$ (remdatip)
String equivalent of ZR.

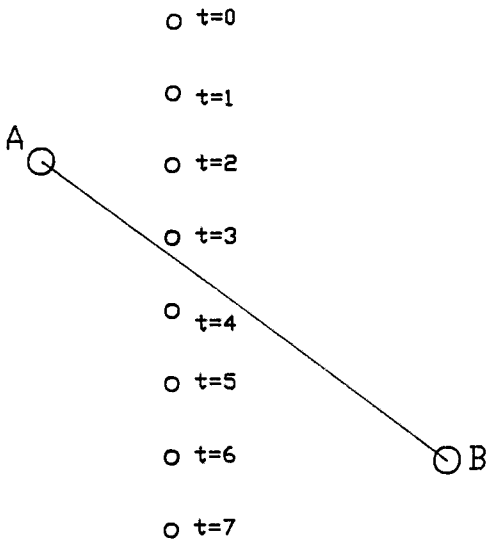
ZRIP (datin, remdatip, fix, cokhat, MATFIX, reproce)
Z coordinate of remote as input from keyboard or from FDF if reprocessing.

ZZ (datin, remdatip, stom)
Vertical distance between device and beacon.

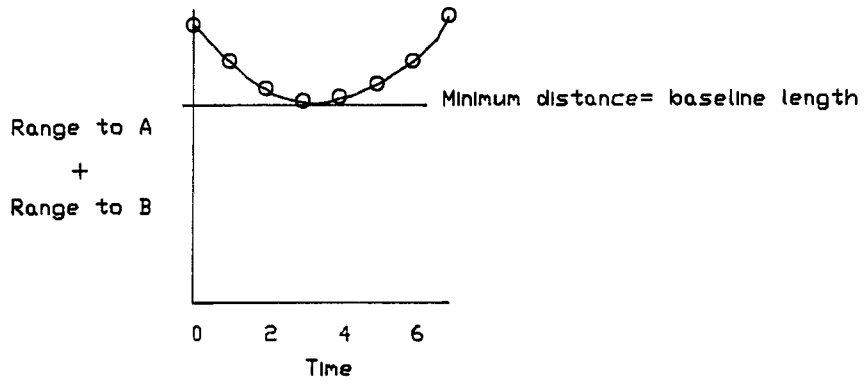
\sjw\navapp4

APPENDIX 4 GRAPHICAL ESTIMATION OF BASELINE LENGTH

i) Ship steams at constant speed and direction across baseline.



ii) Ranges to each beacon are added at frequent intervals.



<tramap on ACNAV View disk>

APPENDIX 5 STRUCTURE OF 'trans' DATAFILE

The trans datafile is used to keep a disk copy of the current values of various parameters. It was initially designed for use in early generation ACNAV programs where different functions were carried out by chaining different programs. Since variables other than A% to Z% are destroyed when loading a new program it was necessary to write and read them to a diskfile. This function is now used to a limited extent, for example i) when a call to the program BEACON is made from NAVPLT during an ACNAV session or ii) when functions are performed which overwrite the variables pertaining to the current fix. The 'trans' file is read at the end of the function to reset the variables to their earlier values.

An 'empty' trans file is created by the procedure PROCnewtrans in NAVPLT. All real variables are given the null value 999999 and string variables are designated "undefnd" i.e. undefined.

The 'full' trans file is written to and read from disk by the procedures PROCtransout and PROCtransin respectively, in NAVPLT.

The format of the file may be revised further in the future and the use of trans may be discontinued when NAVPLT has been sufficiently revised.

ORDER OF VARIABLES IN 'trans'.

SCODE	RCODE	OF$	PREVOF$	BEEP$	
NDAY	NTIME	NBCNS	NRANS		
XMIN	XMAX	YMIN	YMAX	X0 X9	Y0 Y9
XLEN	YLEN	XRAN	RAN	XSCA	YSCA
XMEAN	YMEAN	XMEAN	RYMEAN		
OXMEAN	OYMEAN	OZMEAN	ROXMEAN	ROYMEAN	ROZMEAN
OLTIME	OLDAY	CHEK	QQREM	PRNTR	
SHINT1	SHINT2	SHINT3	RINT1	RINT2	RINT3

X(I), Y(I), Z(I) are filled for I=1 to 14 as follows:-
- I=1 to 6: local metric coords. of Beacons 1 to 6.
- I=7: the latest ship position.
- I=8: the latest remote position.
- I=9, 10: the Std. Deviations of the ship and remote positions.
- I=11, 12: the previous positions of the ship and remote
- I=13: XMIN, YMIN, null respectively.
- I=14: XMAX, YMAX, Z% (=DRINO the drive number for data) respectively.

R(I) I=1 to 6: ranges in m. horizontal from ship to each beacon.
R(7): slant range in m. from ship to remote.
RR(I) I=1 to 6: ranges in m. horizontal from remote to each beacon.
RR(7): slant range in m. from remote to ship (=R(7)).

<fdfmap on ACNAV View disk>

APPENDIX 6 STRUCTURE OF FDF DATAFILE

Numbers refer to byte locations for file pointers as used in random access mode. The first byte is byte 0. Note that real variables occupy 6 bytes, integer variables occupy 5 bytes and string variables occupy N+2 bytes where N is the number of characters in the string. For error-free random access it is essential therefore that character variables are of consistent length, NAVPLT therefore has numerous checks on string length.

At present NAVPLT can read two formats of FDF. The 'old' format stored 31 real variables for each fix, the 'new' format stores 33 real variables, as shown below. NAVPLT now only writes to new format files.

The FDF consists of a header block (bytes 0 to 68 inclusive) which contains parameters describing the whole file. When the FDF is first created each fix is allocated 33 real variable spaces and space is reserved for 100 fixes. All fix variables are set to 0.0 at creation. FDF files are best created before the ACNAV session as the process is slow. The large file size is because only a limited number of files can be stored per disk. Files created at a certain size cannot later be enlarged.

Fix data are written to the file filling it up consecutively from the beginning. As each new fix is written so the details of the header are adjusted to update the total number of real fixes stored and the day and time of the last fix.

See the procedures 'new' for file creation, 'readlfix' for reading, 'datwrit' for writing, and 'hdrd' for reading the header and listing a file.

HEADER (Total No. of bytes=69)

Bytes

0-8	OF$:	the filename.
9-17	PREVOF$:	name of preceeding FDF.
18-26	DTYPE$:	no longer used.
27-32	PTRINT:	the number of bytes allocated for each fix.
33-38	ENDPTR:	the byte following the highest numbered byte occupied by real data.
39-44	NFIXES:	number of real fixes stored in the ile.
45-50	DAY0:	Julian Day of Fix 1.
51-56	TM0:	time (HHMM) of Fix 1.
57-62	DAYEND:	day of latest fix.
63-68	TMEND:	time of latest fix.

FIX DATA (Per Fix: 33 real variables, 6 bytes each, total 198 bytes).

(add NFIX*PTRINT to byte number for respective data of fix number NFIX).

69-74	NFIX:	number of fix in file (1-100)
75-80	NDAY:	Julian day of fix.
81-86	NTIME:	time of fix (HHMM in GMT).

87-92 R(1), 93-98 R(2), 99-104 R(3), 105-110 R(4), 111-116 R(5), 117-122 R(6): ranges of ship from beacons 1 to 6, horizontal m.
123-128 R(7): range of ship from remote transponder.

129-134 RR(1), 135-140 RR(2), 141-146 RR(3), 147-152 RR(4), 153-158 RR(5), 159-164 RR(6): ranges from remote to beacons 1 to 6.
165-170 RR(7): range of remote from ship (=R(7)).

171-176	SCODE:	fix code for ship (see Appendix 13).
177-182	SFXTYP:	fix type code for ship.
183-188	XDEG:	longitude of ship in degrees (- = west).
189-194	YDEG:	latitude of ship in degrees (- = south).
195-200	ZMEAN:	Z position of ship (=0).
201-206	XSD:	SD of X coordinate of ship fix in m.
207-212	YSD:	SD of Y coordinate of ship fix in m.
213-218	ZSD:	SD of Z coordinate of ship fix in m.
219-224	RCODE:	fix code for remote (see Appendix 13).
225-230	RFXTYP:	fix type code for remote.
231-236	RXDEG:	long. of remote in degrees (- = west).
237-242	RYDEG:	lat. of remote in degrees (- = south).
243-248	RZMEAN:	Z position of remote.
249-254	RXSD:	SD of X coordinate of remote fix in m.
255-260	RYSD:	SD of Y coordinate of remote fix in m.
261-266	RZSD:	SD of Z coordinate of remote fix in m.

<bdfmap on ACNAV View disk>

APPENDIX 7 STRUCTURE OF BDF DATAFILE

The Beacon Data File (BDF) is created by the program BEACON, which may be called directly from the program disk or from NAVPLT. It is accessed by both NAVPLT and BEACON. All access is presently in sequential mode, using the OPENIN and OPENOUT functions of BBC basic.

The BDF is usually stored on the same disk as the FDF files, one per disk to avoid confusion.

HEADER

BEEP$:	The BDF filename.
SVEL:	The harmonic mean sound velocity.
DRINO:	The drive number on which the BDF is accessed (usually=1).
NB:	The number of beacons.
XMIN:	The western boundary of the screen plot window in local metres.
XMAX:	The eastern boundary.
YMIN:	The southern boundary.
YMAX:	The northern boundary.
XDEG0:	The longitude of the origin of the local metric grid.
YDEG0:	The latitude of the origin of the local metric grid.

BEACON DATA

(for beacons I=1 to NB)

BCOL$(I):	The reference colour.
TDEL(I):	The receive/transmit delay in ms.
XP(I):	The X-coordinate of the Beacon.
YP(I):	The Y-coordinate.
ZP(I):	The Z-coordinate.

<testdat on ACNAV View disk>

APPENDIX 8 NAVPLT Test Data.

The following data are provided for checking that NAVPLT functions correctly. There are 2 sets of data; the first set is entered into the test BDF file 'DUMPOS' using the program 'BEACON'; the second set is entered as raw fix data into 'NAVPLT'. NAVPLT should be loaded and run as in a normal ACNAV session and a test FDF file should be created. The results obtained for the fix at 123/1020 should agree with the print-out in Appendix 9. The screen plot should show the ship and remote travelling northwest along straight, parallel tracks. The beacons should form a right angled, isosceles triangle.

BDF DATA

BDF filename.....DUMPOS
Sonic velocity.....1517 m/s
Disk drive number.....1
XMIN, XMAX, YMIN, YMAX.....0, 25000, 0, 25000 (m)
Number of beacons.....3
Longitude of grid origin....-25d 30'W
Latitude of grid origin.....+31d 00'N

Beacon	Colour	Delay (ms)	Lat	Long	Z (m)
1	RED	25	31.08999	-25.39491	5200
2	GREEN	25	31.17999	-25.39481	5200
3	BLUE	25	31.17999	-25.28962	5200

RAW FIX DATA

Day	Time	SHP-BCN-SHP TWT (s).			RMT-BCN-SHP lWT's (s).				
		RED	GREEN	BLUE	RED	GREEN	BLUE	DEPTH	RMT-SHP
123	1000	14.54	23.79	21.33	15.88	24.66	21.10	1000	2.08
123	1010	9.51	17.60	17.79	10.58	18.22	16.87	1500	2.27
123	1020	7.92	12.24	16.37	8.01	12.30	14.64	2000	2.38
123	1030	10.92	8.80	17.60	10.01	7.60	15.33	2500	2.57
123	1040	16.27	9.53	20.95	15.02	6.74	18.53	3000	2.80

APPENDIX 9 NAVPLT Printed Output Example.

```
**********************************************************************
                          Day No.:123 Time:1020
SHIP RANGES
Beacon  Slant TWT s.  Slant m. Horiz. m.
RED            7.89     5988.     2970.
GREEN         12.21     9265.     7668.
BLUE          16.34    12398.    11254.

REMOTE RANGES
Beacon   RBdS,1WT s.  Slant m.  Horiz m.      I/P Depth m.
SHIP           2.38     3610.     3006.    -2000.
RED            8.01     6125.     5222.
GREEN         12.30     9356.     8792.
BLUE          14.64     9773.     9235.

***SHIP***
6. good IPs out of 6. possibles.
SHIP Cockhat Fix 3. good ranges IPs:- 2. 3. 6.
                X:11609. Y:12498. Z assumed:0. S.D.:9. m
SHIP   Matrix Fix X:11607. Y:12501.Z:0. Mean Error= 1. m.
SHIP  best Autofix by:MATRIX
   SHIP   USERFIX

 SHIP FIX by:-Matrix........ Fxtyp: 4 IPs/CODE: 123
 X:11607. Y:12501. Z:0. Error:- 1.
***LATITUDE**      :31. d  6.75 ' N
**LONGITUDE**      :25. d 22.68 ' W
   FROM LAST FIX @:1010./123. X:14806. Y:8702. Z:0.
Bearing:320., Distance:4966. m, 10. min ago Speed:16.11 kts
   PREDICTED RANGES from chosen fix position
Beacon  2WT+del. s sl.          Slant m.          Horiz. m.
RED              7.92            5990.             2973.
GREEN           12.24            9266.             7670.
BLUE            16.37           12398.            11255.

***REMOTE***
12.00 good IPs out of 12.00 possibles.
REMOTE Cockhat Fix 4.00 good ranges IPs:- 8. 9. 11.
                X:14609. Y:12508. Z assumed:-2000. S.D.:9. m
REMOTE Matrix Fix X:14601. Y:12499. Z:-2021. Mean Error= 1. m.
REMOTE best Autofix by:MATRIX
   REMOTE USERFIX

 REMOTE FIX by:-Matrix........ Fxtyp: 4 IPs/CODE: 1237
 X:14601. Y:12499. Z:-2021. Error:- 1.
***LATITUDE**      :31. d  6.75 ' N
**LONGITUDE**      :25. d 20.79 ' W
   FROM LAST FIX @:1010./123. X:14899. Y:8701. Z:2021.
Bearing:319., Distance:5028. m, 10. min ago Speed:16.31 kts
   PREDICTED RANGES from chosen fix position
Beacon   1WT+dly s rbs          Slant m.          Horiz. m.
RED              8.01            6126.             5236.
GREEN           12.30            9357.             8800.
BLUE            14.64            9774.             9242.
Distance ship to remote =2994.m.  horiz,  3612. m. slant,   2.38 s. 1WT
Fix No.:3.00        R-A-R CHECK OK
==============================
Datafile:DUMFDF1
SHIP FIX: Matrix........ Type: 4.00 IP Code: 123.00
REMOTE FIX: Matrix........ Type: 4.00 IP Code: 1237.00
```

<fixcode on ACNAV View disk>

APPENDIX 10 FIX TYPES AND CODES.

The parameters FXTYP, FXTYP$ and CODE$ are defined in the procedure PROCfix to record details about the type of fix chosen for the device in question. These are then stored as SFXTYP, SFXTYP$ and SCODE if the device is the ship, or as RFXTYP, RFXTYP$ and RCODE$ if it is the remote. These variables are saved to the FDF file when the fix is complete. They may then be used subsequently to reprocess the data automatically (Section 4.2.9) so that the same fix type is chosen and, if there is a choice the same ranges or intersection points are utilised as in the original fix.

FXTYP is an integer indicating the type of fix. FXTYP$ is a 14 character string describing the fix type. CODE$ is a string consisting of 8 numeric characters. It may be decoded to indicate the ranges (matrix fix) or the intersection points (cokhat, manual fix) used in the fix.

FXTYP	FXTYP$	CODE$	
1	"1 Range No Fix"	"11111111"	only 1 range, no fix
2	"Manual........"	"01020712"	e.g. = IP's 1 2 7 12
3	"Cokhat........"	"00011213"	3 IP's e.g. 1 12 & 13
4	"Matrix........"	"00000134"	e.g. ranges to bcns 1 2 4
8	"No Fix Reqd..."	"88888888"	ranges available, no fix
9	"Not fixed....."	"99999999"	not fixed.

APPENDIX 11 Function Key Strip for NAVPLT program.

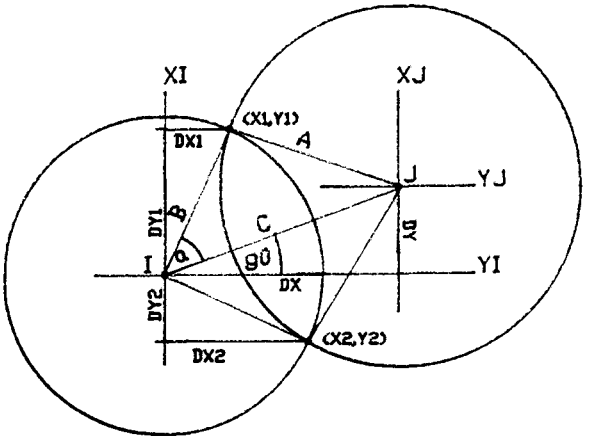
<fstrip on ACNAV View disk>

NAVPLT	F0	F1	F2	F3	F4	F5	F6	F7	F8	F9
	Wipe Screen & List	RUN		MAIN MENU	SPLIT SCREEN	PLOT	FIX	SAVE	MAKE NEW FILES	-

<Matdat on ACNAV View disk>

APPENDIX 12 Mathematics.

1. Calculation of intersection points of 2 circles.



Circles are centred at I (XI,YI) and J (XJ,YJ). Their radii are A and B respectively. The distance between the centres is C. XJ-XI=DX and YJ-YI=DY. The 2 intersection points are (X1,Y1) and (X2,Y2) defined as:-
X1=XI+DX1, Y1=YI+DY1, X2=XI+DX2, Y2=YI+DY2.
The angle g0 is defined by : ARCTAN(g0)=DY/DX.
The angle a is defined as : ARCOS (arg)
arg is defined as : (B.B + C.C - A.A) / (2B.C).
The angle g is defined as : =g0 (when DX >0 and DY >0),
=g0 + PI (when DX <0),
=g0 + 2 PI (when DX >0 and DY <0).

When A+B >C, C >A and C >B (Circles overlap):-
DX1 =B cos(g+a); DY1 =B sin(g+a);
DX2 =B cos(g-a); DY2=B sin(g-a).

When A+B <C (Circles do not overlap):-
DX1 =DX2 =0.5 (C-B-A) cos g;
DY1 =DY2 =0.5 (B+C-A) sin g.

When A >B+C (Circle B enclosed within circle A):-
DX1 =DX2 =0.5(C-B-A) cos g;
DY1 =DY2 =0.5(C-B-A) sin g.

When B >A+C (Circle A enclosed within circle B):-
DX1 =DX2 =0.5(A+B+C) cos g;
DY1 =DY2 =0.5(A+B+C) sin g.

2. Error gradient method of determining ship position.

This method is used in the program NAVVY to derive positions of ship fixes relative to a net of beacons. It is based on the method of Lowenstein as described by Rouse (1987).

The coordinates (Uj,Vj) of the j th ship position are adjusted reiteratively until the error term Ej decreases to an acceptable magnitude:-

$$Uj=Uj -G \frac{d\ Ej}{d\ Uj} =(1-2G)Uj + \frac{2G}{N}\left(\sum_{i=1}^{N} Xi + \sum_{i=1}^{N} Sij(Uj-Xi)\right)$$
$$Tij$$

Vj = likewise.

Xi,Yi,Zi are the coordinates of the i th beacon, i=1 to N. Ej is the mean square error of the j th ship position relative to ranges from all N beacons:-

$$Ej = \frac{1}{N} \sum_{i=1}^{N} (Sij-Tij)^2$$

Sij is the measured slant range from the i th beacon to the j th ship position.

Tij is the calculated slant range from the i th beacon to the j th ship position:-

$$Tij = \left[(Xi-Uj)^2 + (Yi-Vj)^2 + (Zi-0)^2 \right]^{0.5}$$

G is an arbitrary multiplication factor which determines the rate at which the process converges. A value of G=1 is used in the program.

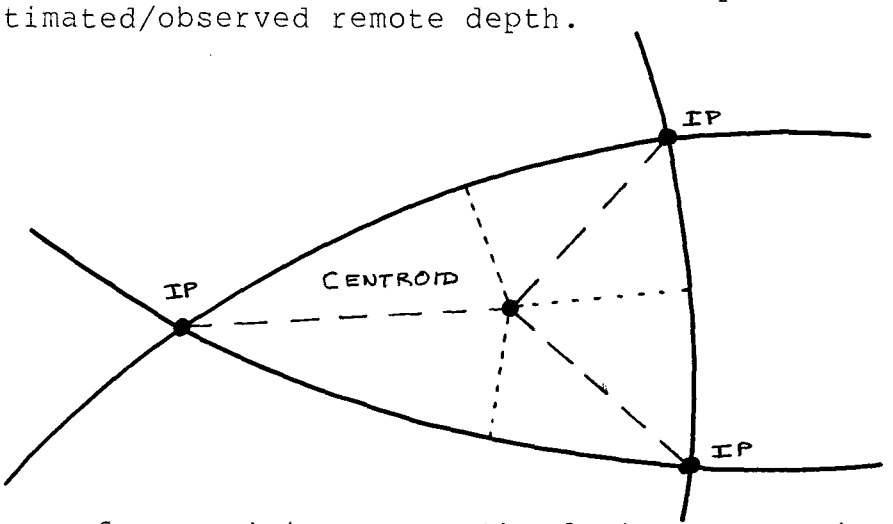
3. Error gradient method of determining the optimum beacon positions.

The method is the same as that described in 2 above except that the beacon coordinates are adjusted reiteratively until the root mean square error between calculated and observed ranges to all ship positions decreases to an acceptable magnitude or until the positional adjustment becomes negligible with further iterations.

The beacon positions are adjusted independently, unconstrained by measured baseline lengths. Once a new set of beacon positions has been calculated, the ship positions may be recalculated and the process repeated. Alternatively if the true geographical coordinates of the ship positions are known and held constant then the true geographical positions of the beacons may be determined.

4. Cokhat method of determining fix position.

The cokhat method assumes that the horizontal plane of the fix is known i.e. z = 0 for the ship or the estimated/observed remote depth.



Ranges from each beacon to the device are projected onto this plane and the intersection points (IP's) of the projected range circles are calculated. Any 3 IP's will form a triangle. The centroid of the triangle is calculated by averaging the x and y coordinates of the 3 IP's. The error is calculated by taking the mean of the distances between each IP and the centroid (dashed lines in the illustration above). The 3 IP's which produce the smallest error are selected for the recommended cokhat fix. Note that the error calculated in this way is generally larger than the error calculated by averaging the deviations between the measured ranges and the predicted ranges to the chosen fix (dotted lines in the illustration).

Sometimes errors in the ranges will cause recommendation of bogus intersection triangles. When this occurs it may be revealed is apparent by comparison with previous positions; the manual option should be used to select alternative IP's.

5. Matrix method for solving Ship position.

The matrix method assumes that the measured ranges are correct, that the ship is at the water surface (z=0) and that the range spheres intersect at a point. The matrix method gives best results when the ranges are fairly accurate. When inaccurate ranges are used the calculated positions are unreliable, often falling outside the recommended cokhat triangle. In such cases the cokhat method is more reliable.

Ship position: Xs,Ys;
Beacon coordinates and ranges Xi,Yi,Zi,Ri, i=1 to 3.

By using the Pythagoras method, each range Ri may be expressed in terms of the ship and beacon coordinates in this form:

$$Xs^2 + Ys^2 - 2\,XiXs - 2\,YiYs - Ri^2 + Xi^2 + Yi^2 + Zi^2 = 0 \qquad i=1,2,3.$$

By algebraic manipulation of these equations we may derive:-

$$\overbrace{Xs.2(X2-X1)}^{a1} + \overbrace{Ys.2(Y2-Y1)}^{b1} = R1^2 - R2^2 + X2^2 + Y2^2 + Z2^2 - X1^2 - Y1^2 - Z1^2 = alfa$$

$$\overbrace{Xs.2(X3-X1)}^{a2} + \overbrace{Ys.2(Y3-Y1)}^{b2} = R1^2 - R3^2 + X3^2 + Y3^2 + Z3^2 - X1^2 - Y1^2 - Z1^2 = beta$$

Here alfa and beta a1, b1, a2, b2 are known.
Arranging in matrix format:-

$$\begin{pmatrix} a1.Xs & b1.Ys \\ a2.Xs & b2.Ys \end{pmatrix} = \begin{pmatrix} alfa \\ beta \end{pmatrix} \quad OR \quad \begin{pmatrix} a1 & b1 \\ a2 & b2 \end{pmatrix}\begin{pmatrix} Xs \\ Ys \end{pmatrix} = \begin{pmatrix} alfa \\ beta \end{pmatrix}$$

The determinant DET = a1b2 -a2b1.
Xs is given by: (alfa b2 - beta b1) / DET
Ys is given by: (beta a1 - alfa a2) / DET.

6. Matrix method for solving Remote position.

A minimum of four ranges are required. The method assumes that the ranges are correct and that the range spheres intersect at a single point.

The ship position Xs,Ys,0 is assumed known, the range from ship to remote is Rs. Beacon coordinates and ranges to the remote are Xi,Yi,Zi,Ri, i=1 to 3.

4 equations may be derived of the form:-

$$Xr^2 + Yr^2 + Zr^2 - 2.XiXr - 2.YiYr - 2.ZiZr - Ri^2 + Xi^2 + Yi^2 + Zi^2 = 0 \quad (i=1,2,3,s)$$

From these we may derive:-

$$2Xr(X2-X1) + 2Yr(Y2-Y1) + 2ZR(Z2-Z1) = R1^2 - R2^2 + X2^2 + Y2^2 + Z2^2 - X1^2 - Y1^2 - Z1^2 = alfa$$

$$2Xr(X3-X1) + 2Yr(Y3-Y1) + 2ZR(Z3-Z1) = R1^2 - R2^2 + X3^2 + Y3^2 + Z3^2 - X1^2 - Y1^2 - Z1^2 = beta$$

$$2Xr(Xs-X1) + 2Yr(Ys-Y1) + 2ZR(Zs-Z1) = R1^2 - R2^2 + Xs^2 + Ys^2 + Zs^2 - X1^2 - Y1^2 - Z1^2 = gamma$$

Here alfa, beta and gamma are known.
Arranging in matrix format:-

$$\begin{pmatrix} a1\,Xr & b1\,Yr & c1\,Zr \\ a2\,Xr & b2\,Yr & c2\,Zr \\ a3\,Xr & b3\,Xr & c3\,Zr \end{pmatrix} = \begin{pmatrix} alfa \\ beta \\ gamma \end{pmatrix}$$

Here a1, b1, etc. are shorthand expressions for 2(X2-X1), 2(Y2-Y1) etc. Hence

$$\begin{pmatrix} a1 & b1 & c1 \\ a2 & b2 & c2 \\ a3 & b3 & c3 \end{pmatrix} \begin{pmatrix} Xr \\ Yr \\ Zr \end{pmatrix} = \begin{pmatrix} alfa \\ beta \\ gamma \end{pmatrix}$$

The determinant DET is given by:

DET = a1(b2c3 -b3c2) -b1(a2c3-a3c2) +c1(a2b3-a3b2).

Xr, Yr and Zr are given by:-

```
Xr.DET =   alfa (b2c3-b3c2)
         - b1   (beta.c3-gamma.c2)
         + c1   (beta.be-gamma.b2)

Yr.DET =   a1   (beta.c3-gamma.c2)
         - alfa (a2c3-a3c2)
         + c1   (a2.gamma-a3.beta)

Zr.DET =   a1   (b2.gamma-b3.beta)
         - b1   (a2.gamma-a3.beta)
         + alfa (a2.b3-a3.b2)
```

FIGURE 1: NAVPLT USER FLOW DIAGRAM

<drawing \sjw\nav006>

Figure shows the usual paths through NAVPLT.

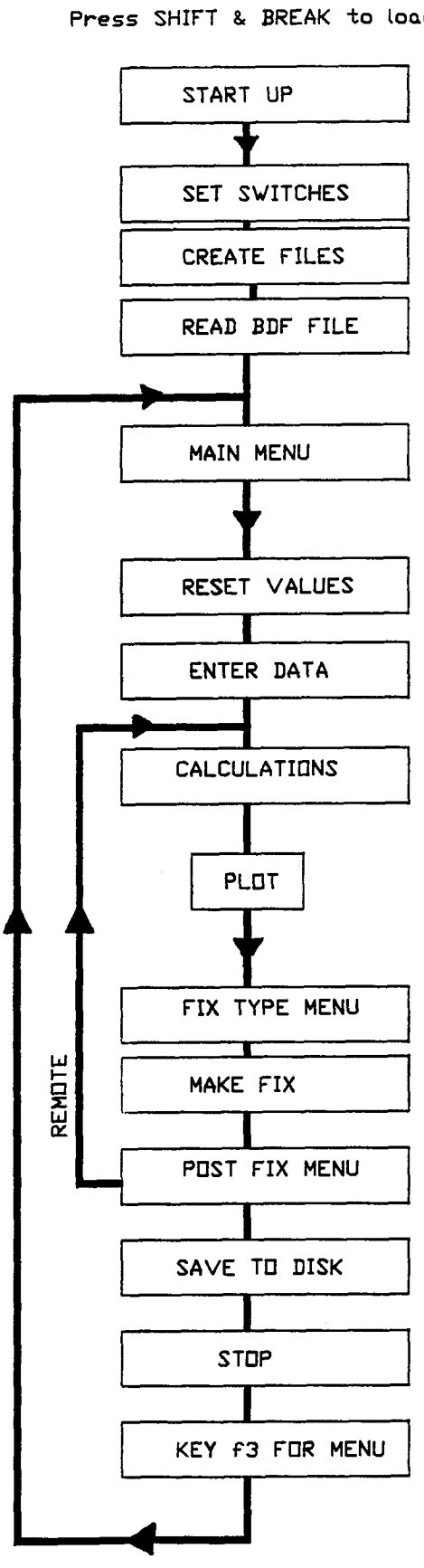


FIGURE 2: NAVPLT CALL PATHS BETWEEN PROCEDURES.

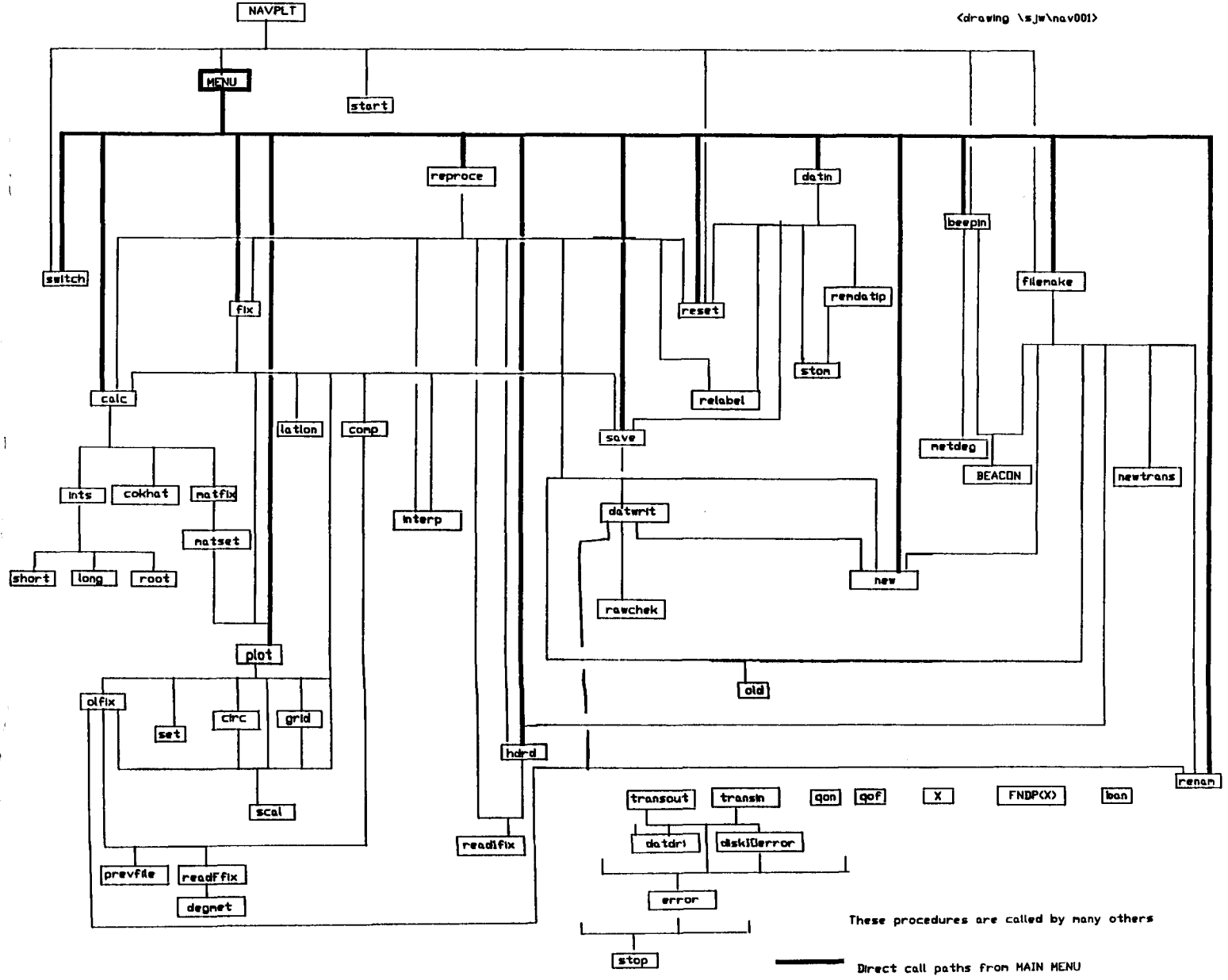
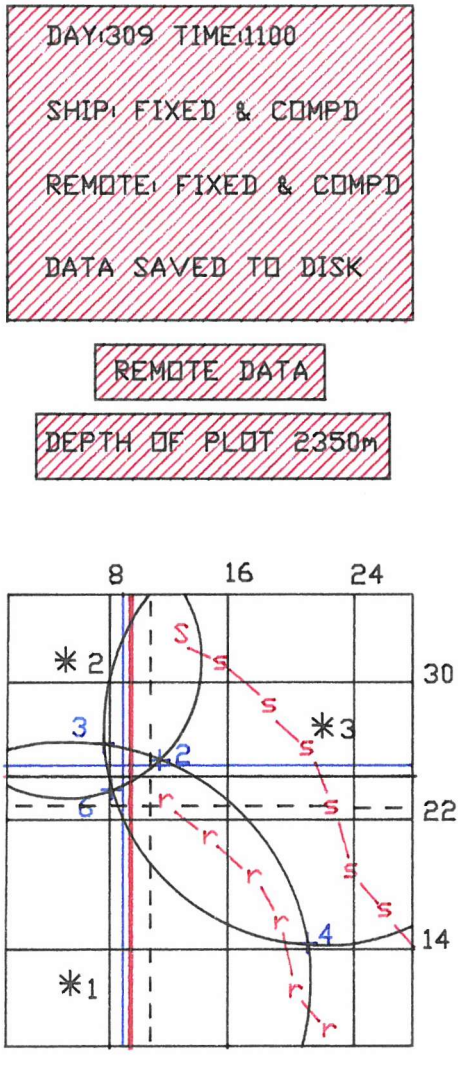


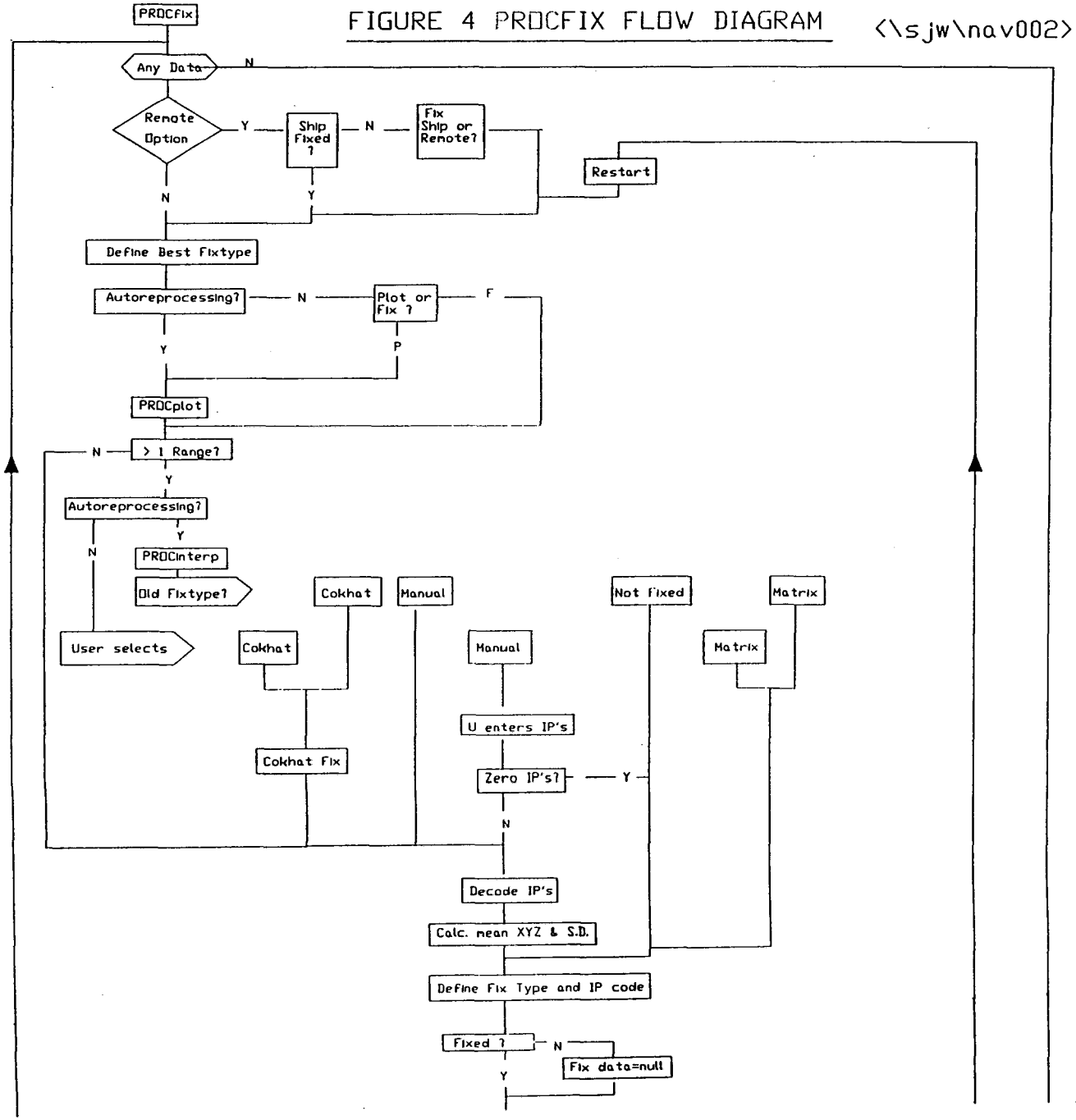
FIGURE 3: NAVPLT SIMULATED GRAPHICS DISPLAY

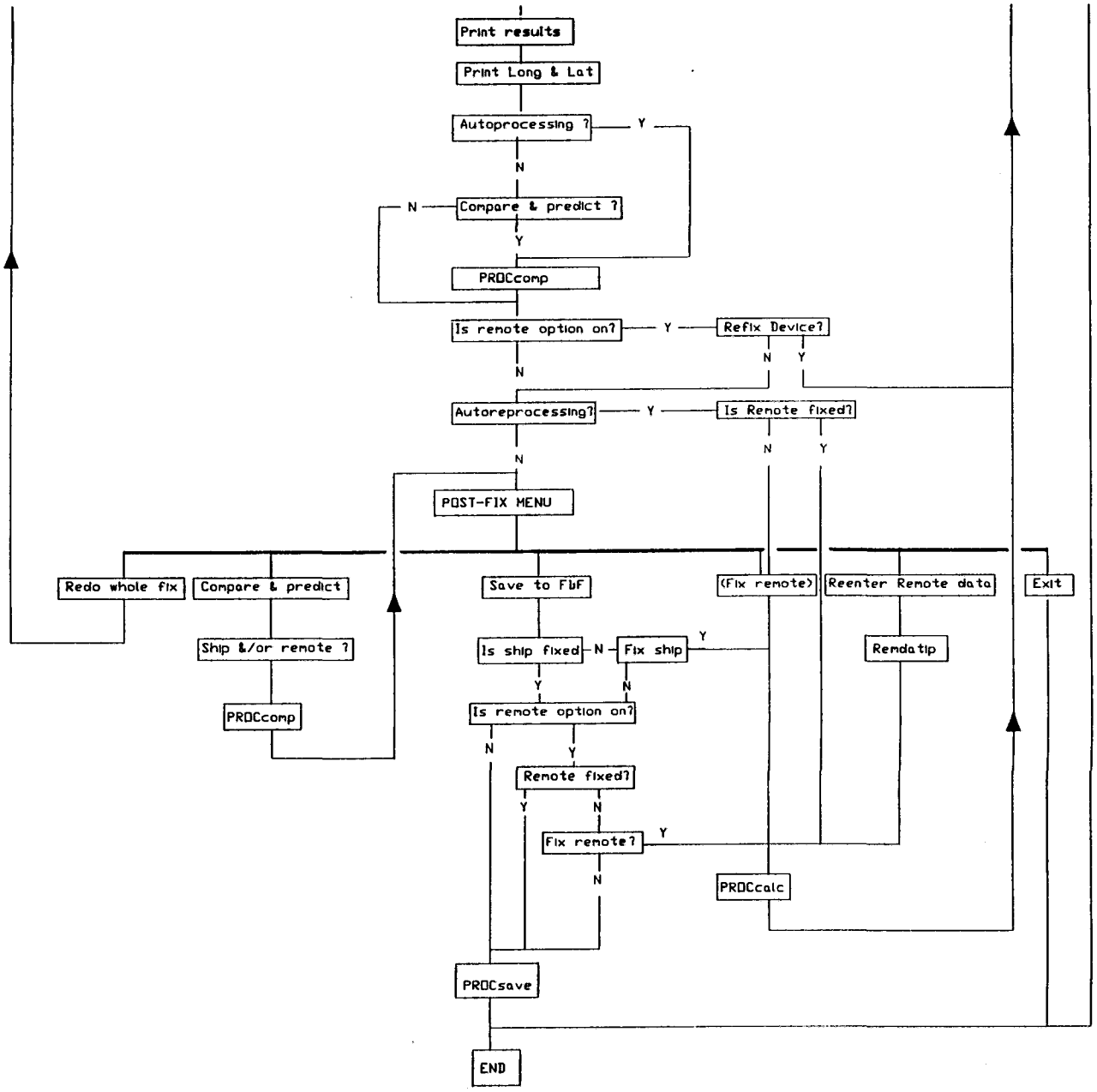
<drawing \sjw\nav004>



KEY		
W: white R: red B: blue		
*3	(W)	Beacon position
	(W)	Range circle
+ 2	(B)	Intersection point
	(R)	Matrix fix position
	(B)	Cokhat fix position
	(W)	User selected fix position
r r r	(R)	Previous remote positions
s s s	(R)	Previous ship positions
S	(R)	Current ship position
R	(R)	Current remote position

FIGURE 4 PROCFIX FLOW DIAGRAM <\sjw\nav002>




Print results
Print Long & Lat
Autoprocessing ?
Y
N
Compare & predict ?
N
Y
PROCcomp
Is remote option on?
Y
Refix Device?
N
Y
N
Autoreprocessing?
Y
Is Remote fixed?
N
Y
N
POST-FIX MENU
Redo whole fix
Compare & predict
Save to FbF
(Fix remote)
Reenter Remote data
Exit
Ship &/or remote ?
Is ship fixed
N
Fix ship
Y
Remdatip
PROCcomp
Y
N
Is remote option on?
N
Y
Remote fixed?
Y
N
Fix remote?
Y
N
PROCcalc
PROCsave
END

CRUISE:- FDF:- BDF:- SHEET:-

		SHIP						REMOTE						
		RED	GREEN	BLUE	ORANGE	BROWN	BLACK		RED	GREEN	BLUE	ORANGE	BROWN	BLACK
DAY:	TWT secs							TWT secs						
TIME:	SLANT mtrs							SLANT mtrs						
FIX:	HORIZ mtrs							HORIZ mtrs						
	LAT:		LONG:		ERROR:			LAT:		LONG:		ERROR:		
								SHIP 1WT:		DEPTH:		CABLE OUT:		
DAY:	TWT secs							TWT secs						
TIME:	SLANT mtrs							SLANT mtrs						
FIX:	HORIZ mtrs							HORIZ mtrs						
	LAT:		LONG:		ERROR:			LAT:		LONG:		ERROR:		
								SHIP 1WT:		DEPTH:		CABLE OUT:		
DAY:	TWT secs							TWT secs						
TIME:	SLANT mtrs							SLANT mtrs						
FIX:	HORIZ mtrs							HORIZ mtrs						
	LAT:		LONG:		ERROR:			LAT:		LONG:		ERROR:		
								SHIP 1WT:		DEPTH:		CABLE OUT:		
DAY:	TWT secs							TWT secs						
TIME:	SLANT mtrs							SLANT mtrs						
FIX:	HORIZ mtrs							HORIZ mtrs						
	LAT:		LONG:		ERROR:			LAT:		LONG:		ERROR:		
								SHIP 1WT:		DEPTH:		CABLE OUT:		
DAY:	TWT secs							TWT secs						
TIME:	SLANT mtrs							SLANT mtrs						
FIX:	HORIZ mtrs							HORIZ mtrs						
	LAT:		LONG:		ERROR:			LAT:		LONG:		ERROR:		
								SHIP 1WT:		DEPTH:		CABLE OUT:		
DAY:	TWT secs							TWT secs						
TIME:	SLANT mtrs							SLANT mtrs						
FIX:	HORIZ mtrs							HORIZ mtrs						
	LAT:		LONG:		ERROR:			LAT:		LONG:		ERROR:		
								SHIP 1WT:		DEPTH:		CABLE OUT:		

FIGURE 5: NAVPLT LOGSHEET

<drawing \sjw\nav007>

FIGURE 6: BEACON FLOW DIAGRAM (SIMPLIFIED)

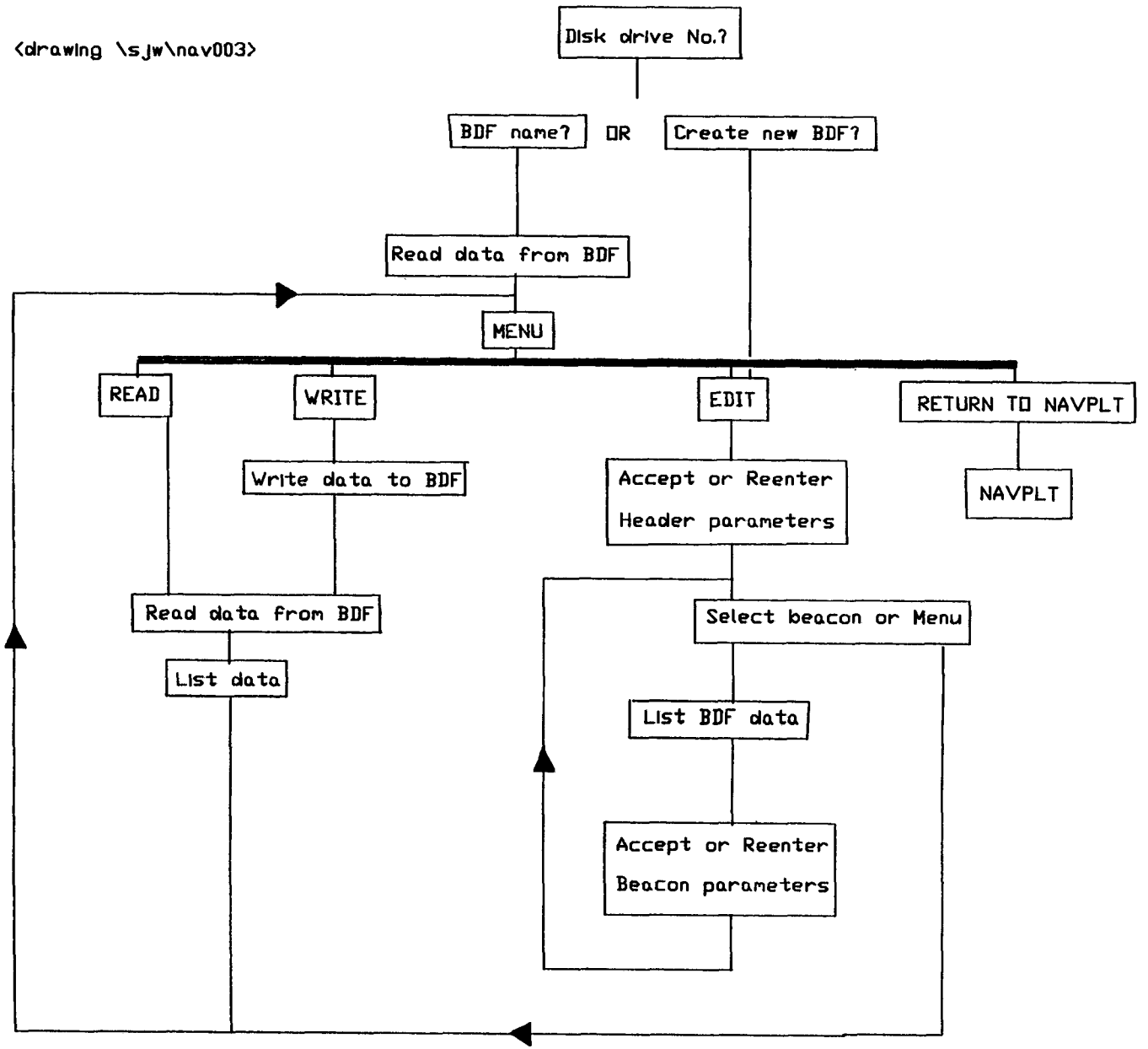
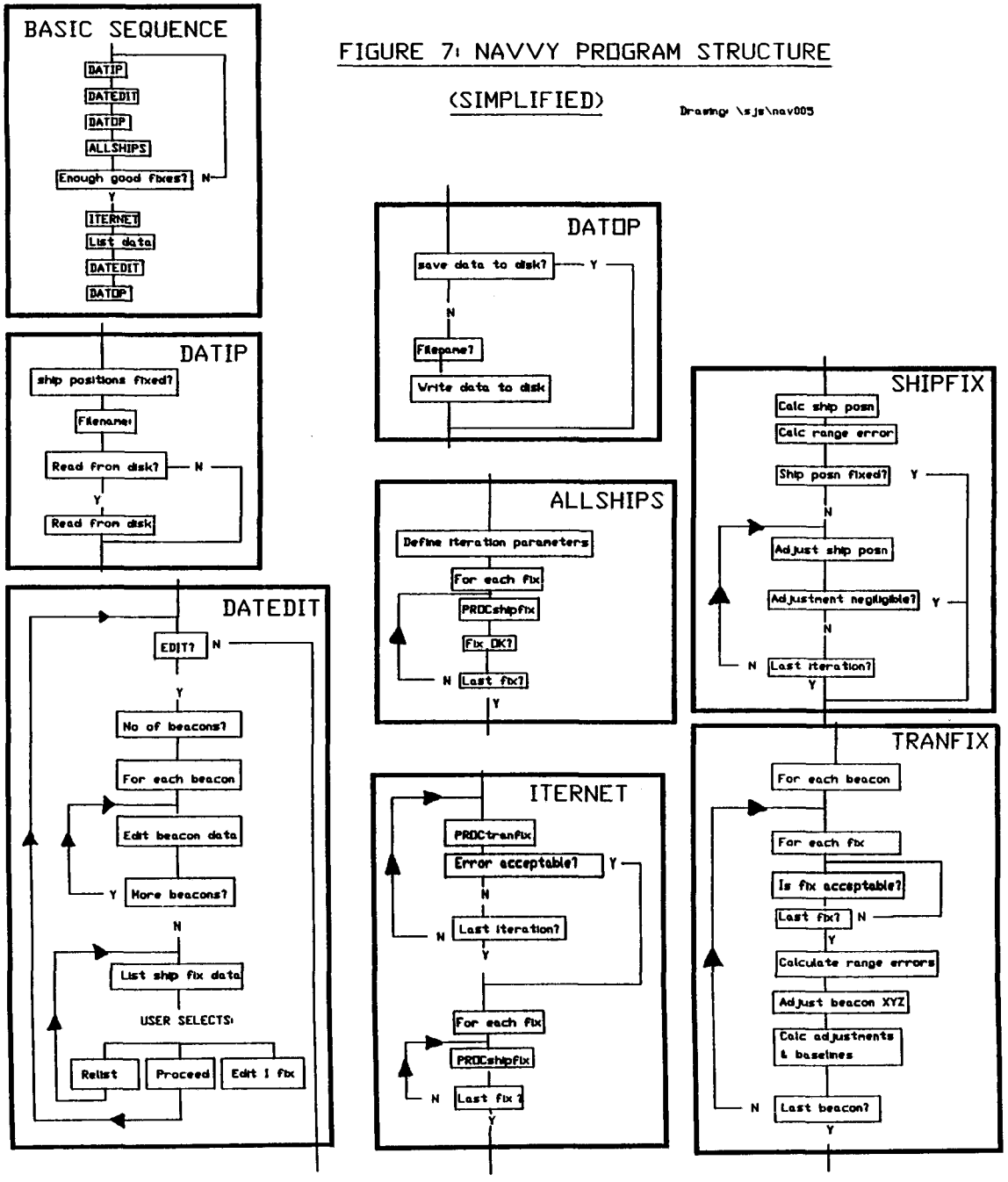


FIGURE 7: NAVVY PROGRAM STRUCTURE

(SIMPLIFIED)

Drawing: \sjs\nav005



NAVVY

```
  1CLS
  100 REM NAVVY by S. Williams, 1/6/87 (after NAV5 by I.Rouse 1986).
  105 REM Program takes a number of ship:txpdr ranges for each fix time. Firstly
 for a new range set, the best ship posn u,v is calculated, given assumed txpdr
posns.
  110 REM Secondly, when sufficient fixes have been entered (>9 for 3 txpdrs), t
he best posn of each txpdr is calc, based on all the range sets (adjusted or not
).
  120 REM Thirdly, given the new set of txpdr posns, the goodness of fit of the
individual ship fixes are calculated. Those with excessive error numbers may be
discarded.
  125 MODE 131
  130 PRINT TIME$
  140 *TV0,1
  150 FORI=1TO 3000:NEXT
  160 *KEY 1 CLS¶M RUN¶M
  170 *KEY 0 CLS¶M LIST¶M
  180 FDAT$="D163GP" :REM  * * * * * * * * * * * * * * * * * * * * * * * * * *
  185 XDEG0=-25 -(40/60):YDEG0= 31 +(05/60)
  190 NTX=5:NFIXES=30:SVEL=1517.6:TDEL=0.030
  200 DIM X(NTX),Y(NTX),Z(NTX),E(NTX),LAT(NTX),LON(NTX)
  210 DIM EFIX(NFIXES),S(NTX,NFIXES),T(NTX,NFIXES),NDAY%(NFIXES),NTIME%(NFIXES),
U(NFIXES),V(NFIXES),LT(NFIXES),LG(NFIXES),SAT%(NFIXES)
  220 REM X,Y,Z are txpdr coords in m., E is error, S,T are slant and true range
s from ship to txpdr, U,V are ship x and y coords, LT and LG are ship lat and lo
ng.
  230 @%=10
  235 REM - - - - - - - - - DATA I/P EDIT & O/P - - - - - - - - - -
  240 PROCdatip:PROCdatedit: PROCdatop
  242 PROCallships: IF NOKFIXES<4 THEN PRINT "More ship fixes required" : GOTO 2
40
  243 PROCiternet
  244 PROCdatlist:PROCdatedit:PROCdatop:STOP
  245 REM ============================================================
  247 DEF PROCallships: REM for each ship fix calls PROCshipfix and counts No. o
f OK fixes, assuming given TXPDR coords.
  250 INPUT"ENTER ITERATION STEP SIZE G (Default=1)";G:IF G=0 THEN G=1
  260 INPUT"Please enter max no. of iterations for each ship fix (Default=100)";
NITMAX:IF NITMAX=0 THEN NITMAX=100
  270 INPUT"Please enter max value of (Fix error/No.  of txpdrs)  before rejecti
on (Default=10)";ERMAX:IF ERMAX=0 THEN ERMAX=10
  280 NOKFXS =0 : REM No. of OK Fixes.
  290 FOR FIX = 1 TO NFIXES : PROCshipfix
  300 IF EFIX(FIX)/NTX < ERMAX THEN NOKFXS = NOKFXS+1 :ELSE PRINT;" REJECTED "
  310 @%=&0A:PRINT;NOKFXS;" OK fixes."
  320 NEXT FIX
  333 ENDPROC
  335 REM ============================================================
```

```
  335 REM ===========================================================
  337 DEF PROCiternet: REM Reiterates PROCtranfix, progressively improving the f
it (reducing the error) of the txpdrs to the fixes  with a fixed set of OK fixes
 as given by PROCallships.
  338 REM Then finally all the shipfixes are run through PROCshipfix assuming th
e given txpdr positions.
  340 NIT=0: REPEAT
  350 NIT=NIT+1:IF NIT MOD 10 =0 THEN VDU2
  351 VDU2
  352 PRINT;"Iteration ";NIT;" ----------------------":PROCtranfix:VDU3
  360 UNTIL R=0 OR NIT >100
  370 FOR FIX=1 TO NFIXES:PROCshipfix:NEXT FIX
  375 ENDPROC
  390 REM ===========================================================
  400 DEF PROCdatip: REM Data I/P from disk PPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP
  403 PRINT; FDAT$; "is present datafile name": INPUT "Type new name (<7 charact
ers) or hit <RTN> if OK";Q$:IF Q$="" THEN GOTO 440:ELSE IFLEN(Q$)>6 THEN GOTO 43
0: ELSE FDAT$=Q$
  404 INPUT"Do you wish to consider ship positions as fixed <Y> or <N>";GEOG$:IF
 GEOG$<>"Y" AND GEOG$<>"N" THEN GOTO 404
  410 INPUT"Hit <RTN> to read from diskfile or <X> to enter all new data";Q$
  416 IF Q$="X"THEN NTX=0:NFIXES=0:ENDPROC
  430 PRINT; FDAT$; "is present datafile name": INPUT "Type new name (<7 charact
ers) or hit <RTN>";Q$:IF Q$="" THEN GOTO 440:ELSE IFLEN(Q$)>6 THEN GOTO 430: ELS
E FDAT$=Q$
  440 PRINT;"I shall read TXPDR data from disk":ICHN=OPENIN(FDAT$):INPUT#ICHN,NT
X
  450 FOR I=1 TO NTX:INPUT#ICHN,X(I),Y(I),Z(I),E(I),LAT(I),LON(I):NEXT
  460 PRINT;"I shall read FIX   data from disk":INPUT#ICHN,NFIXES:
  470 FOR I=1 TO NFIXES:INPUT#ICHN,NDAY%(I),NTIME%(I),U(I),V(I),LT(I),LG(I)
  475 IF GEOG$="Y" THEN XDEG=LG(I):YDEG=LT(I):PROCdegmet:U(I)=UMET:V(I)=VMET:REM
 converts LT(I) & LG(I) into metres Y & X and stores in V(I) & U(I). Uses XDEG0
and YDEG0.
  480 FOR TX=1 TO NTX : INPUT#ICHN, S(TX,I):NEXT TX:NEXT I
  490 CLOSE #ICHN
  500 ENDPROC :REM EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
```

```
  505 REM ================================================================
  510 DEF PROCdatedit: REM For editing all data. PPPPPPPPPPPPPPPPPPPPPPPPPPPPP
  520 INPUT"TYPE <X> to examine/edit/add to  data, else <RTN>";Q$:IF Q$<>"X" THE
N GOTO 660:REM ENDPROCC
  530 PRINT;"No of txpdrs= ";NTX:INPUT"Enter new value or hit <RTN>";Q$:IF Q$<>"
" THEN NTX=VAL(Q$):NNEW=1:ELSE NNEW=0
  540 FOR I=1 TO NTX: PRINT;"X,Y,Z coords (m) ofTxpdr ";I;" =";X(I),Y(I),Z(I):PR
INT;" Lat :";LAT(I);", Lon :";LON(I)
  541 INPUT"<RTN> IF OK OR <X> TO EDIT";Q$:IF Q$="" THEN GOTO 550
  542 INPUT"Do you want to enter positions (1) as X,Y  or (2) as  Lat, Long <1>"
;QQ :IF QQ<>1 AND QQ<>2 THEN GOTO 542
  544 IF QQ=1 THEN INPUT"Please enter:X,Y";X(I),Y(I):XMET=X(I):YMET=Y(I):PROCmet
deg:LAT(I)=YDEG:LON(I)=XDEG:PRINT" I calculate Lat:";LAT(I);", Long:";LON(I)
  546 IF QQ=2 THEN INPUT"Please enter:Lat, Long eg:31,30, -25,40)";LAT(I),LATMIN
,LON(I),LONMIN:LAT(I)=LAT(I)+SGN(LAT(I))*LATMIN/60:LON(I)=LON(I)+SGN(LON(I))*LON
MIN/60:YDEG=LAT(I):XDEG=LON(I):PROCdegmet:X(I)=UMET:Y(I)=VMET
  547 IF QQ=2 THEN PRINT" I calculate X:";X(I);", Y:";Y(I)
  548 INPUT"Please give Z in m.";Z(I)
  550 NEXT I
  560 PROCdatlist
  570 PRINT:NLINE=NLINE+1:IF NLINE=5 THEN NLINE=0:INPUT"Hit <RTN> to continue";Q
$
  590 INPUT"Hit <1> to relist, <2> to edit, <3> to proceed";Q:IF Q<> 1 AND Q<>2
AND Q<>3 THEN GOTO 590: ELSE ON Q GOTO 560,600,660
  600  INPUT"No. of Fix to edit ";I:PRINT;"Fix :";I;", Day :";NDAY%(I);", Time :
";NTIME%(I);", X :";U(I);", Y :";V(I);", Lat :";LT(I);", Lon :";LG(I)
  602 INPUT"<RTN> IF OK OR <X> TO EDIT";Q$:IF Q$="" THEN GOTO 621
  605 INPUT"Do you want to enter positions (1) as X,Y  or (2) as  Lat, Long <1>"
;QQ
  610IF QQ<>1 AND QQ<>2 THEN GOTO 605
  615 IF QQ=1 THEN INPUT"Please enter: Day, time, X,Y";NDAY%(I),NTIME%(I),U(I),V
(I):XMET=U(I):YMET=V(I):PROCmetdeg:LT(I)=YDEG:LG(I)=XDEG:PRINT" I calculate Lat:
";LT(I);", Long:";LG(I)
  620 IF QQ=2 THEN INPUT"Please enter: Day, time, Lat,Long (31,30 -25,40)";NDAY%
(I),NTIME%(I),LT(I),LTMIN,LG(I),LGMIN,:LT(I)=LT(I)+SGN(LT(I))*LTMIN/60:LG(I)=LG(
I)+SGN(LG(I))*LGMIN/60:YDEG=LT(I):XDEG=LN(I):PROCdegmet:U(I)=UMET:V(I)=VMET
  621 FOR TX=1 TO NTX:PRINT;"Txpdr:";TX;", sr(m) :";S(TX,I):NEXT TX:PRINT
  622 INPUT"<RTN> IF OK OR <X> TO EDIT";Q$:IF Q$="" THEN GOTO 640
  624 INPUT"Do you wish to enter ranges as (1) 2WT in s. inc. delay, (2) slant m
etres, (3) horiz m.";QQ
  625 IF QQ<>1 AND QQ<>2 AND QQ<>3 THEN GOTO 624
  630 FOR TX=1 TO NTX:PRINT;"Txpdr ";TX;:INPUT RANGE
  632 IF QQ=1 THEN RANGE=(SVEL*(RANGE-TDEL)/2 )
  634 IF QQ=2 THEN RANGE=RANGE
  636 IF QQ=3 THEN RANGE= SQR( RANGE^2 + (Z(TX))^2)
  637 S(TX,I)=RANGE
  638 NEXT TX
  640 IF I> NFIXES THEN NFIXES=I
  650 GOTO 590
  660 INPUT"Hit <RTN> to proceed <X> to go back and edit from start";Q$:IF Q$="X
" THEN GOTO 530
  665 ENDPROC
  667 REM ================================================================
```

```
  667 REM ===========================================================
  670 DEF PROCdatop: REM Data O/P to disk PPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP
  680 CLOSE #0
  690 INPUT "Do you wish to save new data to disk <Y> or <N>";Q$:IF Q$<>"Y" THEN
 ENDPROC
  700 PRINT;"Writing data to disk"
  710 PRINT; FDAT$; "is present datafile name": INPUT "Type new name (<7 charact
ers) or hit <RTN>";Q$:IF Q$="" THEN GOTO 720 :ELSE IF LEN(Q$)>6 THEN GOTO 710: E
LSE FDAT$=Q$
  720 OCHN=OPENOUT(FDAT$):PRINT#OCHN,NTX
  730 FOR I=1 TO NTX
  732 IF GEOG$<>"Y" THEN XMET=U(I):YMET=V(I):PROCmetdeg:LT(I)=YDEG:LG(I)=XDEG
  734 PRINT#OCHN,X(I),Y(I),Z(I),E(I),LAT(I),LON(I):NEXT
  740 PRINT#OCHN, NFIXES
  750 FOR I=1 TO NFIXES:PRINT #OCHN,NDAY%(I),NTIME%(I),U(I),V(I),LT(I),LG(I)
  760 FOR J=1 TO NTX: PRINT#OCHN, S(J,I):NEXT J:NEXT I
  770 CLOSE #OCHN:PRINT "Data written to:";FDAT$
  780 ENDPROC
  790 REM ===========================================================
  800 DEF PROCshipfix: REM Fixes one ship position: PPPPPPPPPPPPPPPPPPPPPPPPPPP
  810 VDU2:PRINT;"Ship Fix No.:";FIX:VDU3
  820 NITS=0
  830 NITS=NITS+1:A=0:B=0:C=0:D=0:EFIX(FIX)=0
  835 PRINT;"It No:";NITS;
  840 REM ******************** Calculate ship position  +  measured to predicted
 range error **********************
  850 NRANS=0
  860 FOR TX=1 TO NTX
  870 IF S(TX,FIX)=0 THEN GOTO 910: ELSE NRANS=NRANS+1
  880 A=A+X(TX):B=B+Y(TX)
  890 T(TX,FIX)=((X(TX)-U(FIX))^2+(Y(TX)-V(FIX))^2+Z(TX)^2)^0.5
  900 C=C+ (U(FIX)-X(TX))*S(TX,FIX)/T(TX,FIX): D=D+ (V(FIX)-Y(TX))*S(TX,FIX)/T(T
X,FIX) :EFIX(FIX)=EFIX(FIX) + ABS(S(TX,FIX)-T(TX,FIX))
  910 NEXT TX
  915 IF GEOG$="Y" THEN GOTO 980: REM DO NOT ADJUST RELIABLE SHIP POSITIONS
  920 REM ************** Adjust Ship Position ********************************
  930 U=U(FIX): V=V(FIX)
  935 @%=&02020A:PRINT "u= ";U;" v= ";V;"  ";
  940 U(FIX)=U*(1-2*G)+(2*G/NRANS)*(A+C)
  950 V(FIX)=V*(1-2*G)+(2*G/NRANS)*(B+D)
  952 GOTO 960
  955 XMET=U(FIX):YMET=V(FIX):PROCmetdeg:LT(FIX)=YDEG:LG(FIX)=XDEG
  960 R=0
  965 REM ************* Check whether adjustment gradient is negligible yet ****
************************
  968 PRINT ;" dU=";ABS(U(FIX)-U);" dV=";ABS(V(FIX)-V)
  970 IF ABS(U(FIX)-U)>0.1 OR ABS(V(FIX)-V) > 0.1 THEN R=1 :IF NITS <NITMAX THEN
 GOTO 830: REM Limit of NITMAX iterations for ship fix.
  980 @%=&20209
  990 VDU2:PRINT; " X, Y, Error: ";U(FIX);", ";V(FIX);", ";EFIX(FIX)/NRANS;" ":V
DU3
 1000 ENDPROC
 1005 REM ===========================================================
```

```
 1005 REM ========================================================================
 1020 DEF PROCtranfix: REM Fixes all NTX transponders using NOKFXS fixes. PPPPP
 1030 FOR TX= 1 TO NTX
 1040 E=0:F=0:G2=0:H=0:K=0:E(TX)=0
 1050 E(TX)=0
 1060 REM ---------------------------------------
 1070 FOR FIX=1 TO NFIXES
 1080 IF EFIX(FIX)/NTX >ERMAX THEN GOTO 1150
 1090 E=U(FIX)+E: F=V(FIX)+F
 1100 T(TX,FIX)=((X(TX)-U(FIX))^2+(Y(TX)-V(FIX))^2+Z(TX)^2)^0.5
 1110 G2=(X(TX)-U(FIX))*S(TX,FIX)/T(TX,FIX)+G2
 1120 H=(Y(TX)-V(FIX))*S(TX,FIX)/T(TX,FIX)+H
 1130 K=Z(TX)*S(TX,FIX)/T(TX,FIX)+K
 1140 E(TX)= E(TX)+ ABS(S(TX,FIX)-T(TX,FIX))
 1150 NEXT FIX
 1160 REM ---------------------------------------
 1170 X=X(TX): Y=Y(TX):Z=Z(TX)
 1180 X(TX)=X*(1-2*G)+(2*G/NOKFXS)*(E+G2):Y(TX)=Y*(1-2*G)+(2*G/NOKFXS)*(F+H):Z(T
X)=Z*(1-2*G)+(2*G/NOKFXS)*K
 1190 DX= ABS(X(TX)-X): DY= ABS(Y(TX)-Y) :DZ= ABS(Z(TX)-Z)
 1200 IF DX> 0.1 OR DY > 0.1 OR DZ > 0.1 THEN R=1 ELSE R=0
 1210 @%=&020209:PRINT;" Txpdr:";TX;" X,Y,Z,error :";X(TX);", ";Y(TX);", ";Z(TX)
;", ";E(TX)/NOKFXS
 1213 XMET=X(TX):YMET=Y(TX):PROCmetdeg:@%=&020009:PRINT;"BEACON ";TX; " Latitude
= ";INT(ABS(YDEG));",";:@%=&020209:PRINT;60*ABS(YDEG-INT(ABS(YDEG))*SGN(YDEG));
 1214 @%=&020009 :PRINT;" Longitude= ";INT(ABS(XDEG));",";:@%=&020209:PRINT;60*A
BS(XDEG-INT(ABS(XDEG))*SGN(XDEG))
 1215 IF TX=1 THEN GOTO 1220
 1216 FOR I=1 TO TX-1: BLD= SQR( (X(TX) - X(I))^2 + (Y(TX)-Y(I))^2 ):PRINT;"Base
line ";TX;" to ";I;" = ";BLD;" metres.": NEXT
 1220 NEXT TX
 1230 ENDPROC
 1235 REM ========================================================================
 1240 DEF PROCdatlist: REM Lists fix data PPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP
 1250 INPUT"Now I shall list fix data, please enter start and end fix No.s or hi
t <RTN> to list all";Q$:IF Q$<>"" THEN F0=VAL(Q$):INPUT F9:ELSE F0=1:F9=NFIXES
 1260 INPUT "Hit <X> to get print out else <RTN>";Q$:IF Q$<>"" THEN VDU2
 1270 NLINE=0: REM This is used to control screen listing
 1275 FOR I=1 TO NTX: PRINT;"X,Y,Z coords (m) ofTxpdr ";I;" =";X(I),Y(I),Z(I):PR
INT;" Lat :";LAT(I);", Lon :";LON(I):NEXT I
 1280 FOR I= F0 TO F9:PRINT;"Fix=";I;" Day=";NDAY%(I);" Time=";NTIME%(I);" X=";U
(I);" Y=";V(I);" Lat=";LT(I);" Lon=";LG(I)
 1290 PRINT;"Slant ranges (m) to txpdrs: ";: FOR TX=1 TO NTX:PRINT;TX;":";S(TX,I
);" ";:NEXT TX
 1295 PRINT:NEXT I :VDU3
 1300 ENDPROC
 1395 REM ========================================================================
```

```
 1395 REM ===========================================================================
 1400 DEF PROCdegmet:REM For converting LT(I) and LG(I) into VMET and UMET
 1415 VMET=(YDEG-YDEG0)*1852*60:UMET=(XDEG-XDEG0)*1852*COS(RAD(YDEG))*60
 1450 ENDPROC
 1460 REM ===========================================================================
 1470 DEF PROCmetdeg
 1480 YDEG=YDEG0+YMET/(1852*60):MULT=SGN(YDEG)*COS(RAD(YDEG)):XDEG=XDEG0+XMET/(M
ULT*1852*60)
 1490 ENDPROC
 1800 DEF PROCshipfix2: REM Fixes one ship position by moving it from point to p
oint on an 80 x 80 grid (interval=50 m, sides=4 Km).
 1810 PRINT;"Ship Fix No.:";FIX;
 1815 MDEV=9999999:BESTX=0:BESTY=0
 1820 XXX=U(FIX):YYY=V(FIX)
 1830 FOR X=XXX-2000 TO XXX+2000 STEP 50
 1840 FOR Y=YYY-2000 TO YYY+2000 STEP 50
 1845 DEV=0:FOR TX=1 TO NTX
 1850 SPRED=SQR( (X(TX)-X)^2 +(Y(TX)-Y)^2 +Z(TX)^2)
 1860 SMEAS=S(TX,FIX)
 1870 DEV=DEV+SQR( (SMEAS-SPRED)^2)
 1875 IF DEV<MDEV THEN BESTX=X:BESTY=Y :MDEV=DEV
 1880 PRINT;"X,Y,MDEV,BESTX,BESTY";" ";X;" ";Y;" ";MDEV;" ";BESTX;" ";BESTY
 1890 NEXT Y:NEXT X
 1900 PRINT; " X, Y, Error: ";BESTX;", ";BESTY;", ";MDEV/NRANS;" "
 1910 ENDPROC
 2000 REM ===========================================================================
 9999 REM Dummy code for test purposes.
10000 INPUT"Please enter:Lat, Long eg:31,30, -25,40)";LAT(I),LATMIN,LON(I),LONMI
N:LAT(I)=LAT(I)+SGN(LAT(I))*LATMIN/60:LON(I)+SGN(LON(I))*LONMIN/60:YDEG=LAT(I):X
DEG=LON(I):PROCdegmet:X(I)=UMET:Y(I)=VMET
```

```
 6200  REM - - - - - - - - Matrix Fix - - - - - - - - - - - -
 6220 NI=3:CODE$=STR$(MCODE) :FXTYP=4: PRINT;"Matrix Fix"
 6240 IF SOR$="S"THEN :MEANX=SXMAT:MEANY=SYMAT:MEANZ=SZMAT:SDX=SSDMAT:SDY=SSDMAT
:SDZ=SSDMAT:GOTO6700:ELSE MEANX=RXMAT:MEANY=RYMAT:MEANZ=RZMAT:SDX=RSDMAT:SDY=RSD
MAT:SDZ=RSDMAT:GOTO6700
 6260 REM  - - - - - - - - - -  Manual Fix - - - - - - - - - - - - - - - - - - -
 6270 *FX210,0
 6280 CODE$="":FXTYP=2:PRINT;"Manual Fix"
 6300 VDU3:COLOUR 130:PRINT "Please enter up to 4 valid IP No.s (BLUE '+' s) OR
<RTN> if no fix wanted":NI=0:PROCqof
 6320 VDU7:PRINT;"Please enter IP No. ";:INPUT" or <RTN> if no more";DUM$:IF DUM
$="" THEN GOTO 6360 :ELSE IF LEN(DUM$)<2 THEN DUM$="0"+DUM$
 6340 NI=NI+1:VI(NI)=VAL(DUM$):CODE$=CODE$+DUM$:IF NI<4 THEN GOTO 6320
 6360 IF CODE$<>"" THEN GOTO 6440 ELSE PROCX:PRINT;"Zero IPs: NO FIX WANTED???"
 6380 PRINT;" ARE YOU SURE ?? <RTN> to refix OR <Y> for no fix";:PROCqof:INPUT D
UM$:IF DUM$=""THEN GOTO 5720:ELSE FXTYP=8:GOTO 6700
 6400 REM- - - - - - - -Cokhat Fix - - - - - - - - - - - - - -
 6420 CODE$=CHCODE$ : NI=3:FXTYP=3:PRINT;"Cokhat Fix"
 6440 REM- - - - - - - -Cokhat or Manual Fix - - - - - - - - - - -
 6460 VDU3:NI=0:FOR P%=1 TO LEN(CODE$)-1 STEP 2: NI=NI+1:VI(NI)=VAL(MID$(CODE$,P
%,2)):NEXT P%:REM Decode CODE$, No.s of IP's stored in VI() array.
 6480  SUMX=0:SUMY=0:SUM2X=0:SUM2Y=0
 6500 REM - - - - - - - - - - - - - -
 6520  FORI%=1TO NI:IF SOR$="S" THEN IX=XI(VI(I%)):IY=YI(VI(I%)): ELSE IX=RXI(VI
(I%)):IY=RYI(VI(I%))
 6540 SUMX=SUMX+IX:SUMY=SUMY+IY
 6560 SUM2X=SUM2X+IX^2:SUM2Y=SUM2Y+IY^2:NEXT
 6580 REM - - - - - - - - - - - - - -
 6600 MEANX=SUMX/NI:MEANY=SUMY/NI:MEANZ=0:IF SOR$="R" THEN MEANZ=RZCOK
 6620 VARX=(SUM2X/NI -MEANX^2):IF VARX<=0 THEN SDX=0 ELSE SDX=SQR(SUM2X/NI -MEAN
X^2)
 6640 VARY=(SUM2Y/NI -MEANY^2):IF VARY<=0 THEN SDY=0 ELSE SDY=SQR(SUM2Y/NI -MEAN
Y^2)
 6660 SDZ=0
 6680 REM
 6700 REM - - - - - - - - All Fix types - - - - - - - - - - - - - -
 6740 IF FXTYP=1 THEN FXTYP$="1 Range No Fix": CODE$="11111111"
 6760 IF FXTYP=2 THEN FXTYP$="Manual........": CODE$=CODE$
 6780 IF FXTYP=3 THEN FXTYP$="Cokhat........": CODE$=CODE$
 6800 IF FXTYP=4 THEN FXTYP$="Matrix........": CODE$=CODE$
 6820 IF FXTYP=8 THEN FXTYP$="No fix Reqd...": CODE$="88888888"
 6840 IF FXTYP=9 THEN FXTYP$="Not fixed.....": CODE$="99999999"
 6850 IF FXTYP=1 OR FXTYP=8 OR FXTYP=9 THEN MEANX=999999:MEANY=999999:MEANZ=9999
99:SDX=999999:SDY=SDX:SDZ=SDX:IF SOR$="R" THEN MEANZ=ZRIP:SDZ=0
 6860 CODE=VAL(CODE$)
 6880 IF PRNTR=1 THEN VDU2
 6900 PRINT:IF SOR$="R" THEN RFXTYP$=FXTYP$:RFXTYP=FXTYP:PRINT;" REMOTE FIX by:-
";FXTYP$;: ELSE SFXTYP$=FXTYP$:SFXTYP=FXTYP:PRINT;" SHIP FIX by:-";SFXTYP$;
 6920 PRINT;" Fxtyp: ";FXTYP;" IPs/CODE: ";CODE$
 6940 @%=&020009
 6960 PRINT;" X:";MEANX;" Y:";MEANY;" Z:";MEANZ;" Error:- ";(SDX+SDY+SDZ)/3:VDU3
 6980 PROCscal(XMIN,MEANY):XP1=XP:YP1=YP:PROCscal(XMAX,MEANY):XP2=XP:YP2=YP:PROC
scal(MEANX,YMIN):XP3=XP:YP3=YP:PROCscal(MEANX,YMAX):XP4=XP:YP4=YP:VDU4
 7000 GCOL 0,3:VDU5:MOVE XP1,YP1:PLOT 21,XP2,YP2:MOVE XP3,YP3:PLOT 21,XP4,YP4:VD
U4
 7020 PROClatlon:IF SOR$="S" THEN SXDEG=XDEG:SYDEG=YDEG:ELSE RXDEG=XDEG:RYDEG=YD
EG
 7040 IF SOR$="S" THEN SCODE=CODE:SFIX$="Y":XMEAN=MEANX:YMEAN=MEANY:ZMEAN=MEANZ:
SDZ=Z(9):X(7)=XMEAN:Y(7)=YMEAN:Z(7)=ZMEAN:X(9)=SDX:Y(9)=SDY:Z(9)=SDZ
 7060 IF SOR$="R" THEN RCODE=CODE:RFIX$="Y":RXMEAN=MEANX:RYMEAN=MEANY:RZMEAN=MEA
NZ:RXSD=SDX:RYSD=SDY:RZSD=SDZ:X(8)=RXMEAN:Y(8)=RYMEAN:Z(8)=RZMEAN:X(10)=RXSD:Y(1
0)=RYSD:Z(10)=RZSD
 7080 bn$="FIXED":x1=210:x2=390::y0=800:y9=840:IF SOR$="R" THEN y0=760:y9=800
 7100 PROCban(x1,x2,y0,y9,bn$,1): REM Banner FIXED.
```

```
 5600 REM=======================================================================
 5620 DEF PROCfix
 5640 @%=&020209
 5660 IF DATIN$<>"Y" THEN PRINT;"No data to fix":GOTO 7760:REM ENDPROC
 5680 CODE$="":SOR$="S":IF QREM$<>"Y" THEN GOTO 5700: ELSE IF SFIX$="Y" AND RIPC
AL$="Y" THEN PROCqon:PRINT;"Do you wish to fix  ";:PRINT;"<RTN> ship          ":
INPUT"<R>   or remote     ";Q$:PROCqof:IF Q$="R" THEN SOR$="R"
 5700 REM - - - - - - Target for repeat fix - - - - - - - - - - -
 5720 IF AUTPROC$="Y" THEN PROCplot:GOTO 5760
 5740 PROCqon:PRINT;"<RTN>  To Plot       ";:INPUT"<X>  to Fix        ";Q$:IF Q$=
"" THEN PROCplot
 5760 IF PRNTR=1 THEN VDU2
 5780 IF SOR$="R" THEN RFIX$="N":PRINT;"   REMOTE USERFIX   ":ELSE SFIX$="N":PRI
NT;"   SHIP   USERFIX   ":VDU3
 5800 IF SOR$="S" THEN VI(1)=SHINT1:VI(2)=SHINT2:VI(3)=SHINT3: ELSE  VI(1)=RINT1
:VI(2)=RINT2:VI(3)=RINT3
 5820 CHCODE$="":FOR I%=1 TO 3:C$=STR$(VI(I%)):IFLEN(C$)=1 THEN C$="0"+C$
 5840 CHCODE$=CHCODE$+C$:NEXT I%: CHCODE=VAL(CHCODE$)
 5860 VDU3:PRINT;"Best 3 Cokhat IPs: ";CHCODE$
 5880 REM - - - - - - PRE FIX MENU - - - - - - - - - - - - - - - - - - - - - -
 5900  B$=SBSTFX$:IF SOR$="R" THEN B$=RBSTFX$
 5920 IF B$="unfixed" THEN B$="MANUAL"
 5940 PRINT;" Best Autofix method:-";:COLOUR 129:PRINT;B$:PROCqof:VDU3
 5960 IF NGRANS=1 THEN FXTYP=1:GOTO 6700:REM 1 range no fix
 5980 REM - - - - - - - - -   Check fixtype if auto processing - - - - - - - - -
 6000 IF AUTPROC$="N" OR REPROC$="N" THEN GOTO 6120:ELSE CODE=SCODE:IF SOR$="R"
THEN CODE=RCODE
 6010 FXTYP=SFXTYP:IF SOR$="R" THEN FXTYP=RFXTYP
 6020 IF REPROC$="Y" AND PTRINT=186 THEN PROCinterp
 6040 IF PRNTR=1 THEN VDU2
 6060 PRINT;" AUTOPROC: FXTYP= ";FXTYP;" OLD CODE= ";CODE:VDU3
 6080 IF FXTYP=1 OR FXTYP=8 OR FXTYP=9 THEN GOTO 6700:ELSE IF FXTYP=2 THEN GOTO
6440:ELSE IF FXTYP=3 THEN GOTO 6400:ELSE IF FXTYP=4 THEN GOTO 6200
 6100 REM - - - - - - - FIX TYPE MENU - - - - - - - - - - - - - - - - - - - - -
 6120 VDU3:PROCqon:PRINT;"FIX TYPE MENU: Select:":PRINT;"<RTN> as above (RED)";:
PRINT;"<1>  Cokhat Fix ":PRINT;"<2>  Manual Fix ":PRINT;"<3>  Matrix Fix ":PRINT
;"<4>  QUIT            "
 6125 IF AUTPROC$="Y" THEN DUM$="":GOTO 6140
 6130 INPUT DUM$:PROCqof:IF LEN(DUM$)>1 OR VAL(DUM$)>4 THEN GOTO 6120
 6140 IF DUM$<>"" THEN GOTO 6160:ELSE IF B$="COKHAT" THEN DUM$="1":ELSE IF B$="M
ANUAL" THEN DUM$="2":ELSE IF B$="MATRIX" THEN DUM$="3"
 6160  DUM=VAL(DUM$):ON DUM GOTO 6400,6260,6200,7760:REM Endproc
 6180 REM
```

```
 6200  REM - - - - - - - - Matrix Fix - - - - - - - - - - - -
 6220 NI=3:CODE$=STR$(MCODE) :FXTYP=4: PRINT;"Matrix Fix"
 6240 IF SOR$="S"THEN :MEANX=SXMAT:MEANY=SYMAT:MEANZ=SZMAT:SDX=SSDMAT:SDY=SSDMAT
:SDZ=SSDMAT:GOTO6700:ELSE MEANX=RXMAT:MEANY=RYMAT:MEANZ=RZMAT:SDX=RSDMAT:SDY=RSD
MAT:SDZ=RSDMAT:GOTO6700
 6260 REM  - - - - - - - - - -  Manual Fix - - - - - - - - - - - - - - - - - -
 6270 *FX210,0
 6280 CODE$="":FXTYP=2:PRINT;"Manual Fix"
 6300 VDU3:COLOUR 130:PRINT "Please enter up to 4 valid  IP No.s (BLUE '+' s) OR
<RTN> if no fix wanted":NI=0:PROCqof
 6320 VDU7:PRINT;"Please enter IP No. ";:INPUT" or <RTN> if no more";DUM$:IF DUM
$="" THEN GOTO 6360 :ELSE IF LEN(DUM$)<2 THEN DUM$="0"+DUM$
 6340 NI=NI+1:VI(NI)=VAL(DUM$):CODE$=CODE$+DUM$:IF NI<4  THEN GOTO 6320
 6360 IF CODE$<>"" THEN GOTO 6440 ELSE PROCX:PRINT;"Zero IPs: NO FIX WANTED???"
 6380 PRINT;" ARE YOU SURE ?? <RTN> to refix OR <Y> for  no fix";:PROCqof:INPUT D
UM$:IF DUM$=""THEN GOTO 5720:ELSE FXTYP=8:GOTO 6700
 6400 REM- - - - - - - -Cokhat Fix - - - - - - - - - - - - - - -
 6420 CODE$=CHCODE$ : NI=3:FXTYP=3:PRINT;"Cokhat Fix"
 6440 REM- - - - - - - -Cokhat or Manual Fix - - - - - - - - - - -
 6460 VDU3:NI=0:FOR P%=1 TO LEN(CODE$)-1 STEP 2: NI=NI+1:VI(NI)=VAL(MID$(CODE$,P
%,2)):NEXT P%:REM Decode CODE$, No.s of IP's stored in VI() array.
 6480  SUMX=0:SUMY=0:SUM2X=0:SUM2Y=0
 6500 REM - - - - - - - - - - - - - -
 6520  FORI%=1TO NI:IF SOR$="S" THEN IX=XI(VI(I%)):IY=YI(VI(I%)): ELSE IX=RXI(VI
(I%)):IY=RYI(VI(I%))
 6540 SUMX=SUMX+IX:SUMY=SUMY+IY
 6560 SUM2X=SUM2X+IX^2:SUM2Y=SUM2Y+IY^2:NEXT
 6580 REM - - - - - - - - - - - - - -
 6600 MEANX=SUMX/NI:MEANY=SUMY/NI:MEANZ=0:IF SOR$="R" THEN MEANZ=RZCOK
 6620 VARX=(SUM2X/NI -MEANX^2):IF VARX<=0 THEN SDX=0 ELSE SDX=SQR(SUM2X/NI -MEAN
X^2)
 6640 VARY=(SUM2Y/NI -MEANY^2):IF VARY<=0 THEN SDY=0 ELSE SDY=SQR(SUM2Y/NI -MEAN
Y^2)
 6660 SDZ=0
 6680 REM
 6700 REM - - - - - - - - All Fix types - - - - - - - - - - - - - - -
 6740 IF FXTYP=1 THEN FXTYP$="1 Range No Fix": CODE$="11111111"
 6760 IF FXTYP=2 THEN FXTYP$="Manual.....": CODE$=CODE$
 6780 IF FXTYP=3 THEN FXTYP$="Cokhat.....": CODE$=CODE$
 6800 IF FXTYP=4 THEN FXTYP$="Matrix.....": CODE$=CODE$
 6820 IF FXTYP=8 THEN FXTYP$="No fix Reqd...": CODE$="88888888"
 6840 IF FXTYP=9 THEN FXTYP$="Not fixed.....": CODE$="99999999"
 6850 IF FXTYP=1 OR FXTYP=8 OR FXTYP=9 THEN MEANX=999999:MEANY=999999:MEANZ=9999
99:SDX=999999:SDY=SDX:SDZ=SDX:IF SOR$="R" THEN MEANZ=ZRIP:SDZ=0
 6860 CODE=VAL(CODE$)
 6880 IF PRNTR=1 THEN VDU2
 6900 PRINT:IF SOR$="R" THEN RFXTYP$=FXTYP$:RFXTYP=FXTYP:PRINT;" REMOTE FIX by:-
";FXTYP$;: ELSE SFXTYP$=FXTYP$:SFXTYP=FXTYP:PRINT;" SHIP FIX by:-";SFXTYP$;
 6920 PRINT;" Fxtyp: ";FXTYP;" IPs/CODE: ";CODE$
 6940 @%=&020009
 6960 PRINT;" X:";MEANX;" Y:";MEANY;" Z:";MEANZ;" Error:- ";(SDX+SDY+SDZ)/3:VDU3
 6980 PROCscal(XMIN,MEANY):XP1=XP:YP1=YP:PROCscal(XMAX,MEANY):XP2=XP:YP2=YP:PROC
scal(MEANX,YMIN):XP3=XP:YP3=YP:PROCscal(MEANX,YMAX):XP4=XP:YP4=YP:VDU4
 7000 GCOL 0,3:VDU5:MOVE XP1,YP1:PLOT 21,XP2,YP2:MOVE  XP3,YP3:PLOT 21,XP4,YP4:VD
U4
 7020 PROClation:IF SOR$="S" THEN SXDEG=XDEG:SYDEG=YDEG:ELSE RXDEG=XDEG:RYDEG=YD
EG
 7040 IF SOR$="S" THEN SCODE=CODE:SFIX$="Y":XMEAN=MEANX:YMEAN=MEANY:ZMEAN=MEANZ:
SDZ=Z(9):X(7)=XMEAN:Y(7)=YMEAN:Z(7)=ZMEAN:X(9)=SDX:Y(9)=SDY:Z(9)=SDZ
 7060 IF SOR$="R" THEN RCODE=CODE:RFIX$="Y":RXMEAN=MEANX:RYMEAN=MEANY:RZMEAN=MEA
NZ:RXSD=SDX:RYSD=SDY:RZSD=SDZ:X(8)=RXMEAN:Y(8)=RYMEAN:Z(8)=RZMEAN:X(10)=RXSD:Y(1
0)=RYSD:Z(10)=RZSD
 7080 VDU5:MOVE "PIXPP",x1=210:x2=390:y0=800:y9=840:IF SOR$="R" THEN y0=760:y9=800
```

```
7100  PROCban(x1,x2,y0,y9,bn$,1): REM Banner Fixed.
 7120 REM - - - - -Position Fixed - - -
 7140 IF AUTPROC$="Y" THEN PROCcomp:GOTO 7280
 7160 PROCqon:PRINT;"RTN> Compare/Predict";:INPUT"N>  to Skip          ";Q$:PROCq
of:IF Q$<>"N" THEN PROCcomp
 7180 IF QREM$="Y" THEN PROCqon:PRINT;"<X> ReFix the Device";:INPUT"<RTN>  to pr
oceed  ";Q$:PROCqof:IF Q$="X" THEN GCOL 0,0:VDU5:MOVE XP1,YP1:PLOT 21,XP2,YP2:MO
VE XP3,YP3:PLOT 21,XP4,YP4:GCOL 0,3:VDU4:REM Blacks out last fix.
 7200 IF Q$="X" AND SOR$="S"THEN ban$="      ":x1=210:x2=390:y0=800:y9=840:SFIX$=
"N":ELSE IF Q$="X" AND SOR$="R" THEN y0=760:y9=800:RFIX$="N"
 7220 IF Q$="X" THEN PROCban(x1,x2,y0,y9,ban$,1):GOTO 5700:REM De-flag FIXED ban
ner and goto PreFix Menu, SOR$ unchanged.
 7260 REM
 7280 IF AUTPROC$="Y" AND QREM$="Y" AND RFIX$="N" THEN DUM$="R":GOTO 7520
 7300 IF AUTPROC$="Y" THEN DUM$="S":GOTO 7660
 7310 REM - - - - - - - - - - - - - - POST FIX MENU - - - - - - - - - - - - - - -
 7320 PROCqon:PRINT;"**********************";"POST FIX MENU          Select:
    ";
 7340 IF QREM$="Y" AND RFIX$="N" AND SFIX$="Y" THEN PRINT;"RTN> Fix Remote      "
;"S>   Save to Disk    ";:ELSE PRINT;"RTN> Save to Disk    ";
 7350 IF QREM$="Y" THEN PRINT;"R>   Fix Remote      ";"XR>Retype RemoteData";
 7360 PRINT;"F>   Redo whole Fix ";:PRINT;"C>   Compare/Predict";:PRINT;"M>   EX
IT- Main Menu"
 7380 INPUT DUM$:PROCqof
 7383 IF QREM$="Y" AND RFIX$="N" AND SFIX$="Y" AND DUM$="" THEN DUM$="R": ELSE I
F DUM$="" THEN DUM$="S"
 7400 REM - - - - - - - - - - - - - -
 7420 IF DUM$="M" THEN ENDPROC
 7440 IF DUM$="S" AND SFIX$="N" THEN PROCX:PRINT"***SHIP NOT FIXED***":PROCqon:P
RINT;"<RTN> to fix ship";:PRINT;"<C> to continue save";:INPUT Q$:PROCqof:IF Q$<>
"C" THEN SOR$="S": GOTO 5720
 7500 IF DUM$="S" AND RFIX$="N" AND QREM$="Y" THEN PROCX:PRINT;"**REMOTE NOT FIX
ED**";:PROCqon:PRINT;"<RTN> to fix remote";:PRINT;"<C> to continue save";:INPUT
Q$:PROCqof:IF Q$<>"C" THEN SOR$="R":DUM$="R"
 7510 IF DUM$="S" THEN GOTO 7660
 7520 IF DUM$="R" THEN SOR$="R":PROCcalc:GOTO 5700
 7540 REM - - - - - - - - - - - - - - - -
 7560 IF DUM$="C" AND QREM$="Y" THEN SOR$="S":PROCqon:PRINT;"RTN>  C&P Ship
 ";:INPUT"R>  or Remote      ";Q$:PROCqof:IF Q$<>"" THEN SOR$="****":SOR$="R"
 7580 IF DUM$="C" THEN PROCcomp:GOTO 7320
 7600 IF DUM$="F" THEN SFIX$="N":RFIX$="N":PROCrelabel:GOTO 5640:REM
 7620 IF DUM$="F" THEN PROCqon:PRINT;"Hit key";:COLOUR 129:PRINT;"f6";:COLOUR 13
0:PRINT;" to redo fix after stop":PROCqof:STOP
 7660 REM - - - - - - - -SAVE FIX - - - - - - - - -
 7678 IF DUM$<>"S" THEN PROCX:STOP:REM Error!!!!!!
 7680 PROCsave
 7760 VDU3:ENDPROC
 7780 REM=============================================================================
```

```
 7800 DEF PROCcomp:REM calculate  hdng. & speed from last fix and predicted rang
es from beacons and ship/remote
 7820 @%=&020209
 7823 IF AUTPROC$="Y" THEN GOTO 7828: ELSE  PRINT;"Data  file is:  ";OF$:IF OF$="
undefnd" THEN PROCrenam
 7828 IF OF$="undefnd" THEN GOTO 8140: REM predict only.
 7830 PROCdatdri:OCHN=OPENIN (DR$+OF$)
 7832 PTR#OCHN=27:INPUT#OCHN,PTRINT,ENDPTR,NFIXES:IF NFIXES=0 THEN PROCprevfile:
IF PREVOF$="undefnd" THEN GOTO 8140
 7834 NFBEOF=1:PROCreadFfix:CLOSE#OCHN:OXMEAN=FXMEAN:OYMEAN=FYMEAN:ROXMEAN=FRXME
AN:ROYMEAN=FRYMEAN:OLTIME=FTIME:OLDAY=FDAY:REM replace OL's by F's in code below
******.
 7850 IF SOR$="S" AND OXMEAN>900000 THEN:PRINT;"Previous fix No Good":VDU3:GOTO
8140:ELSE IF SOR$="R" AND ROXMEAN>900000 THEN:PRINT;"Previous fix No Good":VDU3:
GOTO 8140
 7860 IF PRNTR=1 THEN VDU2
 7920 IF SOR$="S" THEN DX=XMEAN-OXMEAN:DY=YMEAN-OYMEAN:DZ=ZMEAN-OZMEAN:ELSE DX=R
XMEAN-ROXMEAN:DY=RYMEAN-ROYMEAN:DZ=RZMEAN-ROZMEAN
 7940 IF PRNTR=1 THEN VDU2
 7960 @%=&00020009:PRINT;"   FROM LAST FIX @:";OLTIME;"/";OLDAY;" X:";XMEAN-DX;"
 Y:";YMEAN-DY;" Z:";ZMEAN-DZ
 7980 IF DY=0 THEN BRNG=180-SGN(DX)*90
 8000 DH2= (DX^2 + DY^2):IF DY=0 THEN GOTO 8020: ELSE BRNG=DEG(ATN(DX/DY))
 8020 IF DY<=0 THEN BRNG=180+BRNG:ELSE IF DX<0 AND DY>=0 THEN BRNG=360+BRNG
 8040 IF DH2=0 THEN DH=0 ELSE DH=SQR(DH2)
 8060 NH=INT(NTIME):NM=(NTIME-NH)*100:NT=(NDAY*24*60)+(NH*60)+NM:OH=INT(OLTIME):
OM=(OLTIME-OH)*100:OT=(OLDAY*24*60)+(OH*60)+OM:DTIME=NT-OT:REM in minutes
 8080 HMN=INT(NTIME/100):MN=NTIME-(100*HMN):NT=HMN*60+MN:HMO=INT(OLTIME/100):MO=
OLTIME-(100*HMO):OT=HMO*60+MO:DTIME=NT-OT:IF DTIME<0 THEN DTIME=DTIME+(24*60):EL
SE IF DTIME=0 THEN DTIME=0.1
 8100 IF PRNTR=1 THEN VDU2
 8120 PRINT"Bearing:";BRNG;", Distance:";DH;" m, ";DTIME;" min ago";:@%=&0002020
9:PRINT;" Speed:";DH*(60/1850)/DTIME;" kts":VDU3: IF DH/(DTIME*60)>330 THEN PROC
X:COLOUR 129:PRINT;"Supersonic!!!!!":COLOUR 131
 8140 REM - - - - - - Predict ranges from beacons - - - -- - - -
 8160 IF PRNTR=1 THEN VDU2
 8180 PRINT;"   PREDICTED RANGES from chosen fix position":VDU3
 8200 IF PRNTR=1 THEN VDU2:IF SOR$="S" THEN PRINT;"Beacon  2WT+del. s sl.    ";:
ELSE PRINT;"Beacon   1WT+dly s rbs    ";
 8220 PRINT;"      Slant m.          Horiz. m."
 8240 FOR I=1 TO NBCNS:DHSB=(X(I)-XMEAN)^2+(Y(I)-YMEAN)^2:IF DHSB>0 THEN DHSB=SQ
R(DHSB)
 8260 DSSB=(DHSB^2 +(Z(I)-ZMEAN)^2)^0.5:T2SB=DSSB*2/SVEL
 8280 IF SOR$="R" THEN DHRB=(X(I)-RXMEAN)^2+(Y(I)-RYMEAN)^2:IF DHRB<>0 THEN DHRB
=SQR(DHRB)
 8300 IF SOR$="R" THEN DSRB=(DHRB^2 +(Z(I)-RZMEAN)^2)^0.5:T1RBS=(DSRB+DSSB)/SVEL

 8320 IF PRNTR=1 THEN VDU2
 8340 @%=&020207:IF SOR$="S" THEN PRINT BECOL$(I),TAB(15)T2SB+TDEL(I);:@%=&02000
9:PRINT TAB(30) DSSB,TAB(50)DHSB:VDU3
 8360 @%=&020207:IF SOR$="R" THEN PRINT BECOL$(I),TAB(15)T1RBS+TDEL(I);:@%=&0200
09:PRINT TAB(30) DSRB, TAB(50)DHRB:VDU3
 8380 NEXT I
 8400 x2=390:x9=630:y0=800:y9=840:bn$="& COMPD": IF SOR$="R" THEN y0=760:y9=800
 8420 PROCban(x2,x9,y0,y9,bn$,1):IF SOR$<>"R" THEN GOTO 8520:REM Endproc
 8440 DHSR=(RXMEAN-XMEAN)^2 + (RYMEAN-YMEAN)^2:IF DHSR<>0 THEN DHSR=SQR(DHSR)
 8460 DSSR=(DHSR^2 +(RZMEAN-ZMEAN)^2):IF DSSR<>0 THEN  DSSR=SQR(ABS(DSSR)):T1RS=D
SSR/SVEL
 8480 IF PRNTR=1 THEN VDU2
 8500 @%=&00020009:PRINT;"Distance ship to remote =";DHSR;"m.  horiz,  ";DSSR;"
m. slant,   ";:@%=&00020209:PRINT;T1RS;" s. 1WT slant.":VDU3
 8520 ENDPROC
 8540 REM=============================================================
```

```
 8540 REM=========================================================================
 8620 DEF PROCsave
 8640 IF DATIN$="N" THEN PROCX:PRINT"***No data to save****":GOTO 8840
 8660 IF AUTPROC$="Y" THEN DUM$="":GOTO 8740
 8680 PROCqon:PRINT;"Output file:        ";OF$:PRINT;"RTN> if OK          ";:PRI
NT;"NEW> create new file";"OLD> extend old file";"CAT> catalog disk   ";"HDR> Re
ad FDF header";:PROCqof:INPUT DUM$
 8700 IFLEN(DUM$) <>3 AND DUM$<>"" THEN VDU7:PRINT;" Please retype":GOTO 8680
 8720 IF DUM$="CAT" THEN OSCLI"DR."+STR$(D%):OSCLI "CAT": GOTO 8680
 8730 IF DUM$="HDR" THEN OSCLI"DR."+STR$(D%):PROChdrd:PROCtransin :GOTO 8680
 8740 XMEAN=X(7):YMEAN=Y(7):ZMEAN=Z(7):RXMEAN=X(8):RYMEAN=Y(8):RZMEAN=Z(8):XSD=X
(9):YSD=Y(9):ZSD=Z(9):RXSD=X(10):RYSD=Y(10):RZSD=Z(10)
 8750 IF DUM$="" THEN OF$=OF$
 8760 IF DUM$="NEW" THEN PROCnew ELSE IF DUM$="OLD" THEN PROCold
 8770 IF OF$="" OR OF$="undefnd" THEN PROCX:GOTO 8680
 8780 PROCdatwrit
 8800 PROCtransout:REM This is needed to register the latest data on the data di
sk in case of program shutdown or crash.
 8820 x0=0:x9=600:y0=720:y9=760:ban$="DATA SAVED TO DISK":PROCban(x0,x9,y0,y9,ba
n$,1)
 8822 save$="Y":IF PRNTR=1 THEN VDU2
 8824 PRINT;"==============================":PRINT;"Datafile:";OF$: PRINT;"SHIP
 FIX: ";SFXTYP$;" Type: ";SFXTYP;" IP Code: ";SCODE:IF QREM$="Y" THEN PRINT;"REM
OTE FIX: ";RFXTYP$ ; " Type: ";RFXTYP;" IP Code: ";RCODE
 8826 PRINT:VDU3
 8828 PROCX:PRINT;"WELL DONE...Another Fix Saved!"
 8840 ENDPROC
 8860 REM=========================================================================
 8880 DEF PROChdrd:REM Read in header of a disk data file and list the data to s
creen or printer.
 8890 PRINT;"Current FDF name:";OF$
 8892 PROCqon:PRINT;"<RTN>   If OK       ";:PRINT;"OR Newname <7 chars>";:INPUT
Q$:PROCqof:IF Q$="" THEN IPFILE$=OF$: ELSE IF LEN(Q$)<>7 THEN PROCX:GOTO 8892
 8894 IF Q$<>"" THEN IPFILE$=Q$
 8896 IF IPFILE$="undefnd" THENGOTO 9160
 8920 PROCqon:INPUT"Please give drive No.";DRIP$:PROCqof:IF VAL(DRIP$)>3 THEN PR
OCX:GOTO 8920
 8940 PROCqof:Q$="N":IF PRNTR=1 THEN PROCqon:INPUT"Print Out of Header <Y> or <N
>   ";Q$ :PROCqof
 8960 IF Q$="Y" THEN VDU2
 8980 ON ERROR PROCdiskIOerror
 9000 ICHN=OPENIN (":"+DRIP$+"."+IPFILE$):PTR#ICHN=0:INPUT#ICHN,DUM$,PREVOF$,DTY
PE$:PTR#ICHN=27:INPUT#ICHN,PTRINT,ENDPTR,NFIXES,DAY0,TM0,DAYEND,TMEND
 9020 PRINT "Filename:";DUM$;" Pr File:";PREVOF$;" Dtype:";DTYPE$;" PTRINT:";PTR
INT;",ENDPTR:";ENDPTR;" No fixes:";NFIXES;" First fix:";DAY0;"/";TM0;" Latest fi
x:";DAYEND;"/";TMEND
 9040 HDLEN=ENDPTR-(NFIXES*PTRINT)
 9060 VDU3:VDU6:CLOSE#ICHN
 9080 PROCqon:INPUT"Do you wish to      display fix data    <Y> or <N>
";QQ$:PROCqof:IF QQ$<>"Y" THEN GOTO 9140 ELSE FOR FF=1 TO NFIXES
 9100 IF Q$="Y" THEN PRINT:VDU2:VDU21:@%=&20209
 9120 PROCreadlfix(ICHN,FF): NEXT FF:VDU3:VDU6
 9140 CLOSE#0:ON ERROR PROCerror
 9160 ENDPROC
 9180 REM=========================================================================
```

```
 9180 REM=======================================================================
 9200 DEF PROCreadlfix (ICHN,NFIX): REM For reading data for ONE fix, number NFI
X on file on channel ICHN
 9220 REM At present this Proc can read from files with data in old or new forma
ts. Edit when old format not required.
 9240 ICHN=OPENIN (":"+DRIP$+"."+IPFILE$):PTR#ICHN=27:INPUT#ICHN,PTRINT:REM Do t
his in case O/P PTRINT is different.
 9260 HDLEN=HDLEN:PTRINT=PTRINT:PTR#ICHN=HDLEN + ((NFIX-1)*PTRINT)
 9280 INPUT#ICHN,NFIX,NDAY,NTIME,R(1),R(2),R(3),R(4),R(5),R(6),R(7),RR(1),RR(2),
RR(3),RR(4),RR(5),RR(6),RR(7)
 9300 IF PRNTR=1 THEN VDU2
 9320 PRINT"NFIX=";NFIX;"NDAY=";NDAY;" NTIME=";NTIME;" R(1)=";R(1);" R(2)=";R(2)
;" R(3)=";R(3);" R(4)="; R(4);" R(5)=";R(5);" R(6)=";R(6);" R(7)=";R(7);
 9340PRINT;" RR(1)=";RR(1);" RR(2)=";RR(2);" RR(3)=";RR(3);" RR(4)=";RR(4);" RR(
5)=";RR(5);" RR(6)=";RR(6);" RR(7)=";RR(7)
 9360 IF PTRINT=186 THEN INPUT#ICHN,SCODE,SXDEG,SYDEG,ZMEAN,XSD,YSD,ZSD,RCODE,RX
DEG,RYDEG,RZMEAN,RXSD,RYSD,RZSD
 9380 IF PTRINT=198 THEN INPUT#ICHN,SCODE,SFXTYP,SXDEG,SYDEG,ZMEAN,XSD,YSD,ZSD,R
CODE,RFXTYP,RXDEG,RYDEG,RZMEAN,RXSD,RYSD,RZSD
 9400 SCODE$=STR$(SCODE):RCODE$=STR$(RCODE):DAY$=STR$(NDAY):NTIME$=STR$(NTIME)
 9420 IF ABS(SXDEG)>180 AND SXDEG<999998 THEN PROCmetdeg(SXDEG,SYDEG):SXDEG=XDEG
:SYDEG=YDEG:IF ABS(RXDEG)>180 AND RXDEG<999998 THEN  PROCmetdeg(RXDEG,RYDEG):RXD
EG=XDEG:RYDEG=YDEG: REM if in metres conv. to degs.*******REMOVE***************
 9440 IF PTRINT=186 THEN PRINT" SCODE=";SCODE;" SXDEG=";SXDEG;" SYDEG=";SYDEG;"
ZMEAN=";ZMEAN;" XSD=";XSD;" YSD=";YSD;" ZSD=";ZSD;" RCODE=";RCODE;" RXDEG=";RXDE
G;" RYDEG=";RYDEG;" RZMEAN=";RZMEAN;" RXSD=";RXSD;" RYSD=";RYSD;"RZSD=";RZSD
 9460 IF PTRINT=198 THEN  PRINT" SCODE=";SCODE;"SFXTYP=";SFXTYP;" SXDEG=";SXDEG;
" SYDEG=";SYDEG;" ZMEAN=";ZMEAN;" XSD=";XSD;" YSD=";YSD;" ZSD=";ZSD;
 9480 IF PTRINT=198 THEN  PRINT;" RCODE=";RCODE;" RFXTYP=";RFXTYP;" RXDEG=";RXDE
G;" RYDEG=";RYDEG;" RZMEAN=";RZMEAN;" RXSD=";RXSD;" RYSD=";RYSD;" RZSD=";RZSD
 9500 PRINT;"PTR now at: ";PTR#ICHN:PRINT:VDU3
 9520 CLOSE#ICHN:REPROC$="Y":ENDPROC
 9540 REM=======================================================================
```

```
 9540 REM=========================================================================
 9560 DEF PROCcalc : REM  for calculating range circle intersections
 9580 IF DATIN$<>"Y" THEN PRINT;"No data to calc":ENDPROC
 9600 IF PRNTR=1 THEN VDU2:PRINT
 9620 SORR$=SOR$
 9640IF SORR$="R" THEN NRNGS=NBCNS+1:PRINT;"***REMOTE***":RBSTFX$="MANUAL": ELSE
 NRNGS=NBCNS:PRINT;"***SHIP***":SBSTFX$="MANUAL"
 9660 GCOL 0,1 :VDU3
 9680 NGIPS=0: NGRANS=0: FOR I%=1TO MNRS :REM NRNGS is max poss No of ranges for
 particular device REMT = SHIP+1
 9700 IF SORR$="S" AND  R(I%)<1 THEN GOTO 9960
 9720 IF SORR$="S" AND I% =MNRS THEN GOTO 9960:REM Exclude REMT:SHIP range for S
hip
 9740 IF SORR$="S" THEN NGRANS=NGRANS+1
 9760 IF SORR$="R" AND RR(I%)<1 THEN GOTO 9960
 9780 IF SORR$="R" THEN NGRANS=NGRANS+1
 9800 IF I%>MNRS-1 THEN GOTO 9960: REM Skip to stop J% =>MNRS.
 9820 FOR J%=I%+1 TO MNRS:IF SORR$="S" AND R(J%)<1 THEN GOTO 9940:ELSE IF SORR$=
"R" AND RR(J%)<1 THEN GOTO 9940
 9840 IF SORR$="S" AND J%=MNRS THEN GOTO 9940
 9860 NGIPS=NGIPS+1:REM No.s of Good: IPs.
 9880 CHEK=0
 9900 IF SORR$="S" THEN PROCints(X(I%),Y(I%),R(I%),X(J%),Y(J%),R(J%)):XI(NGIPS)=
XI1:YI(NGIPS)=YI1:NGIPS=NGIPS+1:XI(NGIPS)=XI2:YI(NGIPS)=YI2:IF CHEK=1 THEN VDU2:
PRINT;"IPs:";NGIPS-1;",";NGIPS;" Beacons: ";I%;", ";J%:VDU3
 9920 IF SORR$="R" THEN PROCints(X(I%),Y(I%),RR(I%),X(J%),Y(J%),RR(J%)):RXI(NGIP
S)=XI1:RYI(NGIPS)=YI1:NGIPS=NGIPS+1:RXI(NGIPS)=XI2:RYI(NGIPS)=YI2:IF CHEK=1 THEN
 VDU2:PRINT;"IPs:";NGIPS-1;",";NGIPS;" Beacons: ";I%;", ";J%:VDU3
 9940 NEXT J%:REM
 9960 NEXT I%: IF SORR$="S" THEN SNGIPS=NGIPS:ELSE RNGIPS=NGIPS
 9980 PIPS=0:FOR I=1TO NRNGS:PIPS=PIPS+2*(I-1):NEXT:IF PRNTR=1 THEN VDU2:PRINT;N
GIPS;" good IPs out of ";PIPS;" possibles.":VDU3
10000 PROCcoknat:IF SORR$="S" THEN SIPCAL$="Y":ELSE RIPCAL$="Y"
10020 PROCMATFIX:IF SORR$="R" THEN GOTO 10160
10040 SXAUT=SXCOK:SYAUT=SYCOK:SZAUT=SZCOK:SSDAUT=SSDCOK:IF SSDMAT<SSDCOK THEN SX
AUT=SXMAT:SYAUT=SYMAT:SZAUT=SZMAT:SSDAUT=SSDMAT:SBSTFX$="MATRIX":ELSE IF SSDCOK<
500 THEN SBSTFX$="COKHAT"
10060 IF PRNTR=1 THEN VDU2
10080 PRINT;"SHIP  best Autofix by:";SBSTFX$:VDU3
10100 IF SSDAUT < 500 THEN X(7)=SXAUT:Y(7)=SYAUT:Z(7)=SZAUT:X(9)=SSDAUT:Y(9)=X(9
):Z(9)=0:REM If best fix good then adopt that as ship position for the purpose o
f calculating remote fixes by cokhat and matrix. IS THIS NEEDED???
10120 REM
10140 GOTO 10220: REM implies SORR$="S", QREM$<>"Y" - - - - - - - - - - - - - -
10160 RXAUT=RXCOK:RYAUT=RYCOK:RZAUT=RZCOK:RSDAUT=RSDCOK:IF RSDMAT<RSDCOK THEN RX
AUT=RXMAT:RYAUT=RYMAT:RZAUT=RZMAT:RSDAUT=RSDMAT:RBSTFX$="MATRIX":ELSE IF RSDCOK<
500 THEN RBSTFX$="COKHAT"
10180 IF PRNTR=1 THEN VDU2:PRINT;"REMOTE best Autofix by:";RBSTFX$:VDU3
10200 IF RSDAUT < 50000 THEN X(8)=RXAUT:Y(8)=RYAUT:Z(8)=RZAUT:X(10)=RSDAUT:Y(10)
=X(10):Z(10)=0:REM If best fix good then adopt that as remote position until a n
ew one fixed by user.
10220 ENDPROC
10240 REM=========================================================================
```

```
10240 REM=======================================================================
10260 DEF PROCints(XA,YA,RA,XB,YB,RB)
10280 REM Procedure ints calculates the coordinates of the 2 intersections (XI1,
YI1)&(XI2,YI2) of 2 circles given the coordinates of the circle centres (XA,YA)&
(XB,YB) and their radii (RA & RB).
10300 DX=XB-XA:IF DX=0 THEN DX=0.1:
10320 DY=YB-YA:IF DY=0 THEN DY=0.1:
10340 C=SQR(DX^2 + DY^2)
10360 IF RA=0 OR RB=0 OR C=0 THEN XI1=999999:YI1=999999:XI2=999999:YI2=999999:GO
TO 10560
10380 IF DX=0 THEN GAM0=PI/2 :GOTO 10420
10400 GAM0=ATN(DY/DX)
10420 GAM=GAM0:IF DX<=0 THEN GAM=GAM+PI
10440 IF DX>0 AND DY<0 THEN GAM=GAM+2*PI
10460 ARG=(RA^2 +C^2 -RB^2)/(2*RA*C)
10480 IF ARG/SGN(ARG) > 1 THEN ARG = 1*SGN(ARG)
10500 ALFA=ACS(ARG)
10520 IF RA+RB<C THEN  PROCshort ELSE IF RA>RB+C OR RB>RA+C THEN PROClong ELSE P
ROCroot
10540 XI1=XA+DX1:YI1=YA+DY1: XI2=XA+DX2:YI2=YA+DY2
10560 ENDPROC
10580 REM=======================================================================
10600 DEF PROCroot
10620 DX1=RA*COS(GAM+ALFA):DY1=RA*SIN(GAM+ALFA):DX2=RA*COS(GAM-ALFA):DY2=RA*SIN(
GAM-ALFA)
10640 ENDPROC
10660 REM=======================================================================
10680 DEF PROClong
10700 IF RB>C+RA THEN DX1=0.5*(C-RA-RB)*COS(GAM): DY1=0.5*(C-RA-RB)*SIN(GAM)
10720 IF RB<=C+RA THEN DX1=0.5*(RA+C+RB)*COS(GAM): DY1=0.5*(C+RA+RB)*SIN(GAM)
10740 DX2=DX1:DY2=DY1
10760 ENDPROC
10780 REM=======================================================================
10800 DEF PROCshort
10820 DX1=0.5*(RA+C-RB)*COS(GAM) : DY1=0.5*(RA+C-RB)*SIN(GAM) : DX2=DX1:DY2=DY1
10840 ENDPROC
10860 REM=======================================================================
10880 DEF PROCtransin :PRINT;"Reading from TRANS":CLOSE # 0
10900 ON ERROR PROCdiskIOerror
10920 PROCdatdri:ICHN=OPENIN(DR$+"trans"):INPUT #ICHN,SCODE,RCODE,OF$,PREVOF$,BE
EP$,NDAY,NTIME,NBCNS,NRANS,XMIN,XMAX,YMIN,YMAX,X0,X9
10940 IF CHEK=1 THEN  PRINT SCODE,RCODE,OF$,PREVOF$,BEEP$,NDAY,NTIME,NBCNS,NRANS
,XMIN,XMAX,YMIN,YMAX,X0,X9
10960 INPUT #ICHN,Y0,Y9,XLEN,YLEN,XRAN,YRAN,XSCA,YSCA,XMEAN,YMEAN,RXMEAN,RYMEAN
10980 IF CHEK=1 THEN  PRINT Y0,Y9,XLEN,YLEN,XRAN,YRAN,XSCA,YSCA
11000 INPUT#ICHN,OXMEAN,OYMEAN,OZMEAN,ROXMEAN,ROYMEAN,ROZMEAN,OLTIME,OLDAY,CHEK,
QQREM,PRNTR
11020 INPUT#ICHN,SHINT1,SHINT2,SHINT3,RINT1,RINT2,RINT3
11040 FOR I%=1 TO 14:INPUT #ICHN,X(I%),Y(I%),Z(I%):NEXT
11060 FOR I%=1 TO 7:INPUT #ICHN,R(I%),RR(I%):NEXT
11080 XSD=X(9):YSD=Y(9):ZSD=Z(9):RXSD=X(10):RYSD=Y(10):RZSD=Z(10)
11100 CLOSE #0:ON ERROR PROCerror
11120 ENDPROC
11140 REM=======================================================================
```

```
11140 REM=========================================================================
11160 DEF PROCtransout :PRINT;"Writing to TRANS":CLOSE # 0
11180 ON ERROR CLOSE#0:PROCdiskIOerror
11200 PROCdatdri:OCHN=OPENOUT(DR$+"trans"):PRINT #OCHN,SCODE,RCODE,OF$,PREVOF$,B
EEP$,NDAY,NTIME,NBCNS,NRANS,XMIN,XMAX,YMIN,YMAX,X0,X9
11220 PRINT #OCHN,Y0,Y9,XLEN,YLEN,XRAN,YRAN,XSCA,YSCA,XMEAN,YMEAN,RXMEAN,RYMEAN
11240 PRINT#OCHN,OXMEAN,OYMEAN,OZMEAN,ROXMEAN,ROYMEAN,ROZMEAN,OLTIME,OLDAY,CHEK,
QQREM,PRNTR
11260 PRINT#OCHN,SHINT1,SHINT2,SHINT3,RINT1,RINT2,RINT3
11280 FOR I%=1 TO 14:PRINT #OCHN,X(I%),Y(I%),Z(I%):NEXT
11300 FOR I%=1 TO 7:PRINT #OCHN,R(I%),RR(I%):NEXT
11320 CLOSE #0:ON ERROR PROCerror
11340 ENDPROC
11360 REM=========================================================================
11380 DEF PROCban(x0,x9,y0,y9,ban$,NCOL):REM Draw box colour NCOL and banner.
11400 GCOL 0,NCOL:MOVE x0,y9:DRAW x0,y0:PLOT 85,x9,y9:MOVE x9,y0:PLOT 85,x0,y0:V
DU5:GCOL 0,3:MOVE x0,(y0+y9)/2 +15:PRINT;ban$:VDU4
11420 ENDPROC
11440 REM=========================================================================
11460 DEF PROCdatdri:REM Sets drive No. for writing data.
11480 PRINT;"Drive No.:";D%;
11500 IF D%>3 OR D%<0 THEN PROCqon:PRINT" invalid":PRINT:INPUT"Please give Drive
 No";D%:PROCqof:ELSE PRINT
11520 DR$=":"+STR$(D%)+".":ENDPROC:REM D% is set by user in PROCAC1
11540 REM=========================================================================
11560 DEF PROCerror:ON ERROR OFF
11580 IF ERR=17 THEN PRINT;"Escape"
11600 VDU6:VDU3:VDU4:CLOSE #0
11620 IF ERR<>17 THEN PRINT;"*?*!*?aaarrgghh!!!*?":REPORT: PRINT"at line";ERL:PR
OCX
11640 PROCstop
11660 ENDPROC
11680 REM=========================================================================
11700 DEF PROCdiskIOerror:REM for trapping ESCAPES hit while disk file is open
11720 VDU3:VDU6:ON ERROR PROCerror
11740 REPORT:PRINT;"Error during disk I/O":PROCqon:INPUT"Disk file open      esc
ape may corrupt  <XXX> to escape     <RTN> to continue    ";Q$:PROCqof:IF Q$="XXX
" THEN CLOSE#0:PROCerror:REM Stops prog.
11760 ENDPROC
```

```
12020 REM=======================================================================
12040 DEF PROCnew :REM set up a new empty FDF file
12080 NFIXES=0:HDLEN=(3*9)+(7*6):PTRINT=(33*6): REM This reflects file structure
 (numeric=6, char. =LENGTH+2 bytes)
12100 ENDPTR=HDLEN:EFPTR=HDLEN+(PTRINT*FILEN)
12120 PROCqon:INPUT"Please give a name for the New File   (7 characters)       ";
OF$:PROCqof:VDU3:IF LEN(OF$)<>7 THEN PROCX:GOTO 12120
12140 PRINT;"Previous filename= ";PREVOF$
12160 PROCqon:INPUT"Please give the name (7 chars) of the previous file or press
 <RTN>";Q$:PROCqof:IF Q$<>"" THEN PREVOF$=Q$:IFLEN(Q$)<>7 THEN PROCX:GOTO 12160
12180 PRINT;"Creating a new data file:";OF$;" for";FILEN;" fixes.":VDU3
12200 ON ERROR PROCdiskIOerror
12220 CLOSE #0:PROCdatdri:OCHN=OPENOUT(DR$+OF$):PRINT "PTR=";PTR#OCHN:PRINT #OCH
N,OF$,PREVOF$,".......",PTRINT,ENDPTR,0.,0.,0.,0.,0.
12240  FOR I=1 TO FILEN:FOR J=1 TO 33: PRINT #OCHN, 888888.0:NEXT J
12260 PRINT;"I=";I
12280 NEXT I:IF PTR#OCHN<>HDLEN+(FILEN*PTRINT) THEN PRINT;"PTR @:";PTR#OCHN;" <>
 ";HDLEN+(FILEN*PTRINT): CLOSE#0:PRINT"!!!File Check Error!!!":PROCX:STOP:ELSE C
LOSE # OCHN:PRINT;"File written to Disk"
12300 ON ERROR PROCerror
12320 ENDPROC
12340 REM=======================================================================
12360 DEF PROCold
12380 PROCqon:INPUT"Please give the name(7 characters) of   the Old File you
wish to add to:     ";OF$:PROCqof:IFLEN(OF$)<>7 THEN PROCX:GOTO 12380
12400 ENDPROC
12420 REM=======================================================================
12440 DEF PROCdatwrit:HDLEN=(3*9)+(7*6)
12460 ON ERROR PROCdiskIOerror
12470 IF OF$="undefnd" THEN PROCrenam
12480 CLOSE#0:PROCdatdri:OCHN=OPENUP(DR$+OF$):PTR#OCHN=0:INPUT#OCHN,DUM$,PREVOF$
,DTYPE$:PTR#OCHN=27:INPUT#OCHN,PTRINT,ENDPTR,NFIXES,DAY0,TM0,DAYEND,TMEND
12500 IF NFIXES<FILEN THEN GOTO 12660: ELSE CLOSE#0:PRINT;"File:";DUM$;" Full!":
PROCX
12520 REM - - - - - Disk File Full: Nominate or Create Alternative - - -
12540 PROCqon:INPUT"Type name of existing FDF file (7 chars) or <RTN> to create
new fix datafile.";DUM$:PROCqof
12560 IF DUM$=""THEN PREVOF$=OF$:PROCnew: ELSEIF LEN(DUM$)<>7 THEN PROCX:GOTO 12
540
12580  OF$=DUM$:PRINT;"Drive No.:";D%;:PROCqon:INPUT"<RTN> if OK or type alterna
tive";Q$:PROCqof:IF Q$<>"" THEN D%=VAL(Q$):Z(14)=D%:PREVOF$="undefnd"
12600 IF DUM$="" THEN PROCnew
12620 GOTO 12460: REM Reopen data channel.
12640 REM - - - - - - - - - - -Write to Disk - - - - - - - - - -
12660 VDU3:PRINT;"Writing new data to ";OF$;" Drive ";D%
12680 ENDPTR=ENDPTR+PTRINT
12700 PTR#OCHN=33:PRINT#OCHN,ENDPTR,NFIXES+1:IF NFIXES+1=1 THEN PRINT#OCHN,NDAY,
NTIME
12720 DAYEND=NDAY:TMEND=NTIME:PTR#OCHN=57:PRINT#OCHN,DAYEND,TMEND
12740 NOPFIX=NFIXES+1
12760 PTR#OCHN=ENDPTR-PTRINT
12780 PRINT #OCHN,NOPFIX,NDAY,NTIME,R(1),R(2),R(3),R(4),R(5),R(6),R(7),RR(1),RR(
2),RR(3),RR(4),RR(5),RR(6),RR(7)
12800 PRINT#OCHN,SCODE,SFXTYP,SXDEG,SYDEG,ZMEAN,XSD,YSD,ZSD,RCODE,RFXTYP,RXDEG,R
YDEG,RZMEAN,RXSD,RYSD,RZSD
12860 IF PTR#OCHN<>HDLEN+(PTRINT*(NFIXES+1)) THEN PRINT;"FILE STRUCTURE ERROR":P
ROCX:STOP:ELSE CLOSE #OCHN:VDU3
12880 ON ERROR PROCerror
12900 PROCrarcnek:ENDPROC
12920 REM=======================================================       =======
```

```
12920 REM=========================================================        =======
12940 DEF PROCrarchek
12960 ON ERROR PROCdiskIOerror
12980 PROCdatdri:OCHN=OPENIN (DR$+OF$):PTR#OCHN=0:INPUT#OCHN,X1$,X2$,X3$:PTR#OCH
N=27:INPUT#OCHN,XX1,XX2,XX3,XX4,XX5,XX6,XX7
13000 PNTR=XX2-XX1:FOR I=1 TO 33:INPUT#OCHN,XX:IF CHEK=1 THEN PRINT;" ";XX;:NEXT
:ELSE NEXT
13010 CLOSE #0:ON ERROR PROCerror
13020 IF PRNTR=1 THEN VDU2
13040 PRINT;"Fix No.:";XX3;"        R-A-R CHECK OK":VDU3
13080 ENDPROC
13100 REM=========================================================================
13110 DEF ACPLOT:STOP
13120 DEF PROCACPLOT:IF QREM$="Y" THEN SOR$="*****":SOR$="S":PROCqon:PRINT;"Do y
ou wish to plot:";:PRINT;"<RTN>   Ship        ";:INPUT"<R>     Remote     ";Q$:I
F Q$<>"" THEN SOR$="*****":SOR$="R"
13140  PROCplot
13160 ENDPROC
13180 REM=========================================================================
13200 DEF PROCplot
13220 IF DATIN$<>"Y" THEN PROCqon:PRINT;"No data to plot":INPUT"<RTN> to proceed
     <M>  for menu";DUM$:PROCqof:IF DUM$<>"" THEN GOTO 14280
13240 IPCALC$=SIPCAL$:IF SOR$="R" THEN IPCALC$=RIPCAL$
13260 IF AUTPROC$="Y" THEN DUM$="":GOTO 13320
13280 PROCqon:VDU4:PRINT;"CHOOSE PLOT WINDOW: ";:PRINT"<RTN>  As before    ";:PR
INT;"<I>    Initial      ";:PRINT;"<N>    New          ";:PRINT;"<C> Cokhat clos
e-up ";
13300 INPUT;"<XN>   Change Inital<MM>   Main Menu    <X>    Exit Plotting";DUM$:
 IF DUM$="X" THEN GOTO 14280:ELSE IF DUM$="MN" THEN PROCstop
13320 DDX=1
13340 IF DUM$="" THEN XMIN=XMIN:XMAX=XMAX:YMIN=YMIN:YMAX=YMAX
13360 IF DUM$="I"THEN XMIN=X(13):XMAX=X(14):YMIN=Y(13):YMAX=Y(14)
13380 IF DUM$="C" AND IPCALC$<>"Y"  THEN PRINT;"Sorry, No Autofix Yet":GOTO 1328
0
13400 IF DUM$="N" THEN PROCqon:INPUT"New Graph Coords (in Km)? xmin,xmax,ymin,ym
ax: ";XMIN,XMAX,YMIN,YMAX:PROCqof
13420 IF DUM$="XN" THEN PROCqon:INPUT"New Graph Coords (in Km)? xmin,xmax,ymin,y
max: ";XMIN,XMAX,YMIN,YMAX:PROCqon:X(13)=XMIN*1000:Y(13)=YMIN*1000:X(14)=XMAX*10
00:Y(14)=YMAX*1000
13440 IF DUM$="N" OR DUM$="XN" THEN XMIN=XMIN*1000:XMAX=XMAX*1000:YMIN=YMIN*1000
:YMAX=YMAX*1000
13460 IF DUM$="M" THEN PROCqon:INPUT" I shall draw a box with sides X km away fr
om the last ship fix, please input X";DDX:PROCqof:MEANX=X(11):MEANY=Y(11)
13477 IF DUM$="C" AND SOR$="S"THEN MEANX=SXCOK:MEANY=SYCOK:DDX=SSDCOK: ELSE IF D
UM$="C" AND SOR$="R" THEN MEANX=RXCOK:MEANY=RYCOK:DDX=RSDCOK
13478 IF DUM$="C" THEN PRINT;"Cokhat Fix Std. Devn=";FNDP(DDX/1000);"Km"
13480 IF DUM$="C"THEN DDX=SSDAUT*2/1000
13500 IF DDX<1.1 THEN DDX=1.1:REM this ensures that minimum plot width is 2200 m
etres, thus at least 2 grid lines will appear on each axis.
13520 IF DUM$="M"OR DUM$="C" THEN COLOUR 131:XMIN=MEANX-1000*DDX:XMAX=MEANX+1000
*DDX:YMIN=MEANY-1000*DDX:YMAX=MEANY+1000*DDX
13540 VDU5:GCOL 0,3:NCOL=0:PROCban(0,637,0,720,"",NCOL):PROCset(YMIN,YMAX,XMIN,X
MAX):REM Define Plot Axes and Paints a black box over graph area.
13560 VDU5:PROCgrid(4,4)
13580 GCOL 0,3:FOR I%=1 TO 8:SYM$="*":IF I%>NBCNS THEN SYM$=""
13600 IF I%=7 AND SCODE<>0 THEN SYM$="S": ELSE IF I%=8 AND RCODE<>0 THEN SYM$="R
"
13620 PROCscal(X(I%),Y(I%)):MOVE XP-15,YP+12:PRINT;SYM$;:IF I%<=NBCNS THEN PRINT
;STR$(I%)
13640 NEXT
13660 VDU4:PROColfix
13680 R(7)=RR(7):REM all values now contain the range from remote to ship
13700 IF DATIN$<>"Y" THEN GOTO 14260:REM avoid plotting data
13720 REM - - - - - Plot Current Data - - - - - - - - - - -
```

```
13720 REM - - - - - Plot Current Data - - - - - - - - - - - -
13740 SRAN=0:IF SOR$="R" THEN SRAN=1
13760 x0=0:x9=600:y0=660:y9=720:ban$="  SHIP DATA":PROCban(x0,x9,y0,y9,ban$,1):I
F SOR$="R" THEN ban$="  REMOTE DATA":PROCban(x0,x9,y0,y9,ban$,1):ban$="depth of
plot="+STR$(ZRIP):PROCban(x0,637,620,660,ban$,1)
13780 REM - - - - Plot range circles from beacons(+ ship and remote?)  - - - -
13800 VDU4:FOR J%=1TO MNRS:IF J%>NBCNS AND J%<MNRS THEN GOTO 13860
13820 IF J%>NBCNS AND SOR$="S" THEN  GOTO 13860
13840 IF SOR$="S" THEN PROCcirc(X(J%),Y(J%),R(J%)):ELSE PROCcirc(X(J%),Y(J%),RR(
J%))
13860 NEXT J%
13880 REM --------- Plot intersection points ---------------------------------
13900 IF IPCALC$="N" THEN PRINT;"IP's not calc. yet":GOTO 14260
13920 VDU4:GCOL 0,2:REM BLUE
13940 REM
13960  IF SOR$="S" THEN NGIPS=SNGIPS:ELSE RNGIPS=NGIPS
13980 FOR NI%=1 TO NGIPS STEP 2:NI2%=NI%+1
14000 IF SOR$="S"THEN X1=XI(NI%):Y1=YI(NI%):X2=XI(NI2%):Y2=YI(NI2%) ELSE X1=RXI(
NI%):Y1=RYI(NI%):X2=RXI(NI2%):Y2=RYI(NI2%)
14020 VDU5:PROCscal(X1,Y1):MOVE XP-15,YP+12:PRINT;"+";NI%
14040 IF X2=X1 AND Y2=Y1 THEN SEP$="  ," ELSE SEP$=""
14060 PROCscal(X2,Y2):MOVE XP-15,YP+12:PRINT;"+";SEP$;NI2%:VDU4
14080 NEXT NI%
14100 REM - - - - - -Plot auto & matrix fixes - - - - - - - - - -
14120 PRINT; "Matrix fix in RED":GCOL 0,1:XX=SXMAT:YY=SYMAT:IF SOR$="R" THEN XX=
RXMAT:YY=RYMAT
14140 PROCscal(XMIN,YY):MOVEXP,YP:PROCscal(XMAX,YY):DRAW XP,YP:PROCscal(XX,YMIN)
:MOVE XP,YP :PROCscal(XX,YMAX):DRAW XP,YP
14160 GCOL 0,2:VDU4:REM BLUE
14180 PRINT;"Cockhat fix in BLUE":GCOL 0,2:XX=SXCOK:YY=SYCOK:IF SOR$="R" THEN XX
=RXCOK:YY=RYCOK
14200 PROCscal(XMIN,YY):MOVEXP,YP:PROCscal(XMAX,YY):DRAW XP,YP:PROCscal(XX,YMIN)
:MOVE XP,YP :PROCscal(XX,YMAX):DRAW XP,YP
14220 REM - - - - Exit or Replot - - - -
14240 IF AUTPROC$="Y" THEN GOTO 14280
14260 PROCqon:PRINT;"TYPE<P> to replot   ";:INPUT"<RTN> to proceed    ";DUM$:PRO
Cqof:IF DUM$="P" THEN GOTO 13220
14280 ENDPROC
14300 REM==================== ========================================
```

```
14300 REM==================== ================================================
14320 DEF PROCcirc(XC,YC,R) :REM Draw circle, centre XC,YC radius R.
14340 GCOL 0,3:REM WHITE CIRCLES
14360 PROCscal(XC,YC):MOVE XP,YP:PROCscal(XC+R,YC):PLOT 149,XP,YP
14380 ENDPROC
14400 REM=========================================================================
14420 DEF PROCset(YMIN,YMAX,XMIN,XMAX):REM Sets up plot scaling factors.
14440 X0=0:X9=500:Y0=0:Y9=500:XLEN=X9-X0:YLEN=Y9-Y0
14460 XRAN=XMAX-XMIN:YRAN=YMAX-YMIN
14480 XSCA=XLEN/XRAN:YSCA=YLEN/YRAN
14500 ENDPROC
14520 REM=========================================================================
14540 DEF PROCscal(X,Y): REM scales data for plotting using variables defined in
 PROCsca.
14560 XP=X0+XSCA*(X-XMIN):YP=Y0+YSCA*(Y-YMIN)
14580 ENDPROC
14600 REM=========================================================================
14620 DEF PROCgrid(XDIV,YDIV)
14640 @%=&0:REM reinitialise
14660 FOR I=XMIN TO XMAX STEP XRAN/XDIV:IX%=I
14680 IF IX%<>XMIN AND IX%<>XMAX THEN IX%=INT(I/1000) *1000
14700 PROCscal(IX%,YMIN):MOVE XP,YP:PROCscal(IX%,YMAX):DRAW XP,YP:PLOT 0,0,25:PR
INT;INT(IX%/1000):NEXT
14720 FOR I=YMIN TO YMAX STEP YRAN/YDIV:IY%=I
14740 IF IY%<> YMIN AND IY%<> YMAX THEN IY%= INT(I/1000) *1000
14760 PROCscal(XMIN,IY%):MOVE XP,YP:PROCscal(XMAX,IY%):DRAW XP,YP:PLOT 0,0,25:PR
INT;INT(IY%/1000):NEXT
14780 ENDPROC
14800 REM=========================================================================
14820 DEF PROColfix
14840 IF AUTPROC$="Y" THEN NPF=5:OSOR$=SOR$:GOTO 14882
14860  PROCqon:INPUT"Type how many previous fixes to plot";NPF:PROCqof:IF NPF=0
THEN GOTO 15180
14880 IF QQREM <>1 THEN OSOR$="S":ELSE PROCqon:PRINT;"Do you want to plot:";:PRI
NT"<S>    ship          ";:PRINT"<R>    remote       ";:INPUT"<RTN>  both
 ";OSOR$ :PROCqof
14881 PROCrenam
14882 IF OF$="undefnd" THEN GOTO 15180:REM endproc
14884 PROCdatdri:OCHN=OPENIN (DR$+OF$)
14886 PTR#OCHN=27:INPUT#OCHN,PTRINT,ENDPTR,NFIXES:IF NFIXES=0 THEN PROCprevfile:
 IF PREVOF$="undefnd" THEN GOTO 15180:REM endproc
14888 NFBEOF=0:VDU4
14900 FOR NF=1 TO NPF
14901 REM - - - - - - - - - - - Read 1 fix from disk - - - - - - - - - - - - -
14902 NFBEOF=NFBEOF+1: PROCreadFfix:REM reads old fix data into spare space.
14919 REM - - - - - - - - - - - Screen Plot the Fix  - - - - - - - - - - - - -
14920 VDU5:IF OSOR$="R" THEN GOTO 14980 ELSE GCOL 0,1:PROCscal(FXMEAN,FYMEAN):RE
M IF NF<=1 THEN GOTO 14960
14940 IF NF>1 THEN MOVE XS,YS:DRAW XP,YP
14960 MOVE XP-15,YP+12:PRINT;"s":XS=XP:YS=YP
14980 IF OSOR$="S" THEN GOTO 15010 ELSE GCOL 0,1:PROCscal(FRXMEAN,FRYMEAN):IF NF
>1 THEN MOVE XR,YR:DRAW XP,YP
15000 MOVE XP-15,YP+12:XR=XP:YR=YP:PRINT;"r"
15005 VDU4
15010 IF FFIX=1 AND NF<NPF THEN VDU4:PROCprevfile:IF PREVOF$="undefnd" THEN GOTO
 15180:REM endproc
15020 VDU4: NEXT NF
15030  CLOSE#0
15040 VDU5:IF XMEAN<9*10^5 THEN PROCscal(XMEAN,YMEAN):MOVE XP-15,YP+12:PRINT;"S"
:REM Plot present ship posn. if detrmnd yet
15060 IF RXMEAN<9*10^5 THEN PROCscal(RXMEAN,RYMEAN):MOVE XP-15,YP+12:PRINT;"R":R
EM Plot present remote posn. if detrmnd yet
15180 VDU4:CLOSE#0:ENDPROC
```

```
15200 REM=======================================================================
15220 DEF PROChrdplt: REM For producing hard-copy plot
15230 PRINT;"SORRY Not working yet!"
15240 PROCqon:INPUT"Please give day &   time of start     and day & time of end "
;DAY0,TM0,DAY9,TM9:PROCqof
15260 PROCqon:INPUT"How many fixes back shall I search      ";NPF:PROCqof
15280 ENDPROC
15300 REM=======================================================================
15320 DEF PROCcoknat:REM find 3 intersections on horiz. plane with minimum devia
tion from centroid. Called from PROCcalc only.
15340 A=1:REM This flags that PROCauto has been called. A is set to -1 in PROCda
tin
15360 SOR$=SORR$
15380 NRANS=0:SDCOK=999999:IF PRNTR=1 THEN VDU2
15400 IF SOR$="S" THEN PRINT;"SHIP ";:ZCOK=0:ELSE PRINT;"REMOTE ";
15420 IF SOR$="R" THEN ZCOK=ZRIP
15460 PRINT;"Cockhat Fix ";NGRANS;" good ranges";:IF NGRANS<3 THEN PRINT;"~:";:
GOTO 15720
15480 FOR I=1 TO NGIPS-1:IF SOR$="S" THEN XI=XI(I):YI=YI(I):ELSE XI=RXI(I):YI=RY
I(I)
15500 FOR J=I+1TO NGIPS:IF SOR$="S"THEN XJ=XI(J):YJ=YI(J):ELSE XJ=RXI(J):YJ=RYI(
J)
15520 IF J MOD 2 =0 AND I=J-1 THEN GOTO 15680:REM exclude IPs on same 2 circles.
15540 FOR K=J+1 TO NGIPS:IF SOR$="S"THEN XK=XI(K):YK=YI(K):ELSE XK=RXI(K):YK=RYI
(K)
15560 IF K MOD 2 =0 AND J=K-1 THEN GOTO 15660:REM exclude IPs on same 2 circles.
15580 IF K MOD 2 =0 AND I=K-1 THEN GOTO 15660:REM exclude IPs on same 2 circles.
15600 XXMN=(XI+XJ+XK)/3:YYMN=(YI+YJ+YK)/3
15620 MMSD= (1/3)*( SQR((XI-XXMN)^2 +(YI-YYMN)^2) + SQR((XJ-XXMN)^2 +(YJ-YYMN)^2
) + SQR((XK-XXMN)^2 +(YK-YYMN)^2) )
15640 IF MMSD <SDCOK THEN SDCOK=MMSD:XCOK=XXMN:YCOK=YYMN:NINT1=I:NINT2=J:NINT3=K

15660  NEXT K
15680 NEXT J
15700 NEXT I
15720  IF SDCOK=999999 THEN PRINT;"No Cockhat Fix.":XCOK=999999:YCOK=999999:ZCOK
=ZCOK:NINT1=0:NINT2=0:NINT3=0
15740 IF SOR$="S" THEN SSDCOK=SDCOK:SXCOK=XCOK:SYCOK=YCOK:SZCOK=ZCOK:SHINT1=NINT
1:SHINT2=NINT2:SHINT3=NINT3
15760 IF SOR$="R" THEN RSDCOK=SDCOK:RXCOK=XCOK:RYCOK=YCOK:RZCOK=ZCOK:RINT1=NINT1
:RINT2=NINT2:RINT3=NINT3
15780 IF PRNTR=1 THEN VDU2
15800 IF SDCOK=999999 THEN GOTO 15860
15820 @%=&020009:PRINT;" IPs:- ";NINT1;" ";NINT2;" ";NINT3;';STRING$(17," ");" X
:";XCOK;" Y:";YCOK;" Z assumed:";ZCOK;" S.D.:"FNDP(SDCOK);" m"
15840 VDU3
15860 ENDPROC
16280 REM=======================================================================
```

```
16280 REM=============================================================================
16300 DEF PROCprevfile: REM Opens previous file for reading from
16320 PTR#OCHN=(0+9):INPUT#OCHN,PREVOF$:PRINT;"Prev file:"; PREVOF$
16340 IF PREVOF$="undefnd" THEN PRINT;"No more previous fixes":ELSE CLOSE #OCHN:
OCHN=OPENIN(DR$+ PREVOF$):PTR#OCHN=27:INPUT#OCHN,PTRINT,ENDPTR:NFBEOF=0
16360 ENDPROC
16380 REM=============================================================================
16400 DEF PROCswitch: REM For user to set diagnostics, printer, sound and transp
onder options.
16410 PROCqon:INPUT"Do you want to reset switches RTN/N";DUM$:PROCqof:IF DUM$="N
" THEN GOTO 16620
16420 PROCqon:INPUT"Do you want diagnostics printed Y/RTN";CHEK$:PROCqof
16440 PROCqon:INPUT"Do you want sound   Y/RTN";Q$:PROCqof:IF Q$="Y" THEN SND=1:
ELSE SND=0
16448 *FX210,0
16449 IF SND=1 THEN GOTO 16520:REM avoid sound off.
16450 *FX210,1
16520 PRNTR=0:PROCqon:INPUT"Do you want print-out Y/RTN";DUM$:PROCqof:IF DUM$<>"
Y" THEN GOTO 16600
16540  PRNTR=1:PROCqon:PRINT;"Ensure it is on and ON-LINE and hit <RTN>":PROCqof
:INPUT DUM$
16560 *FX6,10
16580 VDU2:@%=&20209:PRINT:PRINT "READY":PRINT:VDU3
16600 QQREM=0:QREM$="N":PROCqon:INPUT"Do you want option  to fix remote txpdr Y/
RTN            ";DUM$:PROCqof:IF DUM$="Y" THEN QQREM=1
16610 PROCtransout
16620  ENDPROC
16640 REM =============================================================================
16660 DEF PROCnewtrans: REM Sets up new empty trans file no variables except D%
need be defined.
16680 PRINT;"Creating 'trans'"
16700 CHEK=0:QQREM=0:PRNTR=0
16720 ON ERROR PROCdiskIOerror
16740 CLOSE#0:PROCdatdri:OCHN=OPENOUT(DR$+"trans"):PRINT#OCHN,0,0,"undefnd","und
efnd","......",999999,999999,999999,999999
16760 FOR I=9 TO 34:PRINT#OCHN,999999:NEXT:PRINT#OCHN,CHEK,QQREM,PRNTR
16780 FOR I=38 TO 84:PRINT#OCHN,999999:NEXT:PRINT#OCHN,Z(14):FOR I=86 TO 99:PRIN
T#OCHN,999999:NEXT:CLOSE#0
16800 ON ERROR PROCerror
16820 PROCtransin:
16840 ENDPROC
16860 REM=============================================================================
16880 DEF PROCmetdeg(XP,YP): REM Converts from local metric coords (XP,YP) to lo
ngs & lats (XDEG,YDEG)
16900 YDEG=YDEG0+YP/(1852*60)
16920 MULT=SGN(YDEG)*COS(RAD(YDEG))
16940 XDEG=XDEG0+XP/(MULT*1852*60)
16960 ENDPROC
16980 REM=============================================================================
17000 DEF FNDP(X): REM Truncates a long decimal into two or less decimal places.
17020 X=INT(100*X + SGN(X)*0.5)/100
17040 =X
17060  REM=============================================================================
```

```
17060  REM=============================================================
17100 DEF PROCMATFIX: REM For matrix fixing of ship OR  remote with >=3 (or 4)
good ranges.
17120 IF SORR$="R" THEN GOTO 17340
17140 SXMAT=999999:SYMAT=999999:SZMAT=999999:SSDMAT=999999:MCODE$=""
17160 MNNRANS=3:PROCmatset:IF NRANS<MNNRANS THEN GOTO 17660:ELSE SZMAT=0:REM and
 stays =0, unless matrix method modified.
17180 R1=SQR( (R(BCN1))^2 + Z1^2): R2=SQR( (R(BCN2))^2 + Z2^2): R3=SQR( (R(BCN3)
)^2 + Z3^2):REM produces slant ranges in m.
17200 DET=(A1*B2 -A2*B1): ALFA=(R1^2 -R2^2)+(X2^2+Y2^2+Z2^2)-(X1^2+Y1^2+Z1^2):BE
TA=(R1^2 -R3^2)+(X3^2+Y3^2+Z3^2)-(X1^2+Y1^2+Z1^2)
17220 SXMAT=(ALFA*B2 -BETA*B1)/DET:SYMAT=(BETA*A1-ALFA*A2)/DET:SZMAT=0
17240 D1=SQR( ( R1 -SQR((SXMAT-X1)^2 +(SYMAT-Y1)^2 +(SZMAT-Z1)^2 )) ^2): D2=SQR(
 ( R2 -SQR((SXMAT-X2)^2 +(SYMAT-Y2)^2 +(SZMAT-Z2)^2 )) ^2): D3= SQR( ( R3 -SQR((
SXMAT-X3)^2 +(SYMAT-Y3)^2 +(SZMAT-Z3)^2 )) ^2):SSDMAT=(D1+D2+D3)/3
17260 IF PRNTR=1 THEN VDU2
17280 PRINT;"SHIP   Matrix Fix X:";SXMAT; " Y:";SYMAT;"Z:";SZMAT;" Mean Error= "
;SSDMAT;" m."
17300 VDU3:GOTO 17660:REM ENDPROC
17320 REM  - - - - - - REMOTE - - - - - - - - - - - - - - - - - - - - - - - - -
17340 PRINT;"Matrix Fix of Remote"
17360 RXMAT=999999:RYMAT=999999:RZMAT=999999:RSDMAT=999999:
17380 ZR=Z(8):MNNRANS=4:PROCmatset:IF NRANS<MNNRANS THEN GOTO 17660
17400 X4=X(BCN4):Y4=Y(BCN4):Z4=Z(BCN4)
17420 R1=SQR( (RR(BCN1))^2 + (Z1-ZRIP)^2): R2=SQR( (RR(BCN2))^2 + (Z2-ZRIP)^2):
R3=SQR( (RR(BCN3))^2 + (Z3-ZRIP)^2):R4= SQR ( (RR(BCN4))^2 + (Z4-ZRIP)^2):REM ca
lcs slant ranges (m.) using remote depth estimate of I/P.
17440 ALFA=(R1^2 -R2^2)+(X2^2+Y2^2+Z2^2)-(X1^2+Y1^2+Z1^2):BETA=(R1^2 -R3^2)+(X3^
2+Y3^2+Z3^2)-(X1^2+Y1^2+Z1^2)
17460 GAMA=(R1^2-R4^2)+(X4^2+Y4^2)-(X1^2+Y1^2+Z1^2):A3=2*(X4-X1):B3=2*(Y4-Y1):C1
=2*(Z2-Z1):C2=2*(Z3-Z1):C3=2*(Z4-Z1)
17480 RDET=A1*(B2*C3 -B3*C2)-B1*(A2*C3-A3*C2)+C1*(A2*B3-A3*B2)
17500 RXMAT=( ALFA*(B2*C3-B3*C2) -B1*(BETA*C3-GAMA*C2) +C1*(BETA*B3-GAMA*B2) )/R
DET
17520 RYMAT=( A1*(BETA*C3-GAMA*C2)-ALFA*(A2*C3-A3*C2)+C1*(A2*GAMA-A3*BETA) )/RDE
T
17540 RZMAT=( A1*(B2*GAMA-B3*BETA) -B1*(A2*GAMA-A3*BETA) +ALFA*(A2*B3-A3*B2) )/R
DET
17560 RD1=SQR( ( R1 -SQR((RXMAT-X1)^2 +(RYMAT-Y1)^2 +(RZMAT-Z1)^2 )) ^2): RD2=SQ
R( ( R2 -SQR((RXMAT-X2)^2 +(RYMAT-Y2)^2 +(RZMAT-Z2)^2  )) ^2)
17580 RD3= SQR( ( R3 -SQR((RXMAT-X3)^2 +(RYMAT-Y3)^2 +(RZMAT-Z3)^2 )) ^2):RD4= S
QR( ( R4 -SQR((RXMAT-X4)^2 +(RYMAT-Y4)^2 +(RZMAT-Z4)^2 )) ^2):RSDMAT=(RD1+RD2+RD
3+RD4)/4
17600 IF PRNTR=1 THEN VDU2
17620 PRINT;"REMOTE Matrix Fix X:";RXMAT; " Y:";RYMAT;" Z:";RZMAT;" Mean Error=
";RSDMAT;" m."
17640 VDU3
17660 ENDPROC
17680 REM ==========================================================
```

```
17680 REM ==========================================================================
17700 DEF PROCmatset
17720 NRANS=0:MCODE$="":FOR BN=1 TO MNRS:RR=RR(BN):IF SORR$="S" THEN RR=R(BN):IF
 BN=MNRS THEN GOTO 17760: REM Skip ship:remote range for ship.
17740 IF RR>1 THEN NRANS=NRANS+1:BCNO(NRANS)=BN :MCODE$=MCODE$+STR$(BN)
17760 NEXT BN
17780 MNNRNS=3:IF SORR$="R" THEN MNNRNS=4
17800 IF NRANS<MNNRNS THEN PRINT;"<";MNNRNS;" ranges-No Matfix":ENDPROC:ELSE IF
NRANS=MNNRNS THEN PRINT;"Sufficient Ranges for Matfix (";NRANS;")":GOTO 17920
17820 IF NRANS<=MNNRNS OR AUTPROC$<>"Y" THEN GOTO 17860 :ELSE MCODE$=SCODE$:IF S
OR$="R" THEN MCODE$=RCODE$
17840 FOR I%=1 TO MNNRNS:BCNO(I%)=VAL(MID$(MCODE$,I%,1)):NEXT I%: REM Decode MCO
DE$ to give BCNO values.
17860 IF NRANS>MNNRNS THEN PROCqon:PRINT;">";MNNRNS;" good ranges! Which shall I
 use in MATRIX FIX???":PRINT;"(To plot ranges: <RTN> Else: <N>)":PROCqof
17880 IF NRANS>MNNRNS THEN INPUT Q$:IF Q$<>"N" THEN SORR$=SORR$:PROCplot:PROCqon:
PRINT;"WHICH RANGES SHALL I USE IN MATFIX ???":PROCqof
17900 IF NRANS>MNNRNS THEN :PRINT;"N.B. For Ship range type 7":MCODE$="":FOR I=1
 TO MNNRNS:PROCqon:PRINT;"Range ";I;" ";:INPUT BCNO(I):PROCqof:MCODE$=MCODE$+STR
$(BCNO(I)):NEXT:PRINT
17920 BCN1=BCNO(1):BCN2=BCNO(2):BCN3=BCNO(3):BCN4=BCNO(4):BCN5=BCNO(5):BCN6=BCNO
(6):BCN7=BCNO(7)
17940 X1=X(BCN1):Y1=Y(BCN1):Z1=Z(BCN1):X2=X(BCN2):Y2=Y(BCN2):Z2=Z(BCN2):X3=X(BCN
3):Y3=Y(BCN3):Z3=Z(BCN3)
17960 A1=2*(X2-X1):A2=2*(X3-X1):B1=2*(Y2-Y1):B2=2*(Y3-Y1)
17980 MCODE=VAL(MCODE$):ENDPROC
18000 REM ==========================================================================
18020 DEF PROCX: REM Change white toflashing red/cyan and back and beep! 3 times
, then change background to red.
18040 FOR II%=1TO3:VDU19,3,9,0,0,0:VDU7:XX=INKEY(15):VDU19,3,7,0,0,0:XX=INKEY(15
)::NEXTII%:PROCqof:ENDPROC
18060 REM ==========================================================================
```

```
18060 REM ======================================================================
18080 DEF PROCreproce: REM For reading in fixes from old datafile, reprocessing,
 (maybe with new beacon data) and writing to another file.
18090 REPROC$="Y"
18100 QQREM=1:PROChdrd:REM Open IPFILE$, read header, list if necessary.
18120 PROCqon:INPUT"Name of destination datafile";OF$:PROCqof
18140 PRINTOF$
18160 NFIX=0
18180 PROCqon:INPUT"<RTN> To consider fix by fix  <B> To Batch process a chunk o
f fixes  <X>  to exit          ";Q$:PROCqof:IF Q$="X" THEN PROCstop:ELSE IF Q$<>"
B" THEN GOTO 18360
18200 PROCqon:INPUT"Please give No.s of first & last fixes of chunk";NFIX1,NFIX9
:INPUT "<Y> to copy unchanged OR <RTN> to reprocess & copy";Q$:PROCqof:
18220 FOR NFIX=NFIX1 TO NFIX9:
18240 PROCreset:AUTPROC$="Y":PROCreadlfix(ICHN,NFIX):QREM$="N":ZRIP=RZMEAN:Z(8)=
ZRIP
18260 IF Q$="Y" THEN PROCsave : GOTO 18300
18280 PROCrelabel:DATIN$="Y":PROCcalc:PROCfix
18300 NEXT NFIX:IF NFIX>=NFIXES THEN GOTO 18540:ELSE GOTO 18180
18320 REM  - - - - - - - - - - Fix by Fix - - - - - - - - - - - - - - - - -
18340 REM  Do not remove
18360 PROCqon:INPUT"Please give number of fix to read or <RTN> for next fix";DUM
:PROCqof:IF DUM=0 THEN NFIX=NFIX+1: ELSE NFIX=DUM
18380 PROCreset:PROCreadlfix(ICHN,NFIX):ZRIP=RZMEAN:Z(8)=ZRIP
18400 PROCrelabel
18420 PROCqon:PRINT;"SELECT:-              ";"<Y> Copy unchanged  ";"<RTN> Reproc
ess,Copy";"<S> Skip this fix   ";"<B>  Process whole chunk of fixes":INPUT Q$:PR
OCqof:IF Q$<>"Y" AND Q$<>"" AND Q$<>"S" AND Q$<>"B" THEN PROCX:GOTO 18420
18440 IF Q$="B" THEN GOTO 18200
18460 IF Q$="S" THEN GOTO 18360
18480 IF Q$="Y" THEN PROCsave
18500 IF Q$<>"" THEN GOTO 18520:ELSE PROCqon:PRINT;"REPROCESSING:-       ";"<N>
 Monitored      ";"<Y>  Automatic      ":INPUT AUTPROC$:PROCqof:DATIN$="Y":PROCc
alc:PROCfix
18520 GOTO 18180
18540 ENDPROC
18560 REM ======================================================================
18580 DEF PROCrelabel
18600 x0=0:x9=600:y0=840:y9=900:bann$="Day:"+DAY$+" Time:"+NTIME$:PROCban(x0,x9,
y0,y9,bann$,1)
18620 x0=0:x9=630:y0=0:y9=840:PROCban(x0,x9,y0,y9,"",0):REM Blank out plot and l
abels.
18640 x0=0:x1=210:y0=800:y9=840:ban$=" SHIP: ":PROCban(x0,x1,y0,y9,ban$,1):IF QQ
REM=1 THEN y0=760:y9=800:ban$=" REMT: ":PROCban(x0,x1,y0,y9,ban$,1)
18660 x0=0:x9=600:y0=720:y9=760:ban$="DATA not yet saved":PROCban(x0,x9,y0,y9,ba
n$,1)
18680 ENDPROC
18700 REM======================================================================
```

```
18700 REM=========================================================================
18720 DEF PROCdegmet(xd,yd) :REM Converts from degrees to local metres.
18740 ym=(yd-YDEG0)*1851.8 * 60 :xm=(xd-XDEG0)*1851.8 * 60 *COS(RAD(yd))
18760 IF xd=999999 THEN xm=999999:ym=999999
18780 ENDPROC
18800  DEF PROCinterp
18820 IF PRNTR=1 THEN VDU2
18840 PRINT;"Old Code=";CODE;
18860 CODE$=STR$(CODE)
18880 IF CODE=0 THEN FXTYP=9: ELSE IF LEN(CODE$)<=2 THEN FXTYP=2
18900 IF LEN(CODE$)=3 AND CODE=MCODE THEN FXTYP=4:REM Matrix
18920 IF LEN(CODE$)=4 THEN FXTYP=2
18940 IF LEN(CODE$)=5 OR LEN(CODE$)=6 THEN IF CODE=CHCODE THEN FXTYP=3
18960 IF LEN(CODE$)=5 OR LEN(CODE$)=6 THEN IF CODE<>CHCODE THEN FXTYP=2
18980 IF LEN(CODE$)>=7 THEN FXTYP=2
19000 PRINT;" Interpreted fixtype=";FXTYP:VDU3
19020 ENDPROC
19040 DEF PROCstop
19060 PRINT;"***Program Stopped**"
19080 PROCqon:PRINT;"Hit key ";:COLOUR 129:PRINT;"f3";:PROCqon:PRINT;" for MAIN
MENU":PROCqof::STOP
19100 ENDPROC
19399 REM = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
19400 DEF PROCreadFfix: REM reads old fix data into spare space.
19420 PNTR=ENDPTR-(NFBEOF-1)*PTRINT
19440 PTR#OCHN=PNTR-(6*6):INPUT#OCHN,FRXDEG,FRYDEG,FRZMEAN
19460 IF PTRINT=186 THEN PTR#OCHN=PNTR-(13*6):ELSE PTR#OCHN=PNTR-(14*6)
19480 INPUT#OCHN,FXDEG,FYDEG,FZMEAN: REM No escape for >999998?
19500 PROCdegmet(FXDEG,FYDEG):FXMEAN=xm:FYMEAN=ym:PROCdegmet(FRXDEG,FRYDEG):FRXM
EAN=xm:FRYMEAN=ym
19520 IF PTRINT=186 THEN PTR#OCHN=PNTR-(31*6):ELSE PTR#OCHN=PNTR-(33*6)
19540 INPUT#OCHN,FFIX,FDAY,FTIME:PRINT;"Fix,Day,Time:"; FFIX;",";FDAY;",";FTIME
19560 DYTM%=FDAY*10000+FTIME
19600 ENDPROC
```

BEACON

```
    5 REM BEACON by S. WILLIAMS ON 5/6/87
   10 REM Program BEACON prints out current estimates of ACNAV beacon positions
as read from the file "beepos". If required the  values can be updated.
   20 RR=0:QREAD$="N":TITLE$="":SVEL=0:DRINO=0:NB=0:XMIN=0:XMAX=0:YMIN=0:YMAX=0:
XDEG0=0:YDEG0=0
   30 REM The initial plot coordinates and the assumed average water column soni
c velocity are also adjustable.
   40 DIM BCOL$(6),TDEL(6),XP(6),YP(6),ZP(6),X(10),Y(10),Z(10)
   45 *TV0,1
   50 MODE 129:VDU19,2,6,0,0,0:VDU19,1,1,0,0,0:COLOUR 0:COLOUR 131:CLS
   52 DUM=INKEY(5)
   55 PRINT;"BEACON by srjw 5.11.86"
   56 *KEY 0 MODE 131¶M LIST¶M
   57 *KEY 1 CLS¶M RUN¶M
   60 PROCqon:PRINT;"Please give data disk drive No.":PROCqof:INPUT DRINO
   65 PRINT':PROCqon:PRINT;"Give BDF filename or type 'N' to create new file":PR
OCqof:INPUT BEEP$:IF BEEP$="N" THEN PROCedit:ELSE PROCread
   70 RR=0:PROCmenu:GOTO 70
   80 PROCqon:INPUT"Do you wish to return to the program NAVPLT? Please type Y o
r hit 'RTN'";DUM$:PROCqof:IF DUM$="Y" THEN COLOUR 131:CLS:PRINT;"Please wait whi
le I load NAVPLT from drive 0":CHAIN ":0.NAVPLT"
   90 STOP
   95 REM ===========================================================
  100 DEF PROCmenu
  110 PROCqon:PRINT;"Do you wish to:     ":PRINT;"<R> read ";:PRINT;"<E> edit ";
:PRINT"<W> write to/from disk OR:    ":INPUT"<N> return  to NAVPLT";DUM$:PROCqof
  120 IF DUM$="N" THEN CLS:PROCqon:INPUT"Please ensure  Program disk is in Drive
0 and type <RTN>";DUM$:CHAIN ":0.NAVPLT"
  123 IF DUM$="R" THEN PROCread:PROClist
  126 IF DUM$="E" THEN PROCedit:PROClist
  129 IF DUM$="W" THEN PROCwrite:PROCread:PROClist
  130 ENDPROC
  135 REM ===========================================================
  140 DEF PROCnew
  150 TITLE$="":SVEL=0:DRINO=0:NB=0:XMIN=0:XMAX=0:YMIN=0:YMAX=0:XDEG0=0:YDEG0=0
  160 PROCedit
  165 REM ===========================================================
  170 ENDPROC
  180 DEF PROCread:RR=1:QREAD$="Y"
  190 PROCdatdri:ICHN=OPENIN(BEEP$)
  200 INPUT #ICHN,TITLE$,SVEL,DRINO,NB,XMIN,XMAX,YMIN,YMAX,XDEG0,YDEG0:FOR I=1 T
O NB:INPUT# ICHN,BCOL$(I),TDEL(I),XP(I),YP(I),ZP(I):NEXT
  210 CLOSE # 0
  220 PROClist:PROCmenu
  230 ENDPROC
  235 REM ===========================================================
```

```
  235 REM ===========================================================================
  240 DEF PROCedit
  250 PRINT;"Filename=";TITLE$:PROCqon:PRINT;"Please enter alternative filename
or type RETURN if OK":INPUT DUM$:PROCqof:IF DUM$<>"" THEN TITLE$=DUM$:BEEP$=DUM$

  260  PRINT;"Average sonic velocity=";SVEL:PROCqon:PRINT;"Please enter alternat
ive value or type RETURN if OK":INPUT DUM$:PROCqof:IF DUM$<>"" THEN SVEL=VAL(DUM
$)
  270 PRINT;"Disk Drive No. for datafile read/write=";DRINO:PROCqon:PRINT;"Pleas
e enter alternative value or type RETURN if OK":PROCqof:INPUT DUM$:IF DUM$<>"" T
HEN DRINO=VAL(DUM$)
  280  PRINT;"No. of Beacons=";NB:PROCqon:PRINT;"Please enter alternative value
or type RETURN if OK":PROCqof:INPUT DUM$:IF DUM$<>"" THEN NB=VAL(DUM$)
  290  PRINT;"Longitude 0 for local metric grid X coord:";SGN(XDEG0)*INT(ABS(XDE
G0));", ";60*(XDEG0-(SGN(XDEG0)*INT(ABS(XDEG0))))
  300 PROCqon:PRINT;"Please give alternative in (+/-) degrees and (+/-) minutes
(+=E, -=W) e.g.: -25 <RTN> -40<RTN> OR  press <RTN> if OK":PROCqof:INPUT DUM$:IF
 DUM$<>"" THEN INPUT DUM2$:XDEG0=VAL(DUM$)+(VAL(DUM2$))/60:PRINT XDEG0
  310  PRINT;"Latitude 0 for local metric grid Y coord:";SGN(YDEG0)*INT(ABS(YDEG
0));", ";60*(YDEG0-(SGN(YDEG0)*INT(ABS(YDEG0))))
  320 PROCqon:PRINT;"Please give alternative in (+/-) degrees and (+/-) minutes
(+=N, -=S) e.g.: 30 <RTN> 10 <RTN> OR  press <RTN> if OK":PROCqof:INPUT DUM$:IF
DUM$<>"" THEN INPUT DUM2$:YDEG0=VAL(DUM$)+(VAL(DUM2$))/60:PRINT YDEG0
  322 PRINT;"XMIN,XMAX,YMIN,YMAX: ";XMIN,XMAX,YMIN,YMAX
  323 PROCqon:PRINT;"Do you wish to change Plot Coords enter'Y' or 'RTN' ":PROCq
of:INPUT Q$:IF Q$="Y"THEN PRINT;"Type XMIN,XMAX,YMIN,YMAX ": INPUT XMIN,XMAX,YMI
N,YMAX
  325 PROClist
  330 PROCqon:PRINT;"Please enter No. of Beacon for editing OR <RTN> for Menu":P
ROCqof:INPUT I:IF I=0 THEN GOTO 470
  340 PRINT;"Beacon No.:";I;" Colour:";BCOL$(I)
  350 PROCqon:INPUT"Enter new value or press RETURN if OK"; DUM$:PROCqof:IF DUM$
<>"" THEN BCOL$(I)=DUM$
  360 PRINT;"Receive/transmit delay (ms):";1000*TDEL(I)
  370 PROCqon:PRINT;"Enter new value in ms. or press RETURN if OK":PROCqof:INPUT
 DUM$:IF DUM$<>"" THEN TDEL(I)=(VAL(DUM$))/1000
  380 PROCmetdeg(XP(I),YP(I)):PRINT"Latitude (Y):";SGN(YDEG)*INT(ABS(YDEG));", "
;60*(YDEG-(SGN(YDEG)*INT(ABS(YDEG))));"  metres:";YP(I)
  390 PROCqon:PRINT;"Press <RTN> if OK..OR New value (Degs and Mins +:E, -:W) :+
/- DEGREES...":PROCqof:INPUT DUM$:IF DUM$<>"" THEN PRINT;"+/- MINUTES...":PROCqo
f:INPUT YMINT:YDEG=VAL(DUM$)+YMINT/60:YP(I)=(YDEG-YDEG0)*1852*60:GOTO 380
  400 PRINT;"Longitude (X):";SGN(XDEG)*INT(ABS(XDEG));", ";60*(XDEG-(SGN(XDEG)*I
NT(ABS(XDEG))));"  metres:";XP(I)
  410 PROCqon:PRINT;"<RTN> if OK OR New value (Degs & Mins +:N, -:S) :+/- DEGREE
S...":PROCqof:INPUT DUM$:IF DUM$<>"" THEN PROCqon:INPUT"+/- MINUTES...";XMINT:XD
EG=VAL(DUM$)+XMINT/60:XP(I)=(XDEG-XDEG0)*1852*60*COS(RAD(YDEG)):GOTO 400
  420 PRINT;"Z-Pos:";ZP(I):PROCqon:PRINT;"Enter new value or press RETURN if OK"
:PROCqof:INPUT DUM$:IF DUM$<>"" THEN ZP(I)=VAL(DUM$)
  440 GOTO 325
  470 ENDPROC
  475 REM ===========================================================================
```

```
  475 REM ===========================================================
  480 DEF PROCwrite
  490 CLOSE #0
  500 PRINT;"Now to write to disk."
  510 PROCdatdri:OCHN=OPENOUT (BEEP$)
  520 PRINT# OCHN,BEEP$,SVEL,DRINO,NB,XMIN,XMAX,YMIN,YMAX,XDEG0,YDEG0
  530 FOR I=1TO NB:PRINT #OCHN,BCOL$(I),TDEL(I),XP(I),YP(I),ZP(I):NEXT
  540 CLOSE # 0
  550 PRINT;"Will now read back data to check"
  560 ENDPROC
  565 REM ===========================================================
  570 DEF PROClist
  580 PRINT;"Filename:";TITLE$:PRINT;"Sound Velocity (m/s):";SVEL:PRINT;"Data Dr
ive No.:";DRINO:PRINT "There are ";NB;" beacons."
  590 PRINT;"Origin of local grid:Lat(Y):";SGN(YDEG0)*INT(ABS(YDEG0));",";INT(10
0*60*(YDEG0-(SGN(YDEG0)*INT(ABS(YDEG0)))))/100:PRINT;"Lon(X):";SGN(XDEG0)*INT(AB
S(XDEG0));",";INT(100*60*(XDEG0-(SGN(XDEG0)*INT(ABS(XDEG0)))))/100
  595PRINT "XMIN=";XMIN;"XMAX=";XMAX;"YMIN=";YMIN;"YMAX=";YMAX
  600 FOR I=1 TO NB:PRINT:PRINT;"******Beacon #";I;" :";BCOL$(I);" Delay (ms):";
1000*TDEL(I)
  610 PRINT "X: ";INT(XP(I)+.5);" Y: ";INT(YP(I)+.5);"  Z: ";ZP(I)
  620 PROCmetdeg(XP(I),YP(I)):PRINT;" Lat(Y):";SGN(YDEG)*INT(ABS(YDEG));",";INT(
100*60*(YDEG-(SGN(YDEG)*INT(ABS(YDEG))))+.005)/100;:PRINT;" Lon(X):";SGN(XDEG)*I
NT(ABS(XDEG));",";INT(100*60*(XDEG-(SGN(XDEG)*INT(ABS(XDEG))))+.005)/100
  630 NEXT
  650 PROCqon:INPUT"Hit 'RTN' to continue";DUM$:PROCqof
  660 ENDPROC
  665 REM ===========================================================
  670 DEF PROCqon:COLOUR 130:VDU7:ENDPROC
  680 DEF PROCqof:COLOUR 131:ENDPROC
  685 REM ===========================================================
  690 DEF PROCdatdri:REM Sets drive No. for writing data.
  700 TEXT$="*DR."+STR$(DRINO):PROCcfi:ENDPROC
  705 REM ===========================================================
  710 DEF PROCcfi:REM For using command file instructions
  720 PRINT;"Command is:";TEXT$:$&900=TEXT$:Y%=9:X%=0:CALL &FFF7:ENDPROC
  725 REM ===========================================================
  730 DEF PROCmetdeg(XP,YP):REM converts from local metric coordinate (YP,XP) in
to latitude & longitude decimal degrees:YDEG & XDEG
  740 YDEG=YDEG0+YP/(1852*60): REM There are 1852 m in one minute of latitude
  750 MULT=SGN(YDEG)*COS(RAD(YDEG)): REM  Multiplication factor for converting d
egrees longitude into metres based on cos of latitude.
  755 IF YDEG=0 THEN MULT=COS(RAD(YDEG))
  760 XDEG=XDEG0+XP/(MULT*1852*60)
  770 ENDPROC
```