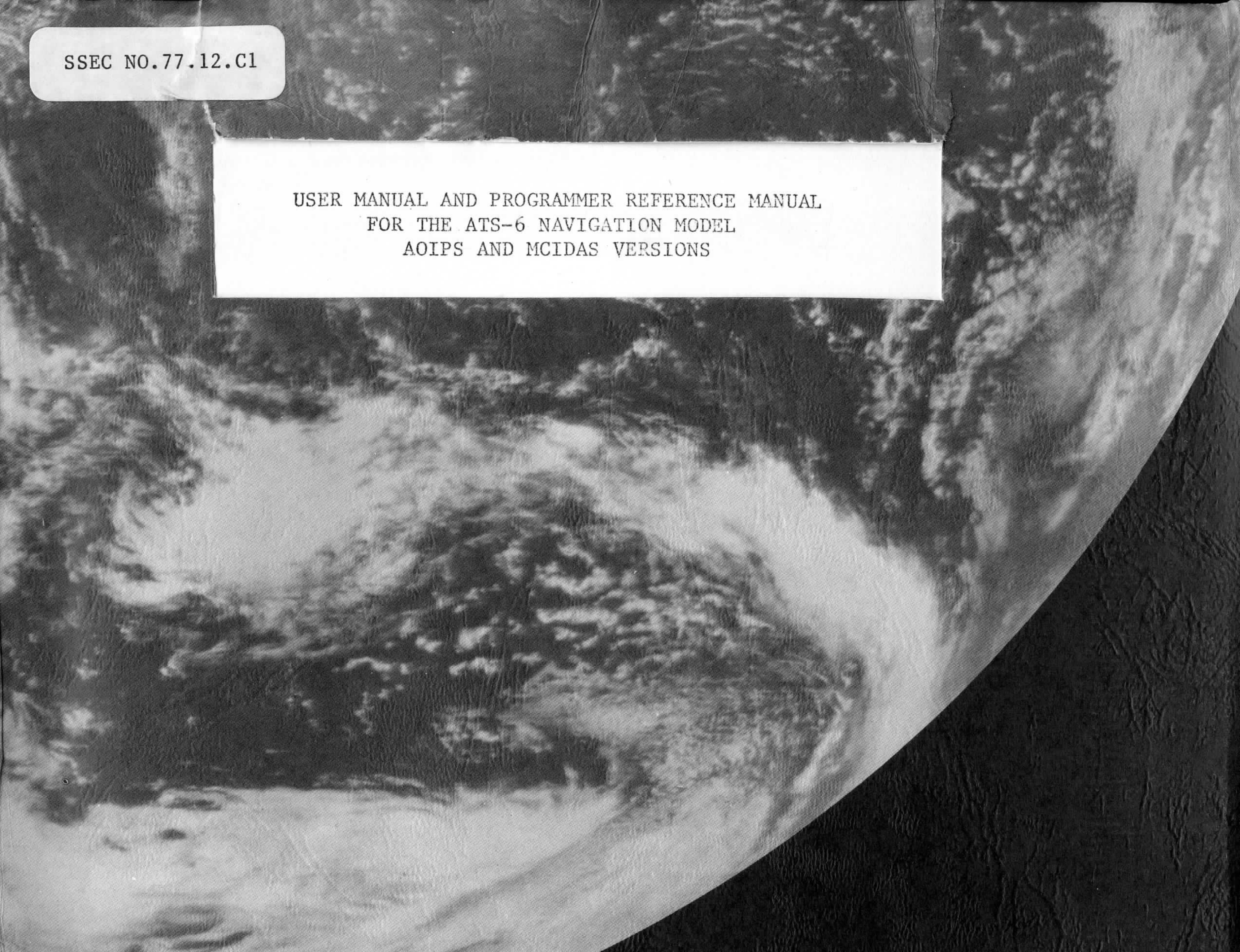
USER MANUAL AND PROGRAMMER REFERENCE MANUAL FOR THE ATS-6 NAVIGATION MODEL AOIPS AND MCIDAS VERSIONS



A REPORT

from the space science and engineering center
the university of wisconsin-madison
madison, wisconsin

USER MANUAL AND PROGRAMMER REFERENCE MANUAL
FOR THE ATS-6 NAVIGATION MODEL
AOIPS AND MCIDAS VERSIONS

Part II of Final Report for Period Ending 31 December 1977

Contract Number NAS5-20974

For
National Aeronautics and Space Administration
Goddard Space Flight Center
Greenbelt, Maryland 20771

By
G. C. Chatters
W. W. Kuhlow

University of Wisconsin
Space Science and Engineering Center
1225 West Dayton Street
Madison, WI 53706

December 1977

Table of Contents

Preface

This user and programmer reference manual together with the Progress Report for the Period Ending 31 October 1977 constitute the Final Report for Contract NAS5-20974. This manual describes the computer programs developed to implement the ATS-6 navigation model on both the AOIPS at GSFC and the McIDAS at SSEC. The above mentioned progress report describes the theory of the navigation model and the results obtained with it. Some of the Appendices of that report have been updated and included here.

Acknowledgements

As with most projects of this size, many people have contributed their efforts. At the Space Science and Engineering Center, Tom Haig helped initiate the project, Ralph Dedecker and Bruce Sawyer worked on the early stages of programming. Dennis Phillips helped with the orbit and attitude models. Dave Martin helped evaluate the wind fields produced by cloud tracking on the ATS-6 images. Rosanne Koehler typed this and previous reports.

At the Goddard Space Flight Center, in addition to the contract monitor, R. K. Squires, the project received considerable help from the AOIPS staff, especially C. T. Mottershead of CSC.

I. The ATS-6 Navigation Problem

A. Introduction-Objective

If a satellite image or sequence of images of the earth is to be useful for quantitative measurement, it is necessary to be able to determine the earth location (latitude and longitude) that corresponds to any given picture element (line and element) in that image or sequence of images. Development of a navigation system for a given satellite involves two procedures: 1) Defining an algorithm for converting a satellite picture element location to earth location and vice versa; 2) Defining a procedure for measuring the set of constants needed by the algorithm in 1) above. Previous progress reports for this project describe how the ATS-6 navigation model was developed. This user manual briefly describes the current version of the navigation model (this section) and how to use the computer programs developed for it (following sections).

In the process of developing a navigation model for satellite images we must establish criteria of validity for the resulting model. Since there is no way to precisely relate the image location to other satellite sensors, the criteria of validity must be referenced to some measurement derived from the image itself. For this work, two tests were used. First, the line and element position of identifiable earth points (landmarks) was measured and compared to the values predicted by the navigation model. Ideally these should always differ by less than one pixel. In general, this navigation model is capable of predicting positions within an error of the order of one pixel. Second, a set of cloud tracer winds was derived from a three-image time sequence. These winds were compared to measurements made on the same clouds in SMS-1 images from approximately the same time. Winds, averaged over approximately one hour, from each of the two satellites showed agreement on the order of one meter per second. In the following sections we describe the model used in making these measurements.

B. Nature of the Problem

A three-axis stabilized satellite contains an attitude measuring and correcting system which attempts to keep the satellite and its camera pointed at some selected point. For ATS-6 the camera is generally aimed at the point on the earth's equator which is closest to the subsatellite point. In order to keep the satellite camera pointing in its desired direction within its required range of accuracy, the satellite's attitude may be changed by significant and unpredictable amounts about all three axes several times during the scanning of an image. If wind speeds are to be measured accurately, the attitude changes must be measured and accounted for.

In addition to attitude changes of the satellite, we find that we must also account for image distortions caused by the scanning mechanism. We refer to this problem as mirror-scan nonlinearity. Ideally the images would be generated by sampling at equal-angle intervals (angular position a linear function of picture element number) as the camera's mirror scans across the earth. However, the ATS-6 camera scans in both left to right and right to left directions and therefore the camera's scanning mirror must change direction and accelerate for each scan. Two resulting effects are evident in the images: scan lines of opposite direction are offset by 7 to 11 pixels and each scan line varies from equal-angle sampling by 2 or 3 pixels maximum over a 200 pixel range.

C. The Solutions to the Problems

In this section we describe the methods currently used for attitude determination and for correcting mirror-scan nonlinearity in ATS-6.

The satellite has attitude sensors and data from these sensors are recorded on magnetic tape along with the image data. Unfortunately, we found that these attitude telemetry data did not accurately reflect changes in satellite attitude as seen in the images. As a result we developed a method to correct for attitude changes using the images themselves. This is the earth edge displacement technique described in Section III of reference 3. These earth edge displacement measurements enable us to compute the changes in attitude with time. We must also compute a reference attitude, usually the mean attitude for the first image in a sequence of three to be studied. This reference attitude is computed from landmark measurements; the technique is described in references 1 and 2.

The problem of mirror-scan nonlinearity cannot be solved completely, but a correction scheme adequate for our purpose of cloud tracking has been developed. Ideally, we would like to define some reference coordinate system to use in converting the images to equal-angle sampling. Unfortunately, we have no way to define such a coordinate system which would be independent of the images. Thus, we use one scan direction (odd numbered scans) as a reference and shift scans of the opposite direction to match. This method corrects the alternate scan offset but leaves an uncorrected nonlinearity of up to two to three pixels at some points in the image. This error is reasonably constant from one image to the next and is small enough and varies slowly enough that the images may still be used for cloud tracking.

II. Using the ATS-6 Navigation System

A. Introduction

This section gives a general discussion of the use of the ATS-6 image correction and navigation software. For specific commands and data inputs to the various programs see Section III. The general organization of the main programs is indicated in Fig. II.1.

B. Image Correction

The alternate scan element shift function $\Delta E(E)$ which is used to correct for mirror-scan offsets (Appendix D) is computed from infrared image data. The program OFSTG computes values of ΔE and a weighting function at points at equal intervals across the image. This procedure is described in detail in Appendix D. Since the image is not viewed before running this program it is not known if some of the computed offset points will be computed for locations off the earth. Some offset points will be averages of computations from data entirely on the earth, some entirely off the earth and some mixed. Because the brightness off the earth is almost uniform, the weighting factor, which is the brightness range in the line segment used in the correlation computation, will cause the off-earth points to have negligible effect on the averaged offset. Data points for correlations done entirely from off-earth data must be eliminated from the curve-fitting process. To do this the user must scan through the listing of offset and weight values from OFSTG. There is a fairly sharp discontinuity at both earth edges. The element location of these edges are then input to program OFSTF.

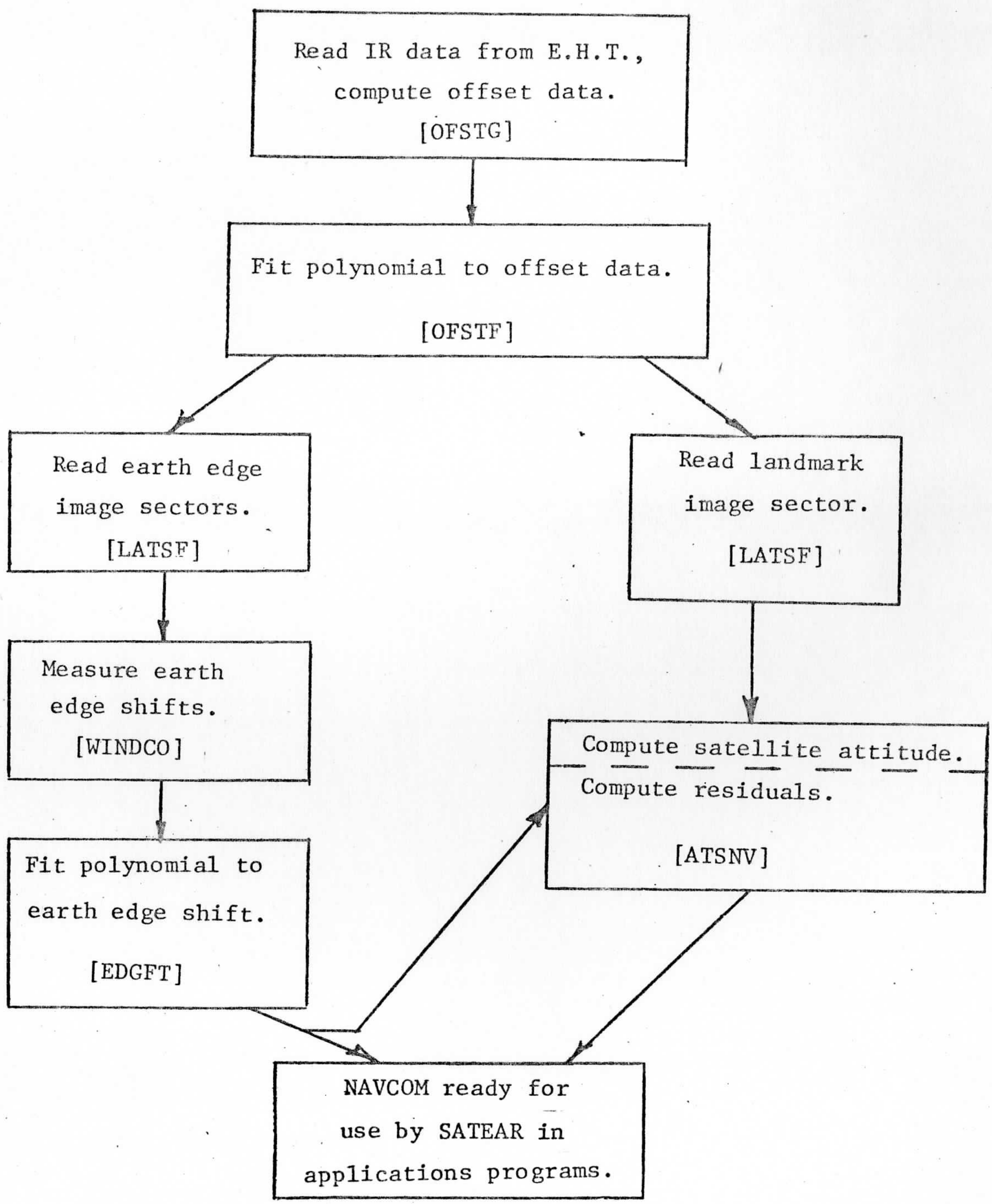


Figure II.1. General organization of ATS-6 image correction and navigation software. Names of corresponding computer programs in brackets.

The offset data points generated are not only discontinuous but also very noisy. Therefore we fit a smoothing and interpolating polynomial to the data points. This is done by program OFSTF. As stated above OFSTF must have the limits of valid data. The program OFSTF sets up arrays for subroutines APCH and APFS. These least-squares polynomial fitting subroutines are standard procedures from the IBM Scientific Subroutine Package (SSP).

Once the alternate scan correction polynomial has been determined, program LATSF may be used to read image segments from the tapes. This program requires input data specifying the date and time of the image for image identifier entries. This program reads a standard size image area which is 512 lines by 512 elements on AOIPS or 500 lines by 672 elements on McIDAS. The location of this segment with respect to the whole image is specified as input data in terms of line and element of the upper left corner of the image segment.

It should be noted here that this alternate scan correction method only shifts data from one scan direction to match that of the other. It does not correct the data to true equal-angle sampling.

C. Attitude Determination

The initial satellite attitude is determined from landmarks and the satellite's orbit. The attitude computation is done using several widely spaced landmarks on the first image (t_1) of the sequence to be used. A minimum of three landmark areas should be used. One or, preferably, more landmark points may be measured in each area. Landmarks should also be measured for other images in the sequence to serve as a check on the computations and on the stability of the attitude.

Attitude changes from one image to the next are determined from earth edge displacement measurements. These displacements are measured using the regular AOIPS or McIDAS cloud tracking program (WINDCO). For this procedure a pair of images at the earth edge are loaded (say t_1 and t_2). Then the wind tracking procedure using a correlation tracking metric is used to track the displacements of the edges. The line direction lag size is set to zero so that the correlation peak search is done by moving the target grid only horizontally (i.e. in element direction) within the search grid. This measurement is made approximately every 5 to 10 lines along the edge. It need only be done for the same range of lines as is covered by the area to be used for cloud tracking; but must be done for both edges. As in the case of the alternate scan data, these measurements form a discontinuous and slightly noisy function. The program EDGFT is used to fit a set of Chebyshev polynomials to the data points for each edge. The coefficients, number of coefficients, scaling factors and valid range of use in terms of time of day at a given line are stored in common block NAVCOM. Reference 3, especially Section III, gives more of a discussion of the earth edge correction techniques.

D. Measuring Cloud Tracer Winds

Once the attitude and earth edge displacement polynomial coefficients have been computed and placed in common block NAVCOM along with the orbit and frame constants, the coordinate transform functions are ready for use in cloud tracking. The functions available are SE for satellite image coordinate (line, element) to earth coordinate (latitude, longitude) transformations and ES for the inverse. For compatibility with SMS systems a subroutine SATEAR, which links to ES and SE is supplied.

III. Program Command Format

This section describes the command formats for using the ATS-6 navigation programs. There are two versions, one on the AOIPS and one on the McIDAS. Part A covers the use of the AOIPS version. AOIPS uses a prompting system; the appropriate responses to each of the prompting requests are explained. McIDAS uses a command input system; the command sequence and input parameters are explained.

A. AOIPS Version of November 1977

There are currently six main programs plus the main menu driver for ATS-6 processing on AOIPS. These are set up to work with the METPAK driver MET2 or Terminal 2. The system is started up by mounting the METPAK disk on RDØ:, then installing the tasks. The main menu driver expects them to have a name ending in 2 (e.g. LATSF2). The installation command file RDØ:[1,162]ATSINS.CMD is available. Operation must be initiated by first running METPAK to restore the global common. Once the common has been restored, the EXIT option is taken and the ATS-6 menu driver ATSF2 is initiated (i.e. RUN DBØ:[35Ø,62]ATSF2). The system is now ready to run.

A. AOIPS Version of November 1977

1. A6INT - Initialize COMMON/NAVCOM/

Initiate by requesting option: 1 - INITIALIZE NAVCOM on the ATS-6 PROCESSING menu.

Input request:
NAVCOM TO BE PRINTED? (1 = YES, DEFAULT = NO)

Response: If a 1 is entered, the updated contents of COMMON/NAVCOM/ will be output to the line printer just before A6INT exits. Otherwise, not.

Input request:
ENTER DAY NUMBER

Response: Enter the Julian day number (1-365) of the data to be worked with. Year is assumed to be 1974.

Input request:
ENTER 3 PICTURE START TIMES - HHMMSS

Response: Enter the picture start time, to nearest second, for the three images to be worked with. As currently set up COMMON/NAVCOM/ can only hold enough coefficients for earth edge correction for three image times.

Input request:
ENTER FIRST ORBIT POSITION: T(HHMMSS), X, Y, Z (KM)

Response: Enter a satellite orbit position vector as read from orbit data on the experimenter history tape. T is the time of the position in packed integer format (hours, minutes, seconds), X, Y, Z are the location in the earth inertial co-ordinate system in floating point kilometers.

Input request:
ENTER SECOND ORBIT POSITION: T(HHMMSS), X, Y, Z (KM)

Response: Same as above for a second satellite orbit position.

Default: Except as noted, the default response causes a return to the ATS-6 processing menu.

Program output: A6INT will enter values into COMMON/NAVCOM/ which is part of the saved global common.

2. OFSTG - Generate alternate scan offset data points.

Initiate OFSTG by requesting option:
2 - GENERATE ELEMENT OFFSET DATA FROM E.H.T.
from the ATS-6 PROCESSING MENU.

Set up: Mount an ATS-6 Experimenter History Tape on tape drive MMØ or MM1.

Input request:
MOUNT TAPE. ENTER DRIVE NUMBER.

Response: After mounting the tape, enter a Ø or 1 for MMØ or MM1 respectively.

Default: The default is tape unit MMØ.

Program output: OFSTG will print a table of element numbers, offsets and weights on the line printer. These same values are stored in global common in COMMON/BUFFER/ and COMMON/BUFF1/ for use by program OFSTF.

Note: This program is a heavy user of CPU cycles.

3. OFSTF - Fit a set of Chebyshev polynomials to the data points generated by program OFSTG.

This program is automatically initiated at the end of program OFSTG. It may also be initiated by requesting option 3 - CURVE FIT TO OFFSET DATA.

Input request:
ENTER LEFT, RIGHT ELEMENTS FROM PRINTER LISTING.

Response: The printer output of offset values should show a fairly continuous variation in the mid-portion of the element range with a noticeable discontinuity near each end. Enter the element number of the points at each end of the continuous midsection of the data. For example, for day 74195 164223Z data these values were 180 and 2270.

Default: On default entry the system returns to the ATS-6 PROCESSING menu.

Program output: The program fits a set of Chebyshev polynomials to the data points and store the coefficients in the global COMMON/NAVCOM/.

4. LATSF - Read image segments from ATS-6 experimenter history tapes.

Initiate by requesting option 4 - READ IMAGE SEGMENT FROM E.H.T. The programs A6INT, OFSTG, and OFSTF should have been run to set up COMMON/NAVCOM/ entries. Mount the E.H.T. on tape drive MMØ or MM1.

Input request:
ENTER PICTURE TIME (HHMMSS)

Response: Enter the picture start time in packed hours, minutes, seconds format. This must be exactly the same as one of the three entries made while initializing NAVCOM. (See A6INT above.)

Input request:
ENTER START LINE AND ELEMENT

Response: Enter the image coordinates for the upper left hand corner of the desired image.

Input request:
ENTER ZOOM AREA (1-7)

Response: Select a number from 1 to 7 to designate this region of the earth. Use the same number for a given area for all three picture times.

Input request:
ENTER TAPE UNIT (Ø,1)

Response: Enter Ø or 1 according to which drive the tape is mounted on (MMØ or MM1).

Default: If a default entry is made, the system returns to the ATS-6 PROCESSING menu.

Program output: This program loads 512 x 512 element image segments onto digital disk files. Both visible and infrared data are loaded for each request.

5. EDGFT - Fit a set of Chebyshev polynomials to earth edge measurements stored in the wind file.

   Initiate this program by requesting option 5 - CURVE FIT TO EARTH EDGE DATA. Earth edge measurements should be stored in the wind file. There are no requests for input.

   Program output: This program fits a set of Chebyshev polynomials to left and right earth edge data. The coefficients are stored in COMMON/NAVCOM/.

6. ATSNV - Compute a nominal attitude from landmarks.

   Initiate this program by requesting option 6 - RUN NAVIGATION. Landmark measurements from ATS-6 should be stored on the landmark file. (Currently this program reads test landmarks from the file DBØ:[350,62]A6LMKS.DAT., however, the program should be modified before general use.) There is no request for keyed in data.

   Program output: This program computes yaw, pitch and roll values and stores them in global COMMON/NAVCOM/. In addition these values are displayed on the operator terminal. The landmarks and computed residuals are output to the line printer.

B. McIDAS Version of November 1977

1. Set up and initializing COMMON/NAVCOM/.

The current ATS-6 navigation system uses the files OFFSTD and ATSCOM. These two twenty sector files should be created before attempting to use the McIDAS ATS-6 navigation.

The navigation program should be run to initialize COMMON/NAVCOM/ and to set the three picture start times used for earth edge corrections. To zero NAVCOM enter:

AN Ø Ø Ø Ø NEW

To set picture start times enter:

AN Ø Ø Ø Ø $HHMMSS_1$ $HHMMSS_2$ $HHMMSS_3$

Orbit, frame geometry and scan period are to be stored in the McIDAS navigation file using standard McIDAS commands DQ, DS and ON. Entries are:

DS SSYYDDD scan period (μsec)
(e.g. DS 1474195 1200000)

DQ FIRST SSYYDDD HHMMSS X_1 Y_1 Z_1 (decameters)
DQ SECOND SSYYDDD HHMMSS X_2 Y_2 Z_2 (decameters)

(e.g. DQ FIRST 1474195 175531 -1198560 4041970 -43760)
DQ SECOND 1474195 164223 133380 4214050 -5950)

ON SSYYDDD Line-angle Total-lines Element-angle Total-elements
Angle are in ±DDDMMSS format
(e.g. ON 1474195 195512 2400 200412 2400)

2. Generate alternate scan offset data points.

Set-up: Mount an ATS-6 experimenter history tape on a tape drive. Enter: MT ‸ 14 ‸ Ø.

Running program: Enter the two letter keyin (currently GC). There are no parameters.

Program output: This program will print a table of element numbers, offsets and weights on the line printer. These same values are stored in the file OFFSTD.

3. Fit a set of Chebyshev polynomials to the data points stored in file OFFSTD.

Set-up: The program to generate the data points must have been run first (see 2 above).

Running program: Enter the two letter keyin (currently GC). The parameters are the left and right ends of the valid element range. These values are determined from the printer listing of offset values generated in 2 above.

Program output: This program does a least squares fit of a set of Chebyshev polynomials to the alternate scan data. The coefficients are stored in COMMON/NAVCOM/.

4. Read image segments from ATS-6 experimenter history tapes into digital areas.

Set-up: Mount tape. COMMON/NAVCOM/ should have been set up by previous programs.

Running program: Enter the two letter keyin and parameters: BK SSYYDDD HHMMSS Area Line Element

Program output: This program loads a visible, infrared or combined area onto digital disk areas. Only standard size areas (500 x 672) may be used.

5. Fit a set of Chebyshev polynomials to earth edge measurements stored in the wind file.

Set-up: Use WINDCO with image coordinates (WC I) and line lag size zero (LS Ø X) to measure earth edge displacement of both left and right earth edges between times $t_1 - t_2$ and $t_1 - t_3$.

Running program: Enter the two letter keyin (currently GC) to initiate the program. There are no parameters.

Program output: This program computes coefficients for a set of Chebyshev polynomials for left and right earth edges. These coefficients are stored in COMMON/NAVCOM/.

6. Compute attitude from landmarks (navigate).

Set-up: Several landmark measurements should have been made from the ATS-6 images and stored in the regular McIDAS navigation file. The t_1 landmarks, from at least three different locations, should have computation code Ø. Landmarks for later times should use code 3ØØ and be used as a check on the navigation.

Running program: Enter the keyin and parameters:

AN SSYYDDD Ø Ø (P)

Program output: This program computes a satellite attitude and stores the yaw, pitch and roll values in COMMON/NAVCOM/ and, via an SQ call to DX, in the McIDAS navigation file.

IV. Software Internal Description

This section contains descriptions of the computer programs and subroutines developed for the ATS-6 navigation model. These programs are available on NASA's AOIPS, on SSEC's McIDAS and most are also available on the University of Wisconsin's Univac 1110. There are some variations in the main programs to allow for peculiarities of each system. The code for the tape read subroutines and for subroutines CRKTHR and CRKATS are unique to each system but yield identical results. Subroutines APCH, APFS and CNPS are from IBM's Scientific Subroutine Package (SSP) and are not documented here. Appendix E of this report contains source code listings for the McIDAS and AOIPS versions of the main programs and for most subroutines. Fig. IV.1 illustrates the coordinate system used in these programs. This section contains three parts:

A. Description of procedure used by main programs

B. Entries in common block NAVCOM

C. Subroutine function descriptions

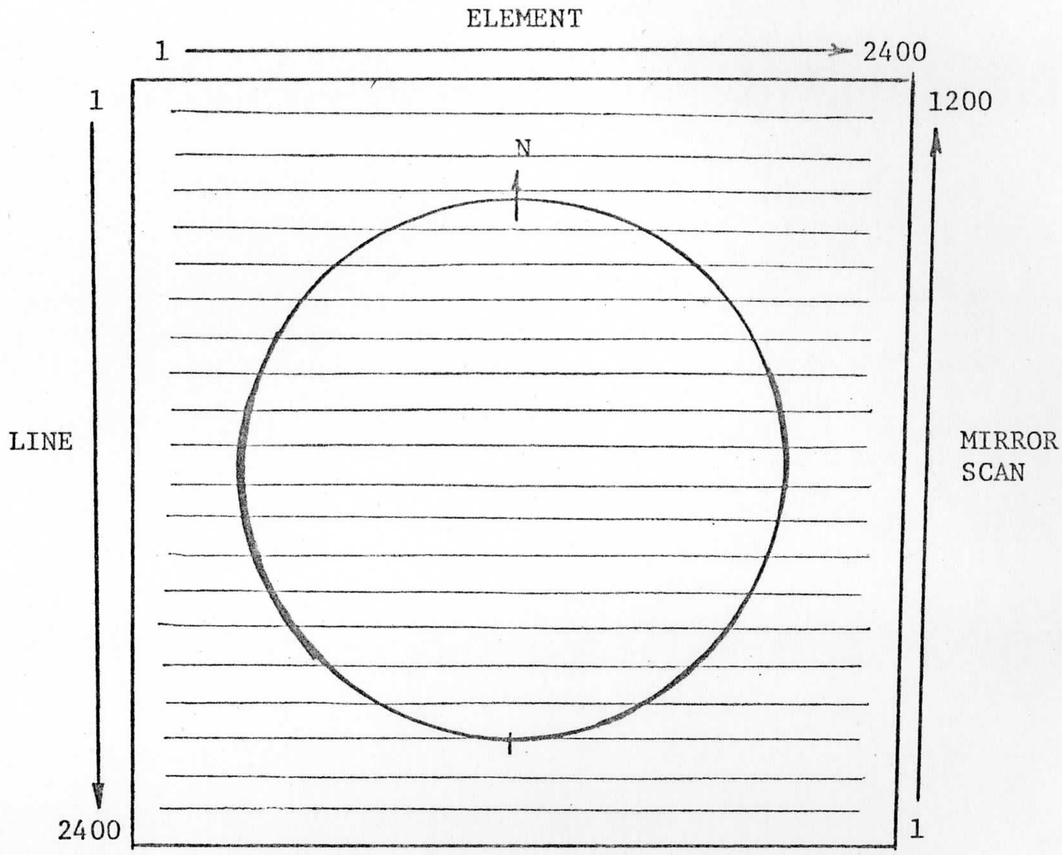


Figure IV.1. ATS-6 Image Coordinate Systems. The ATS-6 satellite scans from south to north with 1200 scans per image. In a visible image each scan consists of 2 lines for a total of 2400 lines per image. The satellite coordinate lines used are numbered 1 to 2400, north to south to be consistent with the convention for SMS images. Satellite coordinate elements are numbered 1 to 2400 left to right. Infrared image data only have one line per scan (but 2400 elements per line). Infrared lines are repeated on image sector loads by LATSF to keep the aspect ratio 1:1 and to keep the coordinates the same for visible and infrared.

A. This section gives a description of the procedure executed by each of the main programs.

A6INT - Initialize NAVCOM

1. Make selected entries in COMMON/NAVCOM/.
2. Print all entries in COMMON/NAVCOM/.
3. If two satellite orbit positions entered, call GASORB to convert to position and velocity.
4. Note: This program only used on AOIPS.

OFSTG - Generate offset data

1. Advance to first even numbered scan to be used (scan 800) by:
   a) Read tape record (IOTPIN)
   b) Crack out scan number (CRKTHR)
   c) If desired scan missed print message and modify, start scan number
   d) Loop back to a.
2. Check to see that following record contains next lower numbered odd scan. If not, modify desired even scan number and return to 1.
3. Desired even and odd scans found. Back up and read whole records into arrays and crack out IR brightness values (CRKATS).
4. Compute offsets for this scan pair. See report of 31 October 1977 Appendix G for details.
5. Add computed offsets and weights to accumulated values for each element position.
6. Print table of element numbers, offsets, and weights.
7. Store number of points and element numbers in common block BUFF1. Store weights and offsets in common block BUFFER. On McIDAS these values are stored in the file OFFSTD.

OFSTF - Fit polynomial for offset data

1. Pick up valid element range as input.
2. Transfer selected range of element numbers, with corresponding offsets and weights to array DATI.
3. Call subroutine APCH to set up matrix for least squares fit.

4. Call subroutine APFS to invert matrix and compute coefficients for least squares set of Chebyshev polynomials.
5. Store coefficients, scaling factors, number of coefficients and valid element range in common block NAVCOM.

LATSF - Load ATS-6 image segment

1. Input information on data request (coordinates, etc.)
2. Set up image label and write to disk.
3. Call subroutine GENOFF to set up a table of offsets, for every element position to be read in, by evaluating the set of Chebyshev polynomials stored in common block NAVCOM.
4. Advance to first data record.
   a) Skip two header records
   b) Read a partial record from tape
   c) Call CRKTHR to crack out scan number
   d) If desired start scan not reached, go back to b.
5. Back up one record so entire record can be read.
6. Read image segment
   a) Read a record.
   b) Check scan number. If less than last scan to be read in, terminate image load.
   c) Pass Visible-2 data to subroutine LINGRB to select and repack desired line segment.
   d) Write image line segment to digital disk area.
   e) Pass Visible-1 data to subroutine LINGRB to select and repack desired line segment.
   f) Write image line segment to digital disk.
   g) Pass Infrared data to subroutine LINGRB to select and repack desired line segment.
   h) Write infrared image line to digital disk.
   i) Write infrared image line to digital disk a second time so that visible and infrared coordinates match.
7. Done now; rewind tape.

EDGFT - Fit polynomials to earth edges

1. Do curve fit for $t_1 - t_2$ image pair then for $t_1 - t_3$ image pair.
   a) Read a wind from disk file.
   b) Check for valid year, day, times, error code. If invalid, return to step a.
   c) If element position of vector less than picture center element, store scan number, shift in array for left edge.
   d) If element position of vector greater than picture center element, store scan number, shift in array for right edge.
   e) Loop back to a till end of wind file encountered.
   f) Fill array DATI with scan numbers, shifts for left edge. Pass array to subroutine APCH to set up matrix.
   g) Pass matrix from APCH to subroutine APFS to invert matrix and compute coefficients for set of Chebyshev polynomials.
   h) Store coefficients, scaling factors in common block NAVCOM.
   i) Repeat steps f, g, h for data from right edge.
   j) Determine valid argument range for left and right edges. Determine the overlapping portion of valid argument range for left and right edges and store this overlapping portion in common block NAVCOM.

ATSNV - Navigate ATS-6 image from landmarks

1. Read landmarks from landmark file.
2. Convert integer values to floating point.
3. Pass landmark data to subroutine ATTTUD to compute satellite attitude.
4. The McIDAS version stores the attitude in the navigation file by SQing DX.
5. Compute and list residuals.
   a) Pass picture time, latitude, longitude of landmark to subroutine ES to compute line and element.
   b) Compute residual equals measured value minus computed value for lines and for elements.
   c) List values and loop back to a through all landmarks.

B. Entries in common block NAVCOM.

NAVN	Navigation sequence number.
INAV	Flag to indicate type of navigation.
IYR	Year of date for which navigation is valid.
IDAY	Day of year for which navigation is valid.
TOTLIN	Total number of lines in image (= 2400.).
DEGLIN	Total sweep angle in line direction (= 19.92 degrees).
TOTIEL	Total number of elements across image (= 2400.).
DEGELE	Total scan angle in element direction (= 20.07 degrees).
PICLIN	Picture center line.
PICELE	Picture center element.
TMPSCL	Scan period (nom. .02 minutes).
IOYR	Year of date for orbit values (IOYR = IYR).
IODAY	Day of year for orbit values (IODAY - IDAY).
TM	Time of orbit location (minutes, GMT).
R1X	X component of location of time TM (earth radii).
R1Y	Y component of satellite location of time TM (earth radii).
R1Z	Z component of satellite location of time TM (earth radii).
R1DX	X component of satellite location of time TM (earth radii/minute).
R1DY	Y component of satellite velocity at time TM (earth radii/minute).
R1DZ	Z component of satellite velocity at time TM (earth radii/minute).
PITCH	Pitch angle of rotation from BC to PF coordinates. Satellite attitude (radians).
ROLL	Roll angle of rotation from BC to PF coordinates. Satellite attitude (radians).
YAW	Yaw angle of rotation from BC to PF coordinates. Satellite attitude (radians).

PTIM(3)	Picture start times for the three images for which earthedge correction applies. PTIM(1) is the time of the reference image. (Time in minutes, GMT)
TMN(3)	Minimum time of time over which earthedge correction valid. Subscript corresponds to image number as in PTIM.
TMX(3)	Maximum time of time over which earthedge correction valid. (e.g. earth edge correction may be used for scan time t within image starting at PTIM(2) only if TMN(2) $\leq$ t $\leq$ TMX(2). Units are minutes, GMT. PTIM(i) $\leq$ TMN(i) $\leq$ TMX(i).

Note: For earthedge correction arrays, dimensions of value 2 refer to image pair. 1 => PTIM(1) - PTIM(2) shifts; 2 => PTIM(1) - PTIM(3) shifts.

NLCOEF(2)	Number of coefficients in polynomial for left edges.
NRCOEF(2)	Number of coefficients in polynomial for right edge.
SCLLØ(2)	Offset for scaling argument value (scan number) for left edges.
SCLL1(2)	Multiplier for scaling argument value (scan number) for left edges.
ELCOEF(11,2)	Coefficients of set of Chebyshev polynomial for left edges.
SCLRØ(2)	Offset for scaling argument value (scan number) for right edges.
SCLR1(2)	Multiplier for scaling argument values (scan number) for right edges.
ERCOEF(11,2)	Coefficients of set of Chebyshev polynomial for right edge.
NASCEF	Number of coefficients in polynomial for alternate scan correction.
SCLASØ	Offset for scaling argument (scan number) for alternate scan offset.
SCLAS1	Multiplier for scaling argument value (scan number) for alternate scan correction.
IELEMN	Minimum element number for which alternate scan correction applies.

IELEMX	Maximum element number for which alternate scan correction applies.
ASCOEF(16)	Coefficients fo set of Chebyshev polynomials for alternate scan correction.

C. Subroutine Function Description

Name: BCTOPF

Call: CALL BCTOPF (X, Y, Z, IDIR)

Input Parameters:

X, Y, Z — Satellite orientation in body centered (IDIR = 1) or picture frame (IDIR = 2) coordinates

IDIR — Direction to rotation
IDIR = 1 Body centered to picture frame
IDIR = 2 Picture frame to body centered

Returned Values:

X, Y, Z — Satellite orientation in picture frame (IDIR = 1) or body centered (IDIR = 2) coordinates.

Algorithm:

BCTOPF uses subroutine ROTATE to create a rotation matrix from the yaw, roll and pitch angles computed by the navigation program. It then multiplies the vector (X, Y, Z) by the matrix (IDIR = 1) or its transpose (IDIR = 2).

Reference:

For discussion of the coordinate systems see "Design and Testing of the Navigation Model for Three Axis Stabilized Earth Oriented Satellites Applied to the ATS-6 Satellite Image Data Base" progress report for 17 Nov. 1975, appendix pp. 2-4 or progress report for 31 June 1976, appendix pp. 2-4.

Name: CRKATS

Call: CALL CRKATS (N, S, D)

Input Values:

N - Number of data words to crack out.

S - Source array of 12-bit words stored in sequence as a continuous bit string after being read from tape.

Returned Values:

D - Destination array of full words.

Function:

ATS-6 experimenter history tapes are written with the image data stored in the 9 least significant bits of successive 12-bit words. CRKATS extracts the 8 most significant data bits (bits 8-1 of a 12-bit word numbered 11-0) and stores the resulting data value in a full computer word.

Name: CRKTHR

Call: CALL CRKTHR (N, S, D)

Input Values:

N - Number of 12-bit words to crack out.

S - Source array of 12-bit words stored in sequence as a continuous bit string after being read from tape.

Returned Values:

D - Destination array of whole words.

Function:

ATS-6 experimenter history tapes are written with the data stored in 12-bit words (actually thirds of 36-bit words). CRKTHR places these 12-bit words in whole words (16-bits for PDP-11).

Name: EDGCOR

Call: CALL EDGCOR (PTIME, ALIN, DELLIN, DELELE)

Input Parameters:

PTIME - Picture start time (minutes, GMT).

ALIN - Line coordinate value at which correction is to apply.

Returned Values:

DELLIN - Line correction value.

DELELE - Element correction value.

Function:

EDGCOR requires that left and right earth edge shift polynomials be stored in COMMON/NAVCOM/. The current version only allows for three picture start times to be in use at once.

EDGCOR evaluates the earth edge shift polynomials for left and right edges, then converts these values to line and element shifts. The polynomial only applies to the line range for which earth edges were measured. Outside this range, the value zero will be returned.

Name: ERTOER

Call: CALL ERTOER (XLAT, XLON, SE, YE, ZE, IDIR)

Parameters:

XLAT	Latitude (degrees) of a point on the earth.
XLON	Longitude (degrees) of same point on the earth.
XE, YE, ZE	Cartesian coordinates in coordinate system rotating with earth (kilometers).
IDIR	Direction of transformation.

Function:

Converts coordinates of a point on the earth from latitude, longitude to rotating Cartesian coordinates (IDIR = 1) or vice versa (IDIR = 2).

References:

See progress report of 17 Nov. 1975 or 31 June 1976, appendix pp. 2-4 for coordinates.

See also subroutine ERTOST.

Name: ERTOST

Call: CALL ERTOST (XE, YE, ZE, X, Y, Z, IDIR, TIME)

Parameters:

XE, YE, ZE	Cartesian coordinates (kilometers) of a point on earth's surface in rotating coordinate system.
X, Y, Z	Unit vector in inertial coordinates pointing from satellite to point (XE, YE, ZE) on Earth's surface.
IDIR	Direction of transformation IDIR = 1 => (XE, YE, ZE) → (X, Y, Z) IDIR = 2 => (X, Y, Z) → (XE, YE, ZE)
TIME	Time of day (minutes, GMT)

Function:

ERTOST a unit vector pointing from the satellite to a given point on the earth (IDIR = 1) or given a pointing vector it computes the location, if any, on the earth's surface that the vector is pointing at (IDIR = 2).

Reference:

See progress report of 17 Nov. 1975 or 31 June 1976, appendix pp. 2-4 for coordinate.

Related subroutine ERTOER, STTOLV.

Name: ES

Call: CALL ES (PTIME, XLAT, XLON, XLIN, XELE)

Input Parameters:

PTIME - Picture start time (minutes, GMT).

XLAT - Latitude (degrees) of a point on the earth's surface.

XLON - Longitude (degrees) of that point.

Returned Values:

XLIN - Line coordinate in the image picture frame coordinate system.

XELE - Element coordinate in image picture frame coordinate system.

Function:

Subroutine ES does an earth (latitude, longitude) to satellite (line, element) coordinate transform based on the satellite's attitude, orbit position, and, if available, attitude correction based on measurement of earthedge shifts.

References:

See subroutine SE.

Name: FLALO

Call: XLAT = FLALO (ILAT)

Input Value:

ILAT = Latitude (or longitude) integer value in the format ±DDDMMSS.
(For PDP-11 an INTEGER*4 value.)

Return Value:

XLAT = Latitude (or longitude) in degrees (floating point).

Function:

Converts an angle stored as an integer in degrees, minutes, seconds format to floating point degrees.

Name: FLIP

Call: CALL FLIP (A, B, I, N. ALTRET)

Parameters:

A - an NxN matrix

B - an NxN matrix

I - row on which to perform operation

N - dimension of A and B

ALTRET - flag indicating an error (LOGICAL)

Return:

A, B are returned in modified form.

Function:

All rows greater than I are added to row I. The same operation is performed on matrices A and B.

Reference:

This subroutine is used only by subroutine INVERT.

Name: FTIME

Call: TIME = FTIME (ITIME)

Input Parameters:

ITIME = Integer time of day in the form HHMMSS. (INTEGER*4 on the PDP-11.)

Returned Value:

TIME = Time of day in minutes (floating point).

Function:

Converts a time of day in the packed integer format hours, minutes, seconds to time of day in minutes (floating point).

Name: GASORB

Call: CALL GASORB (R1, T1, R2, T2)

Input Parameters:

R1 Position vector of satellite at time T1 in earth inertial reference frame.

T1 Time (minutes, GMT) at which satellite is a position R1.

R2 Position vector of satellite at time T2.

T2 Time (minutes, GMT) of position R2.

Returned Values: Note - results are stored in COMMON/NAVCOM/.

TM - Time of position. TM = T1

R1X, R1Y, R1Z - Position of satellite at TM.
(R1X, R1Y, R1Z) = R1/RE
Where RE = radius of earth.

R1DX, R1DY, R1DZ - Velocity of satellite at time TM.

Function:

Given two position vectors and their corresponding times, GASORB computes the position and velocity of the satellite at the time of the first given position vector. The method used in an f,g series from the method of Gauss.

Reference:

Escobal, P. R. Methods of Orbit Determination, John Wiley & Sons, 1965, pp. 196, 197.

See also subroutine ORBIT.

Name: GENOFF

Call: CALL GENOFF (IUELE, NELES)

Input Parameters:

IUELE	First element of offset array.
NELES	Number of elements in offset array.

Function:

The subroutine GENOFF evaluates the alternate scan correction polynomial for all element values across the image to be read in. The values are stored in an array in COMMON/OFFSET/. This array is then used by the program which loads ATS-6 images (LATSF or LDATSF).

References:

See information on programs LATSF and OFSTF and on COMMON/NAVCOM/.

Name: INVERT

Call: CALL INVERT (AA, B, N, ALTRET)

Input Parameters:

AA - an NxN matrix

N - dimension of AA and B

Return Values:

B - the inverse of AA

ALTRET - a flag to indicate AA is singular (LOGICAL)

Function:

INVERT returns in B the NxN inverse of the matrix AA. If AA is singular, ALTRET is set to .TRUE.

Reference:

See also subroutines FLIP, MINMIZ.

Name: LINGRB

Call: CALL LINGRB (INDATA, ISDIR, IELE, NELES, IBDF, IOUTD)

Input Parameters:

INDATA	An array containing the bit string for either the Visible-1, Visible-2 or Infrared sensor as read from tape. Array actually starts 72 bits before first data word.
ISDIR	Indicates scan direction. 0 => Even numbered scan. 1 => Odd numbered scan.
IELE	First element of desired line segment.
NELES	Number of elements in line segment.
IBDF	Sampling factor. (IBDF = 1 only)

Returned Values:

IOUTD	Output array with data packed one pixel per 8-bit byte.

Function:

This subroutine unpacks pixel data from the bit string read from magnetic tape. It then selects out the desired line segment (512 elements on AOIPS), shifts the even scans based on the evaluation of the alternate scan correction polynomial, and stores the pixels in an array to be written in the image file.

References:

See main program LATSF.

Name: LS

Call: CALL LS(X, Y, VAL, DD, DIR)

Input Parameters:

X Starting point for line search (a vector).

DD Unnormalized directional derivative $\overrightarrow{\nabla S(Y)} \cdot \overrightarrow{DIR}$.

DIR Direction to do search (a vector).

Returned Values:

Y The selected point (a vector).

VAL Value of the objective function S evaluated at Y.

Function:

This routine performs an Armijo line search from the point "X" in the direction "DIR" and returns the selected point in "Y" and the objective function value S(Y) in "VAL". On call, "DD" is the unnormalized directional derivative $\overrightarrow{\nabla S} \cdot \overrightarrow{DIR}$. This line search routine returns in Y the point $X + 2^{-N}*DIR$ where N is the least nonnegative integer such that $-S(2^{-N}*DIR)$ represents at least 40% of the functional drop in the linearization of S at X in moving from X to $X + 2^{-N}*DIR$.

References:

See also subrountine MINMIZ and function S. See the report for 17 Nov. 1975 Appendix section IV or report for 31 June 1976 Appendix section II.C.

Name: MINMIZ

Call: CALL MINMIZ (PTIN, PTOUT, GNORM, VAL, ITN)

Input Parameter:

PTIN Starting point for search for minimum value of objective function (PTIN is a vector)

Returned Values:

PTOUT Optimal point. The objective function has a minimum at PTOUT. (PTOUT is a vector)

GNORM The norm of the gradient of the objective function at PTOUT.

VAL The value of the objective function at PTOUT.

ITN Number of iterations done.

Function:

MINMIZ finds the point PTOUT at which an objective function is a minimum. In this case PTOUT is the satellite's attitude (PITCH, ROLL, YAW). The objective function is defined in the reports (report of 17 Nov. 1975 Appendix eqn. 15, report of 31 June 1976 Appendix eqn. 17). Basically MINMIZ serves as a driver for PRTIAL, INVERT and LS.

References: See also subroutine ATTTUD.

Name: NRMLIZ

Call: CALL NRMLIZ (VX, VY, VZ, VNORM)

Input Parameters:

VX, VY, VZ   Cartesian components of any vector.

Returned Values:

VX, VY, VZ - Normalized components of the input vector.

VNORM - Length of the input vector.

Function:

NRMLIZ computes the length of the vector with components (VX, VY, VZ) then divides each component by that length to return a unit vector.

Name: ORBIT

Call: CALL ORBIT (X, Y, Z, T)

Input Parameters:

T - Time of day (minutes, GMT)

Returned Values:

X, Y, Z - Position of satellite at time T (kilometers)

Function:

Given the position and velocity of the satellite at some reference time as computed by GASORB, the subroutine ORBIT computes the satellite's position at the time T.

Reference:

Escobal, P.R. Methods of Orbit Determination, John Wiley & Sons, 1965, pp. 427, 428.

Name: PFTOTC

Call: CALL PFTOTC (XLIN, XELE, X, Y, Z, IDIR, INIT)

Parameters:

XLIN	Line number of a point on the ATS-6 image.
XELE	The element number of that point.
X, Y, Z	Unit vector, in the picture frame coordinate system, pointing at location defined by (XLIN, XELE).
IDIR	Direction of coordinate transformation.
INIT	Initialization flag. Set INIT = 1 before first call.

Function:

PFTOTC converts from picture frame cartesian (X, Y, Z) coordinates to picture frame image (LINE, ELEMENT) coordinates (IDIR = 1) or vice versa (IDIR = 2).

References:

See progress report of 17 Nov. 1975 or 31 June 1976, appendix, pp. 2-4, for coordinates.

See also subroutine BCTOPF.

Name: PRTIAL

Call: CALL PRTIAL (PT, GRAD, HESS)

Input Parameters:

PT Point (vector) at which to evaluate GRAD and HESS.

Returned Values:

GRAD The gradient of the objective function, evaluated at PT.

HESS The hessian of the gradient function evaluated at PT.

Function:

PRTIAL computes values of gradient and hessian for a given function.

References:

See also subroutine MINMIZ. See the reports of 17 November 1975 Appendix page 10 ff or report of 31 June 1976 Appendix page 13 ff.

Name: ROTATE

Call: CALL ROTATE (A, R, IR, IDERIV)

Parameters:

A	A matrix.
R	An angle of rotation (radians)
IR	Axis number (1, 2, 3)
IDERIV	Derivative of rotation matrix.

Function:

This routine returns in "A" the product of the input matrix "A" and a matrix RM, where, if "IDERIV=1", RM represents a rotation through an angle "R" (in radians) about the axis "IR". IF IDERIV=2, the first derivative of RM is operated on A, and if IDERIV=3, the second derivative of RM is used.

References: See also subroutine PRTIAL.

Name: S

Call: SVAL = S (PT)

Input Parameter:

PT The point (a vector) at which S is evaluated.

Returned Value:

S The objective function which is minimized in computing the satellite attitude.

Function:

S is the objective function which is minimized by MINMIZ in computing the satellites attitude. See the report of 31 June 1976 Appendix equation 17 or report of 17 November 1975 Appendix equation 15.

References:

See also the subroutines MINMIZ and LS.

Name: SATEAR

Call: CALL SATEAR (PICTIM, XLIN, XELE, XLAT, XLON, ITYPE, INAV, BETAIN, BETDOT, ATFRAC)

Parameters:

PICTIM	Picture start time (minutes, GMT)
XLIN	Satellite image line (master coordinate)
XELE	Satellite image element (master coordinate)
XLAT	Latitude (degrees, +North, -South)
XLON	Longitude (degrees, +East, -West)
ITYPE	Type of conversion 1 => Satellite to earth 2 => Earth to satellite
INAV BETAIN BETDOT ATFRAC	Dummy variable used so call will match SMS version.

Function:

Subroutine SATEAR calls subroutines SE or ES.

Name: SE

Call: CALL SE (PTIME, XLIN, XELE, ALOT, ALON)

Input Parameters:

PTIME - Picture start time (minutes, GMT).

XLIN - Line number of a point on the ATS-6 image.

XELE - Element number of that point.

Returned Values:

ALAT - Latitude (degrees) of the point on the earth's surface at (XLIN, XELE).

ALON - Longitude (degrees) of that point.

Function:

Subroutine SE does a satellite (line, element) to earth (latitude, longitude) coordinate transform based on the satellite's attitude, orbit position, and, if available, attitude corrections based on measurement of earthedge shifts.

References:

See also subroutine ES.

Name: STTOLV

Call: CALL STTOLV (X, Y, Z, IDIR, TIME)

Parameters:

X, Y, Z	Unit pointing vector in the satellite inertial coordinate system or in the satellite local vertical coordinate system.
IDIR	Direction of transformation. Satellite inertial to local vertical (IDIR = 1) Local vertical to satellite vertical (IDIR = 2)
TIME	Time at which transform applies. (minutes, GMT)

Function:

STTOLV uses the satellite's position at time TIME, and converts a unit vector in the satellite inertial coordinate system to the satellite's local vertical system (IDIR = 1) or vice versa (IDIR = 2).

Reference:

See progress report for 17 Nov. 1975 or 31 June 1976, appendix, pp. 2-4.

See also subroutines ERTOST, LVTOBC, and BCTOPF.

Name: UNIT

Call: CALL UNIT (A)

Returned Values:

A - a 3x3 identity matrix

Function:

Subroutine UNIT returns a 3x3 identity matrix in array A.

References

1. Kuhlow, W. W., Design and Testing of the Navigation Model for Three Axis Stabilized Earth Oriented Satellites Applied to the ATS-6 Satellite Image Data Base, Progress Report for Period Ending 17 November 1975, Contract Number NAS5-20974, University of Wisconsin.

2. Kuhlow, W. W., G. C. Chatters, Design and Testing of the Navigation Model for Three Axis Stabilized Earth Oriented Satellites Applied to the ATS-6 Satellite Image Data Base, Progress Report for Period Ending 31 June 1976, Contract Number NAS5-20974, University of Wisconsin.

3. Kuhlow, W. W., G. C. Chatters, Design and Testing of the Navigation Model for Three Axis Stabilized Earth Oriented Satellites Applied to the ATS-6 Satellite Image Data Base, Progress Report for Period Ending 31 October 1977, Contract Number NAS5-20974, University of Wisconsin.

APPENDIX A

BASIC NAVIGATION MODEL APPLIED TO ATS-6 IMAGE DATA BASE

Contents

NOMENCLATURE

Coordinate Systems (C.S.)

EI = Earth-centered Inertial

ER = Earth-centered Rotating

LV = Local Vertical (satellite-centered)

BC = Body Centered (satellite-centered)

PF = Picture Frame (satellite-centered)

Orthogonal Matrices

R_{LV} = rotation into LV C.S.

R_{BC} = rotation into BC C.S.

R_{PF} = rotation in PF C.S.

R_A = optimized "navigation" matrix

$R(\theta,k)$ = rotation about axis k (k-1,2,3) in a ccw sense by an angle θ

Vectors

$\hat{r}$ = unit vector

$\vec{r}_s$ = satellite radius vector

$\vec{r}_1$ = landmark pointing vector

$\vec{r}_2$ = earth-coordinate, EI C.S.

$\vec{r}_E$ = earth-coordinate, ER C.S.

$\hat{r}_{LV}$ = pointing vector in LV C.S.

$\hat{r}_{BC}$ = pointing vector in BC C.S.

$\hat{r}_{PF}$ = pointing vector in PF C.S.

$\hat{e}_3 = (0,0,1)^T$

Other

$\theta_y, \theta_r, \theta_p$ = attitude yaw, roll, pitch angles

θ_L, ϕ_L = geodetic latitude, longitude

θ_E = sidereal time of Greenwich prime meridian

ρ_L, ρ_E = radians/line, radians/element

λ = mirror step angle

δ = mirror sweep angle

α_i = navigation parameter

$\bar{\alpha}_i$ = optimized navigation parameter

LIN, ELE = satellite image coordinates: Line, Element

L_c, E_c = picture-center coordinates (line, element)

e = eccentricity of earth oblate spheriod model

r_{eq} = earth's equatorial radius

r_p = earth's polar radius

s = distance from satellite to landmark

t = time

t_M = epoch time, lies within image frame interval

$\|\cdot\|$ = Euclidean norm

$|\cdot|$ = absolute value operator

I. PRELIMINARIES

A. Coordinate Systems

All coordinate systems used in the model are 3-D right-handed orthogonal coordinate systems. There are five all together. Two have their origins placed at the dynamical center of the earth.

The plane formed by the x- and y-axes of the earth-centered inertial coordinate system (EI) lies in the earth's equatorial plane. The x-axis points at the vernal equinox (γ) which is assumed to be inertially fixed and the z-axis points north. Rotating relative to this inertial frame is the earth-centered rotating coordinate system (ER) with its x-axis passing through the Greenwich meridian and its z-axis coincident with the EI z-axis (Fig. A.1).

In the local vertical (LV) system, the z-axis points to the center of the earth, i.e. the unit vector representing this axis at time t is given as $\hat{z} = -r_s(t)/\|\vec{r}_s(t)\|$, where $\vec{r}_s$ is the satellite radius vector (Fig. A.1). The x-axis is parallel to the earth's equatorial plane and nominally points east.

Fixed in the satellite body is the body-centered (BC) coordinate system whose axes are nominally coincident with the LV system. Departures from this alignment are measured by the yaw, pitch, and roll time dependent angles which in part make up the attitude telemetry data. These rotations are explicitly defined later on. Also nominally coincident with these coordinates systems is the last to be defined, the image or picture frame (PF) coordinate system (Fig. A.2). The z-axis points to the picture center (earth image center) which for ATS-6, is the image point occurring at the midpoint of the mirror sweep angle for the mid-mirror scan number of a full

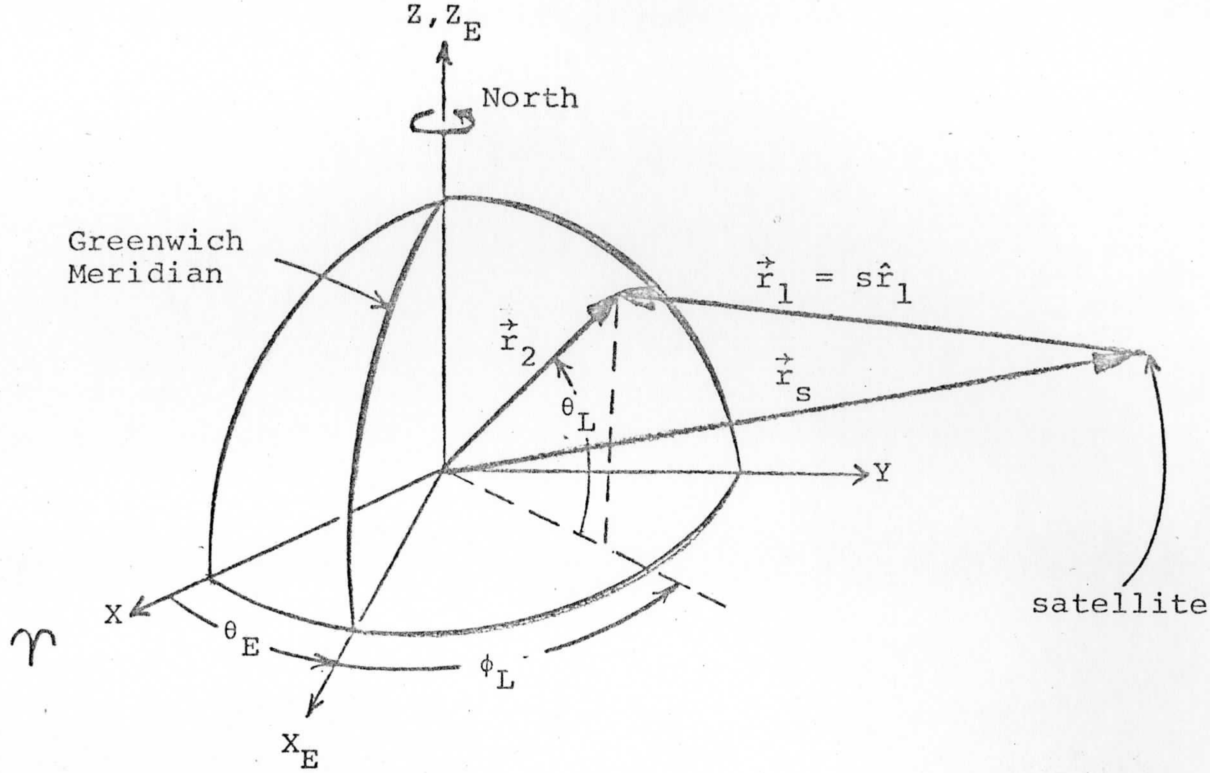


Figure A.1. Satellite-Earth Geometry (See Text for definition of symbols)

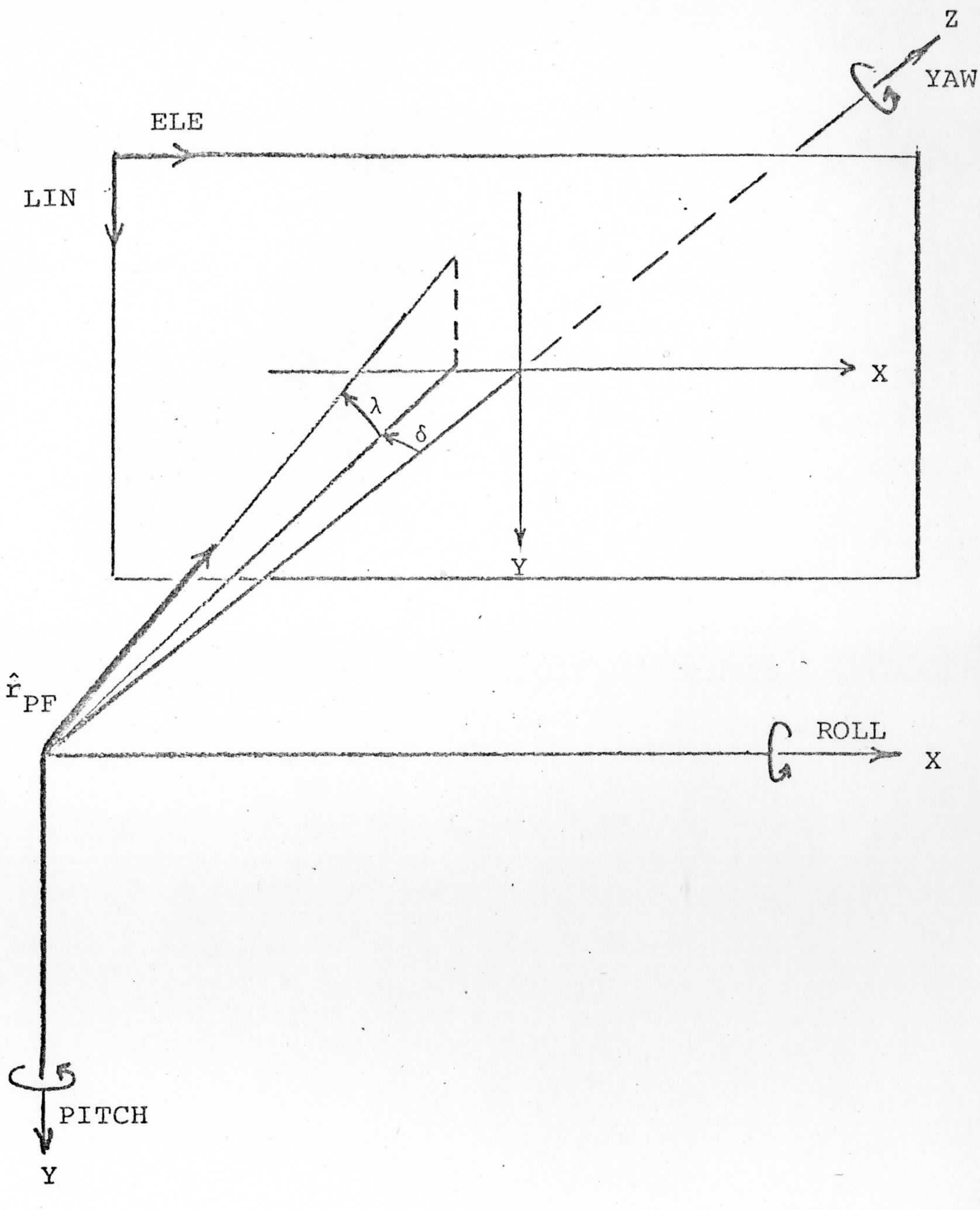


Figure A.2. Picture Frame (PF) Coordinate System
(See Text for explanation of symbols)

image frame scan. (For ATS-6, one complete image scan consists of 1200 mirror scans with 2400 samples or elements per scan line.) The PF x-axis is parallel to the center mirror sweep scan at the picture center and nominally points east.

B. Convention for Orthogonal Matrices

Two basic orthogonal transformation representations in transforming a vector from one coordinate system to another with common origin are given here. Their forms arise naturally in the development of the navigation model depending on convenience or available information.

Given two coordinate systems with a common origin whose axes are unprimed xyz and primed x'y'z' respectively, let R represent the orthogonal transformation of the vector $\vec{r}$, (whose components are expressed in the unprimed system) to the vector $\vec{r}'$ whose components are expressed in terms of the primed system, i.e. $\vec{r}' = R\vec{r}$.

The first form R can take is, in matrix representation, $R = [\hat{x}|\hat{y}|\hat{z}]^T$, where $\hat{x}$, $\hat{y}$, $\hat{z}$ are the unit column vectors of the primed coordinate axes whose components are expressed in the unprimed coordinate system. The "T" refers to the transpose of the matrix. To see that this transformation is valid, one need only to carry out the operation implied,

$$\vec{r}' = [\hat{x}|\hat{y}|\hat{z}]^T\vec{r} = (\hat{x}\cdot\vec{r},\ \hat{y}\cdot\vec{r},\ \hat{z}\cdot\vec{r})^T .$$

Thus, $\hat{x}\cdot\vec{r}$, $\hat{y}\cdot\vec{r}$, $\hat{z}\cdot\vec{r}$ represent the projections of $\vec{r}$ onto the x', y', z' axes respectively. This is precisely the representation of $\vec{r}$ in the primed system.

The second form of R is expressed in terms of rotation angles where in its simplest form $R = R(\theta,k)$ represents a rotation by an angle θ counterclockwise about the k-th axis as viewed from above where k=1,2,3 refers respectively to the x,y,z-axes. Thus, using the conventions defined above, if the z and z' axes were coincident and the x'y'z' system were rotated by an angle θ counterclockwise relative to the xyz system about the z-axis,

$$\vec{r}' = R(\theta,3)\vec{r} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \vec{r}\,. \qquad \text{[ROTATE]*}$$

In general for $R(\theta_K,k)$,

$$R_{ki} = R_{ik} = \begin{Bmatrix} 0 \text{ if } i \neq k \\ 1 \text{ if } i = k \end{Bmatrix} \; ; \; R_{ii} = \cos\theta,\ i\neq k;\ R_{ij} = -R_{ji} = \sin\theta,$$
$$i < j \text{ and } i,j\neq k\ .$$

With this notation, the sequence of Euler rotations can be represented in a compact form. For example, the three rotations about the ATS-6 body axes to define the ATS-6 attitude relative to the LV coordinate system are, in sequence, a ccw rotation by an angle θ_y about the BC z-axis (yaw), followed by a ccw rotation by an angle θ_r about the BC x-axis (roll), followed by a cw rotation by an angle θ_p about the BC y-axis (pitch). Thus a vector expressed in the LV system $\vec{r}_{LV}$, is transformed to the BC system ($\vec{r}_{BC}$) by the operation:

$$\vec{r}_{BC} = R(-\theta_p,2)\ R(\theta_r,1)\ R(\theta_y,3)\ \vec{r}_{LV}\ , \qquad \text{[LVTOBC]}$$

where the angles θ_y, θ_r, θ_p are generally dependent on time.

C. Orbit Methods Used in Predicting Satellite Position, $\vec{r}_s(t)$

The orbit parameters used in the model are derived from the ephemeris data available from the ATS-6 magnetic tapes. These data are in the form of position ($\vec{r}_s$) and velocity ($\dot{\vec{r}}_s$) components expressed in the EI coordinate system approximately every three seconds of ephemeris time. For a two body

---

*Note: Label in brackets (e.g. [ROTATE]) is corresponding FORTRAN subroutine. See Appendix E for listing.

orbit, a position and velocity vector at any time is theoretically sufficient to uniquely determine the orbit. However, the velocity vectors, unlike the position vectors, are not given with sufficient accuracy for the purposes of this model. Therefore, a Gaussian orbit determination method is used in which two position vectors at different times are used to calculate an accurate velocity vector corresponding to one of the selected position vectors. In symbols,

$$\vec{r}_s(t_1), \vec{r}_s(t_2) \rightarrow \vec{r}_s(t_1), \dot{\vec{r}}_s(t_1), \quad t_1 \neq t_2 \quad \text{[GASORB]}$$

The vector spread between $\vec{r}_s(t_1)$ and $\vec{r}_s(t_2)$ must be less than 70°. Once $\vec{r}_s(t_1)$, $\dot{\vec{r}}_s(t_1)$ are determined, the satellite's position for any other time is determined by an iterative f, g computational procedure:

$$\vec{r}_s(t) = f\,\vec{r}_s(t_1) + g\,\dot{\vec{r}}_s(t_1) \quad \text{[ORBIT]}$$

The details of the computational algorithms for these two procedures are given elsewhere.† A thorough testing of these two routines which are incorporated into the ATS-6 model has indicated that any errors generated by them or by uncertainties in the satellite's position are negligibly small compared to errors arising from other sources.

II. THE NAVIGATION MODEL

A. Earth Coordinates to Satellite Image Coordinates (Subroutine ES)

Let t be the instant at which a point on the earth (landmark) is imaged by the satellite scanning system and let $\hat{r}_1(t)$ be the vector which points from the satellite-center to the landmark in question. However, t is not known. Therefore, we make a guess then use the derived line to get a better value and iterate till the solution converges to within a line. The relation

---

†P. R. Escobal, Methods of Orbit Determination, J. Wiley and Sons, New York, 1965. Gaussian orbit algorithm, pp. 196-197, f, g method, p. 423.

of this vector to the satellite radius vector $\vec{r}_s$ and the landmark $\vec{r}_2$ is given by:

$$\vec{r}_1(t) = \vec{r}_2(t) - \vec{r}_s(t) , \qquad \text{[ERTOST]} \qquad (1)$$

where $\vec{r}_2$, $\vec{r}_s$, and therefore $\vec{r}_1$, are expressed in EI coordinates (Figure 1). The position of the satellite $\vec{r}_s(t)$ is determined from the orbit routine discussed above, while $\vec{r}_2$ is derived as follows:

$$\vec{r}_2 = R(-\theta_E, 3)\ \vec{r}_E , \qquad \text{[ERTOST]} \qquad (2)$$

where $\vec{r}_E = \rho(\cos\theta_L \cos\phi_L,\ \cos\theta_L \sin\phi_L,\ (1 - e^2)\sin\theta_L)^T$,

$\rho = r_{eq}/\sqrt{1 - e^2\sin^2\theta_L}$,

θ_L, ϕ_L = the geodetic latitude and longitude of the landmark,

e = eccentricity of the oblate spheriod earth model = 8.1812×10^{-2} ,

r_{eq} = earth's equatorial radius = 6378.15 km.

The landmark $\vec{r}_E$ expressed in ER coordinates is transformed into the EI coordinate system via the transformation $R(-\theta_E, 3)$ where, θ_E, the angle between the x-axes of the two systems at time t is given by

$$\theta_E = a_1 + a_2 * DDD + a_3 * t \qquad \text{[ERTOST]}$$

where, θ_E is the sidereal time of the Greenwich prime meridian,

DDD is the day of the year,

t is universal time,

a_1, a_2, a_3 are constants derived for a specific Julian date; for January 0, 1974

a_1 = 99.59477026 degrees

a_2 = .985647336 degrees/DDD

a_3 = .2506844773 degrees/decimal minute.

It is convenient at this stage to transform the pointing vector $\vec{r}_1$ into a unit vector, $\hat{r}_1(t) = \vec{r}_1(t)/\|\vec{r}_1(t)\|$. Transformation of $\hat{r}_1$ into the LV frame is accomplished by R_{LV},

$$\hat{r}_{LV} = R_{LV}\,\hat{r}_1 = [\hat{x}|\hat{y}|\hat{z}]^T\,\hat{r}_1 \qquad \text{[STTOLV]} \qquad (3)$$

where, from the definition of the LV coordinate system,

$$\hat{z} = -\vec{r}_s(t)/\|\vec{r}_s(t)\| = -(x_s, y_s, z_s)^T,$$

$$\hat{x} = (\hat{z} \times \hat{e}_3)/\|\hat{z} \times \hat{e}_3\| = (-y_s/d,\ x_s/d,\ 0)^T,$$

$$\hat{y} = \hat{z} \times \hat{x} = (x_s z_s/d,\ y_s z_s/d,\ -d^2)^T,$$

$$d = \sqrt{x_s^2 + y_s^2},$$

where $x_s^2 + y_s^2 + z_s^2 = 1$.

In matrix form,

$$R_{LV} = \frac{1}{d} \cdot \begin{bmatrix} -y_s & x_s & 0 \\ x_s z_s & y_s z_s & -d^2 \\ -dx_s & -dy_s & -dz_s \end{bmatrix}.$$

The transformation from LV to BC coordinates is accomplished by using the attitude telemetry data. However, since we have found these data to be unusable for our purposes, the LV and BC coordinates are set equal to each other. Thus:

$$\hat{r}_{BC} = \hat{r}_{LV} \qquad (4)$$

The transformation of $\hat{r}_{BC}$ into the image frame vector, $\hat{r}_{PF}$, is

$$\hat{r}_{PF} = (x_{PF}, y_{PF}, z_{PF})^T = R_{PF}\,\hat{r}_{BC}, \qquad (5a)$$

where $R_{PF} = R(\overline{\theta}_p, 2)\ R(\overline{\theta}_r, 1)\ R(\overline{\theta}_y, 3)$, (5b) and $\bar{\theta}_y$, $\bar{\theta}_r$, $\bar{\theta}_p$ are the optimized angles resulting from the least-squares navigation technique employing landmark measurements. Since R_{PF} is the transformation from the satellite's body-centered frame to its picture (image) frame, it is expected that this transformation will be reasonably time-independent. The effect of thermal and mechanical stresses on the stability of $\bar{\theta}_y$, $\bar{\theta}_r$, $\bar{\theta}_p$ over a period of time on the order of days is something that can only be deduced indirectly by updating the navigation.

With $\hat{r}_{PF}$ determined from equations (5a) and (5b) and adding line and element corrections deduced from earth edges, the satellite image coordinates can be calculated

$$LIN = L_c + (\sin^{-1} y_{PF})/\rho_L + \Delta L_e \qquad [PFTOTC] \qquad (6a)$$

$$ELE = E_c + (\tan(x_{PF}/z_{PF}))/\rho_E + \Delta E_e \qquad \text{and } [EDGCOR] \qquad (6b)$$

where,

LIN = line number,

ELE = element number,

ΔL_e = line correction due to attitude shifts,

ΔE_e = element correction due to attitude shifts,

L_c = picture center line = 1200 for visible, 600 for IR ATS-6 image data,

E_c = picture center element = 1200 for ATS-6 image data,

ρ_L = number of radians/line
ρ_E = number of radians/element } nominal ATS-6 scanned field is 20° × 20°.

Summarizing equations (1) - (5),

$$\hat{r}_{PF} = R_{PF} R_{BC} R_{LV} \hat{r}_1, \qquad [ES] \qquad (7a)$$

where $\hat{r}_1 \cdot \|\vec{r}_1\| = \vec{r}_2(t) - \vec{r}_s(t) = R(-\theta_E, 3)\, \vec{r}_E - \vec{r}_s(t)$. (7b)

Equations (7a) and (7b) effectively define the earth coordinates to satellite image coordinates-transformation.

B. Satellite Image Coordinates to Earth Coordinates (Subroutine SE)

For a given LIN, ELE, it is apparent from equations (6) and Figure 2 that

$$\hat{r}_{PF} = (-\cos\lambda\sin\delta, -\sin\lambda, \cos\lambda\cos\delta)^T \quad \text{[PFTOTC]} \quad (8a)$$

where

$$\lambda = (L_c - LIN + \Delta L_e)\ \rho_L = \text{mirror step angle} \quad (8b)$$

$$\delta = (E_c - ELE + \Delta E_e)\ \rho_E = \text{mirror sweep angle} \quad (8c)$$

From equation (7a),

$$\hat{r}_1 = R_{LV}^T R_{BC}^T R_{PF}^T \hat{r}_{PF} \ . \quad \text{[SE]} \quad (9)$$

(The three successive transformations in equation (9) are orthogonal matrices; therefore the inverse of each is equal to its transpose).

Now $$\vec{r} = \vec{r}_s + s\hat{r}_1 , \quad (10)$$

where $s = \|\vec{r}_1(t)\|$ equals the distance from the satellite-center to the landmark. The solution of s is achieved by using eq. (10) and the equation of the earth spheroid,

$$(x_E^2 + y_E^2)/r_{eq}^2 + z_E^2/r_p^2 = 1 \quad (11)$$

where,

r_{eq} = earth's equatorial radius = 6378.15 km,

r_p = earth's polar radius = 6356.77 km,

x_E, y_E, z_E are the vector components of the landmark, $\vec{r}_E$, in the ER frame.

Equations (2), (10) and (11) represent a system of equations of four unknowns (x_E, y_E, z_E, s). The solution of s can easily be accomplished as follows:

Divide the x and y components by r_{eq} and the z component by r_p in (10).

This results in the equation,

$$\vec{r}_2{}^* = \vec{r}_s{}^* + s\vec{r}_1{}^* \tag{12}$$

where,

$$\vec{r}_2{}^* = R(-\theta_E,3)\ \vec{r}_E{}^* = R(-\theta_E,3)(x_E/r_{eq},y_E/r_{eq},z_E/r_p)^T,$$

$$\vec{r}_1{}^* = (x_1/r_{eq},y_1/r_{eq},z_1/r_p)^T,$$

$$\vec{r}_s{}^* = (x_s/r_{eq},y_s/r_{eq},z_s/r_p)^T\ .$$

Now $\|\vec{r}_2{}^*\|^2 = \|R(-\theta_E,3)\|^2 \cdot \|\vec{r}_E{}^*\| = 1$, since $R(\theta_E,3)$ is an orthogonal matrix, and $\|\vec{r}_E\|^2$ equals the left side of equation (11). Therefore the equation

$$\|\vec{r}_2{}^*\| = 1 = \|\vec{r}_s{}^* + s\vec{r}_1{}^*\|\ , \tag{13}$$

contains only s as an unknown. The solution of (13), expressed in a form to minimize computational round-off errors, is:

$$s = -(B + \sqrt{RAD})/2A, \tag{14}$$

where

$$\left.\begin{aligned} RAD &= B^2 - 4AC, \\ A &= F + (1 - F)\ z_1{}^2, \\ B &= = 2(x_1x_s + y_1y_s)\ F + 2z_1z_s, \\ C &= (x_s^2 + y_s^2)\ F + z_s^2 - r_p^2, \\ F &= r_p^2/r_{eq}^2\ . \end{aligned}\right\}\ \text{[ERTOST]}$$

With the solution of s, it follows from (1) and (10) that

$$\vec{r}_E = R(\theta_E,3)(\vec{r}_s + s\hat{r}_1) = (x_E,y_E,z_E)^T,\quad \text{[ERTOST]} \tag{15}$$

and hence

$$\theta_L = \tan^{-1}[z_E/((1 - e^2)\sqrt{x_E^2 + y_E^2})],\quad \text{[ERTOER]} \tag{16a}$$

$$\phi_L = \tan^{-1}[y_E/x_E], \qquad \text{[ERTOER]} \qquad (16b)$$

where

θ_L = geodetic latitude,

ϕ_L = longitude

$e^2 = (r_{eq}^2 - r_p^2)/r_{eq}^2$ = earth's eccentricity squared.

Equations (9), (15), and (16) constitute the basic - satellite image coordinates to earth coordinates - transformation.

C. Navigation Optimization Procedure (Subroutine ATTTUD]

Navigation of the satellite image data base consists of finding a time dependent transformation to predict the earth coordinates from the associated satellite image coordinates. This is accomplished by using landmark measurements from the data base and the model discussed above to determine the optimal transformation in a least-squares sense. A landmark measurement consists of the earth coordinates, (θ_L,ϕ_L), the associated image coordinates, (LIN,ELE), and the time, t, at which the landmark was imaged.

Let

$\alpha_i = i^{th}$ parameter to be optimized,

$R_A(\alpha_i)$ = transformation associated with the α_i,

$\hat{r}_k'$ = unit pointing vector of the k^{th} landmark derived from $(\theta_L,\phi_L)_k$,

$\hat{r}_k$ = unit pointing vector of the k^{th} landmark derived from $(LIN,ELE)_k$,

and

$$S(\alpha_i) = \sum_k \|\hat{r}_k - R_A(\alpha_i)\, \hat{r}_k'\|^2 \qquad \text{[S]} \qquad (17)$$

which is a sum over all landmark measurements included in the optimization. The navigation is complete when a set of parameters α_i are found (call them $\overline{\alpha}_i$) which minimizes S. The optimized values $\overline{\alpha}_i$ are then used in the transformation to predict (θ_L, ϕ_L) from an arbitrary (LIN,ELE).

As an example, consider equation (5a),

where $$\hat{r}_{PF} = R_{PF}\, \hat{r}_{BC} \qquad \text{[BCTOPF]}$$

and let us assume that the attitude telemetry data is known with a reasonable degree of accuracy but that the orientation of the PF frame relative to the BC frame is not. Thus, we wish to optimize θ_y, θ_r, θ_p in the transformation R_{PF}. Using the landmark measurements and the attitude telemetry data, calculate $\hat{r}_{BC}$ as given by equations (1) to (4a) and $\hat{r}_{PF}$ from equations (8) for each landmark; therefore S in this case takes the form

$$S(\theta_y, \theta_r, \theta_p) = \sum_k \left\| (\hat{r}_{PF})_k - R_{PF}(\theta_y, \theta_r, \theta_p)(\hat{r}_{BC})_k \right\|^2, \qquad \text{[S]}$$

and the values $\overline{\theta}_y$, $\overline{\theta}_r$, $\overline{\theta}_p$ which minimize S are the ones then used in equation (5b).

The method used to minimize $S(\alpha_i)$ is an iterative procedure which uses a modified Newton's method.

Let

α = $\{\alpha_i\}$ for convenience,

α^n = value of α resulting from the n^{th} iteration,

N = total number of parameters α_i,

H = $N \times N$ matrix (Hessian) whose ij^{th} component is $[H]_{ij} = \partial^2 S/\partial\alpha_i \partial\alpha_j$,

∇S = gradient of S, $(\nabla S)_i = \partial S/\partial\alpha_i$.

The iterative procedure works as follows:

1) Start with m = 0 and increase m by 1 until a value M is found such that

$$S(\alpha^n) - S(\alpha^n - 2^{-M} H^{-1}\nabla S(\alpha^n)) \geq (.4)2^{-M} H^{-1}\nabla S(\alpha^n) \cdot \nabla S(\alpha^n), \qquad \text{[LS]}$$

2) When the inequality is satisfied, set

$$\alpha^{n+1} = \alpha^n - 2^{-M} H^{-1}\nabla S(\alpha^n), \qquad \text{[MINMIZ]}$$

3) Check to see if the following convergence criteria is met

$$|S(\alpha^{n+1}) - S(\alpha^n)| \leq 10^{-18}|S(\alpha^n)| .$$

If not, set $\alpha^n \leftarrow \alpha^{n+1}$ and go to 1); if yes, then $\alpha^{n+1} = \bar{\alpha}$ and the procedure is finished.

APPENDIX B. ANALYSIS OF METHOD WHICH DETERMINES ATS-6 SSP IMAGE COORDINATE DISPLACEMENTS BETWEEN SUCCESSIVE IMAGES RESULTING FROM ATTITUDE CHANGES

This appendix presents an analysis of the technique used to calculate the ATS-6 Subsatellite Point (SSP) image coordinate changes between successive images resulting from attitude changes during the image-scan time. Appendix C shows how these measurements are used to account for the attitude changes in the process of computing accurate cloud displacements.

1. Method

Let T_1, T_2 designate two successive ATS-6 data images where T_1 is the "earlier" of the two images and (L_c, E_c) (Line scan and Element numbers) are its SSP image coordinates (Fig. B.1) determined from the ATS-6 navigation model. Let (L,E) be the T_1 image coordinates for a point on the right earth edge and ΔE_R, ΔE_L the measured displacements along line L of the Right and Left T_2- earth edges relative to the T_1 earth edges. (These measurements are obtained in practice on McIDAS using the infrared ATS-6 data images and an image-matching technique which is constrained to measure displacements of the earth edge only along a scan line. It is worth emphasizing that the image-matching method indeed measures displacements of the geometrical earth edge and not features near it - such as clouds. This is not surprising since the greatest contrast is between earth and space - not within features near the earth's edge.)

The object, then, is to compute the displacement coordinates $(\Delta L, \Delta E)$ of the T_2-SSP relative to the T_1- SSP at line L using the measured values of $\Delta E_r, \Delta E_L$.

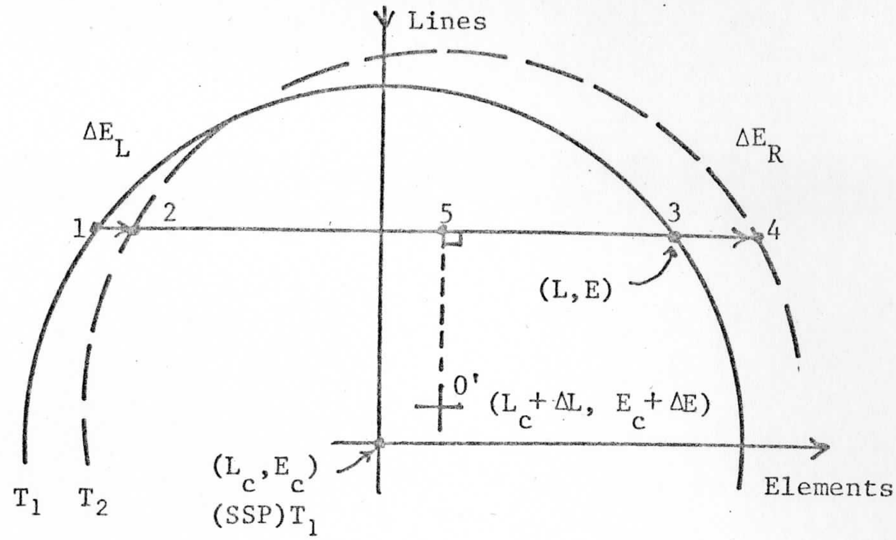


FIGURE B.1. Earth-edge Displacement Measurement Geometry

Let points 1-5 be on line L containing the earth edge points indicated in the figure, and points 0 and 0' the T_1-SSP and T_2-SSP. Furthermore, point 5 is the bisect point for the T_2-chord coinciding with line L, and both T_1- and T_2-earth circles have equal radii a.

Letting $\overline{P_i - P_j}$ be the distance (always non-negative) between points i and j, $\Delta L, \Delta E$ can be derived from two simple geometrical identities:

The first is

$$\overline{P_4 - P_5} = \overline{P_5 - P_2} \text{ , or}$$

$$(E + \Delta E_R) - (E_c + \Delta E) = (E_c + \Delta E) - (2E_c - E + \Delta E_L) \text{ ,}$$

solving for ΔE,

$$\Delta E = (\Delta E_R + \Delta E_L)/2 \text{ .} \tag{1}$$

The second identity is

$$\overline{P_4 - P_2} - \Delta E_R = \overline{P_3 - P_1} - \Delta E_L \text{ , where} \tag{2}$$

$$\overline{P_4 - P_2} = 2\sqrt{(P_4 - P_{0'})^2 - (P_5 - P_{0'})^2}$$

$$= 2\sqrt{(P_3 - P_0)^2 - (P_5 - P_{0'})^2}$$

$$= 2\sqrt{(L - L_c)^2 + (E - E_c)^2 - [L - (L_c + L)]^2} \text{ , and} \tag{3}$$

$$\overline{P_3 - P_1} = 2(E - E_c) \; . \tag{4}$$

Substitution of (3) and (4) into (2) and rearranging yields

$$\delta = \sqrt{X^2 + Y^2 - (Y - \Delta L)^2} - X \text{ , where} \tag{5}$$

$$\delta = (\Delta E_R - \Delta E_L)/2$$

$$X = E - E_c$$

$$Y = L - L_c \; .$$

Since δ and (L,E) are obtained from the $\Delta E_R, \Delta E_L$ measurements and a nominal value used for (L_c, E_c), ΔL can be solved in (5) to yield

$$\Delta L = Y \pm \sqrt{RAD} \text{ , where} \tag{6}$$

$$RAD = Y^2 - 2 X \delta - \delta^2 \; .$$

The choice of sign in (6) is determined by substituting the expression for δ in Eq. (5) in RAD; the result is

$$\begin{aligned} \Delta L = Y \pm \sqrt{(Y - \Delta L)^2} &= Y + \sqrt{RAD} \text{ , } Y > \Delta L \\ &= Y - \sqrt{RAD} \text{ , } Y < \Delta L \; . \end{aligned} \tag{7}$$

The peculiar condition implied by (7) that ΔL must be known before ΔL can be calculated is really not a problem since generally ΔL is small ($\pm$ 5 lines) and $|Y|$ is usually >> $|\Delta L|$. For most cases then, the conditions are

$$\Delta L = Y \pm \sqrt{RAD} \text{ , } Y \gtrless 0 \text{ ,} \tag{8}$$

$$RAD = Y^2 - 2 X \delta - \delta^2 \; .$$

For the case where $|Y|$ approachs (ΔL) in value there are other problems; these are discussed at the end of this appendix.

Equations (1) and (8), then, define the displacement in image coordinates of the T_2-SSP relative to the T_1-SSP for line L. By repeating this entire process for other scan lines the SSP-shifts $(\Delta L, \Delta E)$ as function of T_1-line position are obtained. In practice the right and left earth-edge displacements are first measured over the entire range of scan lines of interest. Curves are then separately fit to the right and left edges measurements resulting in a ΔE_R vs L and a ΔE_L vs L curve over the scan-line range of interest. The curve values themselves are then used in (1) and (8) to compute $\Delta L, \Delta E$ curves.

APPENDIX C. ANALYSIS OF ALGORITHM WHICH ACCOUNTS FOR RELATIVE ATTITUDE CHANGES IN SUCCESSIVE ATS-6 DATA IMAGES USING EARTH-EDGE SHIFT MEASUREMENTS

This appendix provides a semi-rigorous analysis of the algorithm used to account for the relative attitude changes in a sequence of ATS-6 data images using the earth-edge measurements derived by the technique discussed in the main portion of the report and APPENDIX B. In effect, this appendix shows why the algorithm "works"; the following appendices discuss in more detail the limitations of and the errors associated with this method.

Consider a sequence of n ATS-6 images designated by T_1, T_2, ... T_n, from which the displacements in earth coordinates of a feature (cloud) between successive images are to be determined. Let

(L_i, E_i) = image coordinates (Line, Element) of the cloud feature for the T_i^{th} image,

$\hat{r}_{PFi}$ = $\hat{r}_{PF}(L_i, E_i)$ = unit pointing vector in Picture Frame coordinates derived from (L_i, E_i) using the ATS-6 scan-camera geometry,

$\hat{r}_{LVi}$ = unit Local Vertical pointing vector associated with $\hat{r}_{PFi}$,

R_i = 3x3 rotation matrix which transforms <u>without error</u> r_{LVi} into $\hat{r}_{PFi}$ at the time the feature was scanned, i.e.

$$\hat{r}_{PFi} = R_i \, \hat{r}_{LVi} \ . \tag{1}$$

1. A Two-Image Sequence

Consider now the first two images T_1, T_2 in the sequence whose relationship between LV and PF coordinates are given by

$$\hat{r}_{PF1} = R_1 \, \hat{r}_{LV1} \tag{2}$$

$$\hat{r}_{PF2} = R_2 \, \hat{r}_{LV2} \ . \tag{3}$$

The relationship of the cloud feature in LV coordinates for image T_2 can be related to T_1 as follows:

$$\hat{r}_{LV2} = \hat{r}_{LV1} + \Delta\vec{r}_{LV}(\Delta T_{12}) \ , \tag{4}$$

where $\Delta\vec{r}_{LV}$ is the displacement of the cloud in LV coordinates from T_1 to T_2 and results from two effects:

(i) apparent change in the earth's orientation over the time interval ΔT_{12} caused by a non-zero angle of inclination in the satellite's orbit (the eccentricity of the ATS-6 orbit was so small ($\sim 10^{-4}$) that no significant change in angular size occurs)

(ii) motion of the cloud relative to the earth's surface.

Now the orientation of the earth relative to the LV frame depends only on the orbit and the earth's position relative to celestial coordinates; thus if $\hat{r}_{LV1}$ and $\hat{r}_{LV2}$ were accurately known, the displacement of the cloud over the time interval associated with these two vectors could be calculated with an accuracy limited only by the equations describing the dynamical relationship between the satellite orbit and the earth's position relative to celestial coordinates. It is assumed that for ATS-6 that this transformation produces negligible error.

Returning to equations (2) and (3), the rotation matrices R_1, R_2 differ from each other slightly because of an attitude change of the PF frame relative to the LV frame between T_1 and T_2; they are related to each other by an infinitesimal rotation $I + E_{12}$ where I is the unit matrix and E_{12}, an antisymmetic matrix whose elements are the small-angle differences between the PF axes at times T_1 and T_2. Thus

$$R_2 = (I + E_{12})\, R_1 \; ; \tag{5}$$

substituting (4) and (5) in (3) and expanding

$$\hat{r}_{PF2} = R_1 \hat{r}_{LV1} + E_{12} R_1 \hat{r}_{LV1} + R_1 \Delta\vec{r}_{LV}(\Delta T_{12}) + E_{12} R_1 \Delta\vec{r}_{LV}(\Delta T_{12}) \tag{6}$$

The first term on the RHS of (6) is $\hat{r}_{PF1}$, the last term is negligible ($\sim 10^4$ times smaller than the first order terms), the second term is the change in PF coordinates due to an attitude change and the third term is the change in PF coordinates due to the two changes in the LV frame discussed above. Using (4) to combine the first and third terms, equation (6) then can be rewritten as

$$\hat{r}_{PF2} = R_1 \hat{r}_{LV2} + E_{12} \hat{r}_{PF1} \; . \tag{7}$$

Comparison of (7) with (2) implies that if E_{12} and R_1 were known, $\hat{r}_{LV1}$ and $\hat{r}_{LV2}$ – and therefore the cloud displacement – could be computed with a high degree of accuracy.

A good approximation to E_{12} is obtained from the earth-edge displacement measurements which provide the displacement in image coordinates $\Delta L_{12}(L)$, $\Delta E_{12}(L)$ of T_2 relative to T_1 as a function of line number L. The line shift ΔL_{12} is proportional to a small change in roll ΔR_{12} about the PF-X axis and the element shift is proportional to small changes in pitch ΔP_{12} about the PF-Y axis, i.e.

$$\Delta L_{12} = \Delta R_{12}/\rho \; , \tag{8a}$$

$$\Delta E_{12} = \Delta P_{12}/\rho \; , \tag{8b}$$

where ρ is the angular size of a pixel = 1.45×10^{-4} radians.

Letting $\hat{r}_{PF1} = [X, Y, Z]^T$ where T is the transpose, the explicit form of E_{12} and the last term in (7) is

$$E_{12}\hat{r}_{PF1} = \begin{bmatrix} 0 & 0 & \Delta P \\ 0 & 0 & \Delta R \\ -\Delta P & -\Delta R & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

$$= [\Delta PZ,\ \Delta RZ,\ -\ (\Delta PX + \Delta RY)]^T, \qquad (9)$$

where it is understood that ΔR, ΔP refer to the T_1 to T_2 changes in attitude at the times the cloud was scanned.

Note E_{12} contains no non-zero yaw angle element (small rotation about the axis passing through the image SSP) since there is no way that the earth-edge displacement measurement technique can provide this angle; however, the analysis given in Appendix E of reference 3 shows that the effect on the accuracy of wind measurements (displacements) for yaw changes is generally small. Analysis of landmark measurements and the ATS-6 wind sets shown in this report, indicate that for the data images studied thus far, the yaw changes are negligible.

With E_{12} and hence the last term in (7) determined, equation (7) can be rewritten as

$$\hat{r}'_{PF2} = \hat{r}_{PF2} - E_{12}\hat{r}_{PF1} = R_1\hat{r}_{LV2}\ , \qquad (10)$$

where $\hat{r}'_{PF2}$ is interpreted as the T_2 cloud pointing vector with the attitude changes removed.

2. Computing $\hat{r}'_{PF2}$ in Practice

Cloud displacement measurements and computation of $\hat{r}'_{PF2}$ is accomplished as follows:

(1) the T_1 image coordinates (L_1, E_1) of the cloud are recorded,

(2) the total displacement of the cloud in image coordinates $(\Delta L^T, \Delta E^T)$ from T_1 to T_2 are measured on McIDAS using an image matching technique,

(3) from previously determined earth-edge displacement parameters, the displacement coordinates ΔL_{12}, ΔE_{12} due to relative attitude changes are computed and subtracted from ΔL^T, ΔE^T. These differences are added to L_1, E_1 and the associated PF vector computed. That this vector is a good approximation of $\hat{r}'_{PF2}$ is shown below.

By definition,

$$\hat{r}_{PF1} = \hat{r}_{PF}(L_1, E_1) \quad (11)$$

$$\hat{r}_{PF2} = \hat{r}_{PF}(L_1 + \Delta L^T, E_1 + \Delta E^T) = \hat{r}_{PF}(L_2, E_2) \quad (12)$$

Expanding $\hat{r}_{PF}(L_1 + \Delta L^T - \Delta L_{12}(L_2), E_1 + \Delta E^T - \Delta E_{12}(L_2))$ about $(L_1 + \Delta L^T, E_1 + \Delta E^T)$ keeping first order terms, we have

$$\hat{r}_{PF}(L_1 + \Delta L^T, E_1 + \Delta E^T) - \frac{\partial}{\partial L} \hat{r}_{PF}(L_2, E_2)\, \Delta L_{12} - \frac{\partial}{\partial E} \hat{r}_{PF}(L_2, E_2)\, \Delta E_{12} \, . \quad (13)$$

From the geometry of the ATS-6 scan-camera,

$$\begin{aligned} \hat{r}_{PF} &= [-\cos\lambda\sin\delta,\ -\sin\lambda,\ \cos\lambda\cos\delta]^T \quad (14) \\ &= [X,\ Y,\ Z]^T \end{aligned}$$

where

$\lambda = (L_c - L)\rho$ = mirror step angle,

$\delta = (E_c - E)\rho$ = mirror sweep angle,

(L_c, E_c) = image center line and element value,

ρ = angular size of a line or element.

Taking the partials of $\hat{r}_{PF}$ in (14) with respect to L,E, and retaining first order terms with the partials evaluated at (L_2,E_2):

$$\frac{\partial}{\partial L}\hat{r}_{PF}\,\Delta L_{12} + \frac{\partial}{\partial E}\hat{r}_{PF}\,\Delta E_{12}$$

$$= [0,\ PZ,\ -PY]^T \Delta L_{12} + [PZ,\ 0,\ -PX]^T \Delta E_{12}\ , \qquad (15)$$

substituting (8a) and (8b) into the above expression and adding the two vectors in (15) yields

$$[Z\Delta P_{12},\ Z\Delta R_{12},\ -(X\Delta P_{12} + Y\Delta R_{12})]^T\ ,$$

which is equivalent to (9); therefore (15) is the same as $E_{12}\hat{r}_{PF1}$. Since the first term in (13) is $\hat{r}_{PF_2}$, we have

$$\hat{r}_{PF}(L_1 + \Delta L^T - \Delta L_{12}(L_2),\ E_1 + \Delta E^T - \Delta E_{12}(L_2)) \doteq$$
$$\hat{r}_{PF2} - E_{12}\hat{r}_{PF} = \hat{r}'_{PF2} \quad \text{by} \qquad (16)$$

comparison with (10). Thus the method discussed above correctly yields $\hat{r}'_{PF2}$.

3. <u>Transforming PF to LV coordinates</u>

With the attitude changes between T_1 and T_2 removed, we have

$$\hat{r}_{PF1} = R_1\hat{r}_{LV1} \qquad (17)$$

$$\hat{r}'_{PF_2} = R_1\hat{r}_{LV2} \qquad \text{(see (10))} \qquad (18)$$

Let $\overline{R}$ be the transformation from LV to PF coordinates derived from T_1 landmark measurements in a least squares sense assuming a constant attitude over the time interval the T_1 landmarks were scanned, i.e. $\overline{R}$ is a constant matrix. $\overline{R}$ then will differ slightly from R_1 due to small attitude changes over the T_1 scan time and, in addition, will differ from R_1 (assumed to be the error-free transformation) due to mirror scan nonlinearities. Similarly, as above, we can relate $\overline{R}$ to R_1 by an infinitesimal transformation

$$\overline{R} = R_1 \, (I - e) \;, \tag{19}$$

where e is the infinitesimal antisymmetric error matrix which is a function of line and element. It is important to realize that the elements of e contain absolute errors which the ATS-6 model cannot account for. However, our previous work using landmark measurements has provided us with bounds on these errors and the rate at which they change as a function of image coordinates. The worst-case estimates correspond to about $\pm$ 2.5 pixels ($\pm$ 3.6 $\times 10^{-4}$ radians). The errors tend to be oscillatory as a function of line or element position with a period of 100 to 200 pixels. Thus for typical feature displacement measurements (corresponding to tens of pixels), e can be considered to be a constant locally. (It would be appropriate to remind the reader at this point that the absolute mirror scan nonlinearities as a function of image coordinates repeat from image to image.)

Taking the transpose of (19) and operating on (17) and (18)

$$\hat{r}'_{LV1} = \overline{R}^T \hat{r}_{PF1} = (I + e) R_1^T R_1 \hat{r}_{LV1} = \hat{r}_{LV1} + \vec{r}_e \tag{20a}$$

$$\hat{r}'_{LV2} = \overline{R}^T \hat{r}'_{PF2} = (I + e) R_1^T R_1 \hat{r}_{LV2} = \hat{r}_{LV2} + \vec{r}_e \;, \tag{20b}$$

where

$$\vec{r}_e = e \hat{r}_{LV} \;. \tag{21}$$

Application of $\overline{R}$ then, results in the correct LV vectors to within the same additive vector constant. It is shown in Appendix C of reference 3 that this constant has a negligible effect in computing cloud velocities or displacements.

In addition location errors on the earth due to $\vec{r}_e$ is given in this appendix.

4. General Case of n-image Sequence

The case for images T_1, T_2, ... T_n is simply a generalization of the 2-image case. The error-free LV to PF frame matrix for T_i can be related to R_1 by the image pair relative-attitude transformation matrices $E_{i,i+1}$ where

$$R_n = (I + E_{n-1,n}) \cdots (I + E_{23})(I + E_{12})R_1$$

$$= \left(I + \sum_{i=1}^{n-1} E_{i,i+1}\right) R_1 \;, \qquad (22)$$

where only first order terms are retained. Thus the analog to (10), generalized to n images, is

$$\hat{r}'_{PFn} = \hat{r}_{PF} - \left(\sum_{i=1}^{n-1} E_{i,i+1}\right)\hat{r}_{PF1} = R_1 \hat{r}_{LVn} \;,$$

where in practice $\hat{r}'_{PFn}$ is computed by evaluating the expression

$$\hat{r}'_{PFn} = \hat{r}_{PF}\left(L_n - \sum_{i=1}^{n-1} \Delta L_{i,i+1}(L_{i+1}),\; E_n - \sum_{i=1}^{n-1} \Delta E_{i,i+1}(L_{i+1})\right) \qquad (23)$$

where $L_{i,i+1}(L_{i+1})$, $E_{i,i+1}(L_{i+1})$ are evaluated from the T_i to T_{i+1} earth-edge displacement measurements evaluated at L_{i+1}, where L_{i+1} is the T_{i+1} line number of the "center of gravity" of the feature being tracked.

Application of (19) to the $\hat{r}'_{PFi}$ results in

$$\hat{r}'_{LVi} = \hat{r}_{LVi} + \vec{r}_e \qquad i = 1, 2, \ldots n \;. \qquad (24)$$

APPENDIX D. METHOD OF OBTAINING MIRROR-SCAN-OFFSET CORRECTION CURVES

The nature of the mirror-scan nonlinearity problem has been discussed in previous reports. In order to correct the alternate scan offset caused by this effect, we need a table of offset values as a function of element number (i.e. a $\Delta E(E)$ function). For our initial efforts to produce a $\Delta E(E)$ function, we used the McIDAS cloud-tracking program to measure the displacement of a feature seen in the odd number scans of an image to its position in the even numbered scans of the same image. The method described here is a somewhat more automated scheme and does not require viewing the ATS-6 images. The method has been applied to an IR image (74195 173134Z) and gives good agreement with the old method with less scatter of individual points about a polynomial least-squares fit curve (Fig. D.1.).

In the automated method of computing the $\Delta E(E)$ function a correlation value is computed for the match between a small segment of an odd scan and a shifted segment of an adjacent even scan. The amount of shift which gives the best correlation for that small line segment is taken to be the ΔE value for that scan line and element location. Values are computed over many scans and elements, then averaged over the scans. The result is a table of ΔE values as a function of E(element). The table values are then smoothed by using a least squares fit polynomial. In more detail the method is as follows:

Let:

e = element number ($1 \leq e \leq 2400$)

s = scan number ($1 \leq s \leq 1200$)

$p_e(s,e)$ = pixel digital value at a given even-scan, element position

$p_o(s,e)$ = pixel digital value at a given odd-scan, element position

δ = an element shift value

t = a small number defining the length of the scan segment used for a correlation

$C(s,e,\delta)$ = a measure of the match between line segments from adjacent odd and even scans.

Now define C by:

$$C(s,e,\delta) = \sum_{e'=e-t}^{e+t} [p_o(s,e') - p_e(s+1, e'+\delta)]^2 \qquad \text{(s odd)}$$

The element shift, ΔE, for a given point on the image (s,e) is the value of δ that gives a minimum value to C:

$$C(s,e,\Delta E(s,e)) \leq C(s,e,\delta) \qquad \text{for all values of } \delta$$

After an array of these values is generated, a weighted average over scans is taken:

$$\Delta E(e) = \frac{\sum_s w(s,e)\Delta E(s,e)}{\sum_s w(s,e)}$$

The weighting factor is the range of brightness values from the odd scan used in computing C.

$$w(s,e) = MAX(p_o(s,e')) - MIN(p_o(s,e'))$$

where $e - t \leq e' \leq e + t$.

The reason for using this weighting is simply that high contrast features should, in general, produce a correlation value of more significance than features of nearly uniform brightness.

The computer program to do this used the values: $t = 7$ and $+4 \leq \delta \leq 12$. Values of $\Delta E(s,e)$ were computed for every eight scans for the middle third of the image ($s = 399, 407, \ldots, 799$) and for every 10 elements across most of the width of the image ($e = 100, 110, \ldots, 2300$). Using this line and element range some ΔE values will be computed for points off the earth. When off-earth and on-earth points are averaged together the weighting factor will give only a very small or zero weighting to the off-earth points. For some element values no on-earth points are encountered. We found that these points could be determined from the table of $\Delta E(e)$ values. There is a discontinuity between the on-earth and off-earth values.

The $\Delta E(e)$ function is then used to create a corrected image by shifting even scans. An improved method of correcting the alternate scan offset has also been developed. In the old method a fixed shift value was used across the width of a McIDAS image (672 pixels). Although this gave fairly good results, there could be a one or two element alignment error near the edges of the image. The new method, as before, uses the odd scans as a fixed reference. However, the required amount of shift is computed, or looked up in a table, for each pixel in the even scans. Thus the amount of shift varies from one side of the image to the other. Pixels are dropped or doubled, as appropriate, between regions of different shift values.

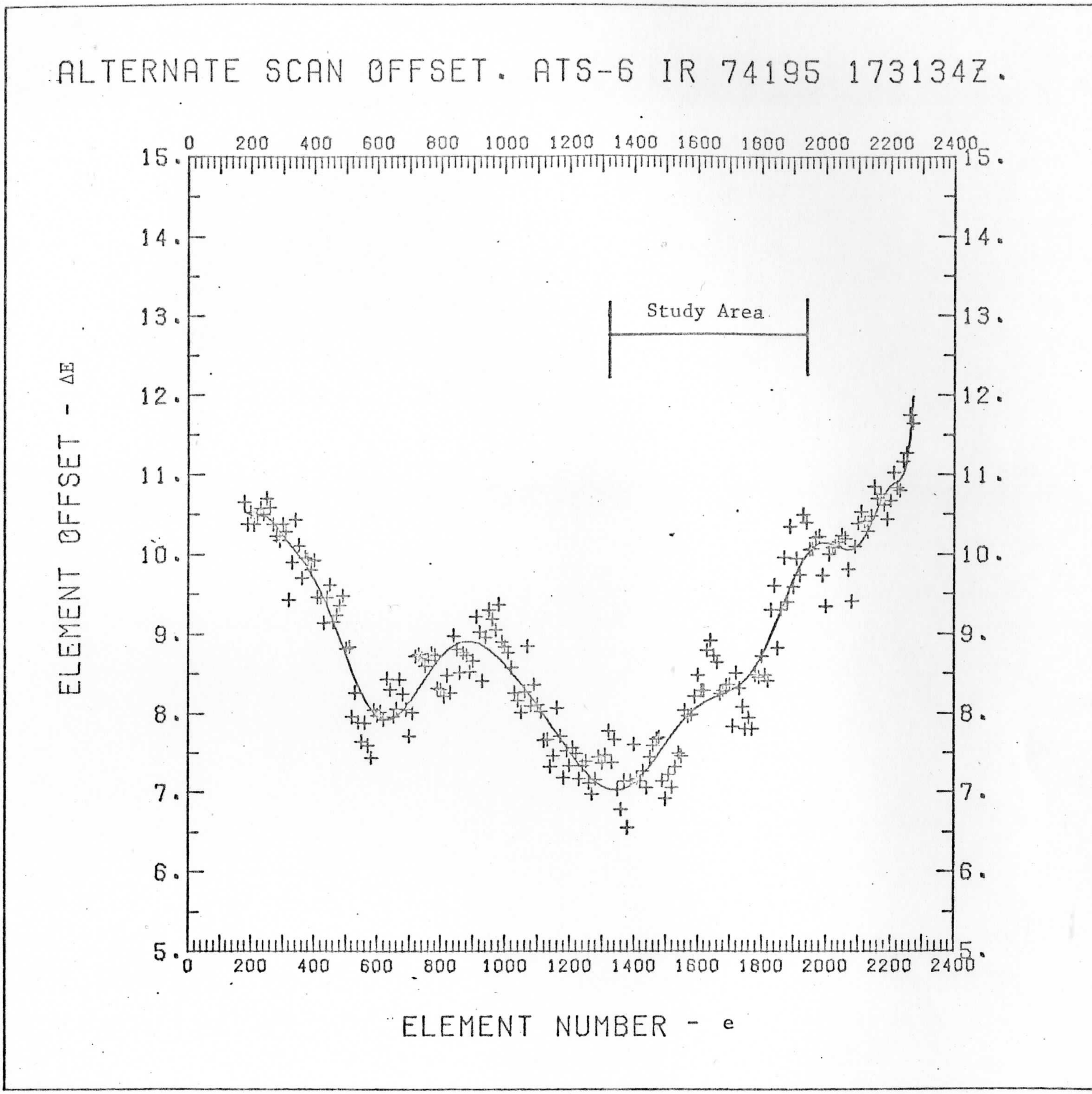


FIGURE D.1. Mirror-Scan Nonlinearity Curve ΔE(e) using newly developed procedure.

Appendix E

Source Code Listings (FORTRAN) of Programs and Subroutines

This appendix contains FORTRAN source listings for the main programs and subroutines for the ATS-6 image correction and navigation system. For main programs both AOIPS and McIDAS versions are given. For subroutines only the McIDAS version is given as the AOIPS version would be identical.

ATSF2/AOIPS

```
      DIMENSION MIN(10)
711   FORMAT(/////'        ***   ATS 6   PROCESSING    ***')
712   FORMAT( 74H       1    INITIALIZE NAVCOM
     *                                                               )
713   FORMAT( 74H       2    GENERATE ELEMENT OFFSET DATA FROM E.H.T.
     *                                                               )
714   FORMAT( 74H       3    CURVE FIT TO OFFSET DATA
     *                                                               )
715   FORMAT( 74H       4    READ IMAGE SEGMENT FROM E.H.T.
     *                                                               )
716   FORMAT( 74H       5    CURVE FIT TO EARTH EDGE DATA
     *                                                               )
717   FORMAT( 74H       6    RUN NAVIGATION
     *                                                               )
718   FORMAT( 74H       7    EXIT ATS6 PROCESSING
     *                                                               )
      IT=5
      WRITE(IT,711)
      WRITE(IT,712)
      WRITE(IT,713)
      WRITE(IT,714)
      WRITE(IT,715)
      WRITE(IT,716)
      WRITE(IT,717)
      WRITE(IT,718)
      NL=8
      MENU=1
      CALL INCOM(' ',0,NL,1,MIN,MENU,1,1)
      IF(MENU.LE.0) GO TO 990
      IJ=MIN(1)
      GO TO (1,2,3,4,5,6,7),IJ
1     CALL REQUES(RAD50('A6INT2'))
      GO TO 999
2     CALL REQUES(RAD50('OFSTG2'))
      GO TO 999
3     CALL REQUES(RAD50('OFSTF2'))
      GO TO 999
4     CALL REQUES(RAD50('LATSF2'))
      GO TO 999
5     CALL REQUES(RAD50('EDGFT2'))
      GO TO 999
6     CALL REQUES(RAD50('ATSNV2'))
      GO TO 999
7     CALL REQUES (RAD50('MET2'))
      GO TO 999
990   CONTINUE
      CALL REQUES(RAD50('MET2'))
999   CONTINUE
      END
```

ATSNV/AOIPS

```
C     PROGRAM TO NAVIGATE ATS6 IMAGES FROM LANDMARKS.
      INTEGER*4 ILAT,ILON,IYD,ITIME
      DIMENSION PTIME(150),ALIN(150),AELE(150),ALAT(150),ALON(150)
      DIMENSION ITIME(150),ILIN(150),IELE(150),ILAT(150),ILON(150),ICODE
     *(150)
      DIMENSION MIN(10),MOUT(24)
      COMMON/NAVCOM/NAVN,INAV,IYR,IDAY,TOTLIN,DEGLIN,TOTIEL,DEGELE,PICLI
     *N,PICELE,IMPSCL,IOYR,IODAY,IM,R1X,R1Y,R1Z,R1DX,R1DY,R1DZ,PITCH,ROL
     *L,YAW,PTIM(3),TMN(3),IMX(3),NLCOEF(2),NRCOEF(2),SCLL0(2),SCLL1(2),
     *ELCOEF(11,2),SCLR0(2),SCLR1(2),ERCOEF(11,2),NASCEF,SCLAS0,SCLAS1,
     *IELEMN,IELEMX,ASCOEF(16)
      DATA PI/3.1415926535/
      DATA IT/5/,LP/6/
      RADDEG=PI/180.0
      LUN=10
      OPEN(UNIT=LUN,NAME='DB0:[350,62]A6LMKS.DAT',TYPE='OLD',READONLY)
2     CONTINUE
      NL=0
10    CONTINUE
      NL=NL+1
      READ(LUN,730,END=40)IYD,ITIME(NL),ICODE(NL),ILIN(NL),IELE(NL),ILAT
     *(NL),ILON(NL)
730   FORMAT(2X,8I9)
      GO TO 10
40    CONTINUE
        NL=NL-1
      CLOSE(UNIT=LUN)
      KC=2HDL
      DO 50 I=1,NL
      WRITE(LP,708)ITIME(I),ICODE(I),ILIN(I),IELE(I),ILAT(I),ILON(I)
708   FORMAT(1X,8I9)
50    CONTINUE
51    CONTINUE
      WRITE(IT,706)IDAY,NL
706   FORMAT(1X,'DAY=',I5,' NUMBER OF LANDMARKS=',I3)
      NLMK=0
      DO 100 I=1,NL
      IF(MOD(ICODE(I)/100,10).NE.0) GO TO 100
      NLMK=NLMK+1
      PTIME(NLMK)=FTIME(ITIME(I))
      ALIN(NLMK)=FLOAT(ILIN(I))
      AELE(NLMK)=FLOAT(IELE(I))
      ALAT(NLMK)=FLALO(ILAT(I))
      ALON(NLMK)=FLALO(ILON(I))
        WRITE(LP,780)PTIME(NLMK),ALIN(NLMK),AELE(NLMK),
     *ALAT(NLMK),ALON(NLMK)
780     FORMAT(1X,6F15.5)
100   CONTINUE
        CLOSE(UNIT=LP)
      IF(NLMK.LE.0) GO TO 990
      CALL ATTITUD(PTIME,ALIN,AELE,ALAT,ALON,NLMK)
      PD=PITCH/RADDEG
      RD=ROLL/RADDEG
```

```
      YD=YAW/RADDEG
      WRITE(IT,701)PD,RD,YD
701   FORMAT(1H0,1X,'PITCH=',E16.9,'  ROLL=',E16.9,'  YAW=',E16.9)
195   CONTINUE
      DO 200 I=1,NL
      PTM=FTIME(ITIME(I))
      XLA=FLALO(ILAT(I))
      XLO=FLALO(ILON(I))
      CALL ES(PTM,XLA,XLO,XLIN,XELE)
      RLIN=ILIN(I)-XLIN
      RELE=IELE(I)-XELE
      WRITE(LP,710)ITIME(I),ICODE(I),RLIN,RELE
200   CONTINUE
201   CONTINUE
710   FORMAT(3X,I6,5X,I5,5X,2F10.2)
      GO TO 999
990   CONTINUE
999   CONTINUE
      CALL REQUES(RAD50('ATSF2'))
      END
```

E-4

****** ATSNV/MCIDAS ****** DAT

```
@ELT,L AF.ATSNV/MCIDAS
ELT007 RLIB62 12/22-17:01:39-(0,)
000      $JOB ATSNAV U3200
000      $OPTION .8,9,20
000      $FORTRAN
000            SUBROUTINE MAIN
000      C     PROGRAM TO NAVIGATE ATS6 IMAGES FROM LANDMARKS.
000            LOGICAL OPTION
000            INTEGER POS,GLEB
000            EQUIVALENCE (ICOM,NAVN)
000            DIMENSION ICOM(1)
000            DIMENSION PTIME(150),ALIN(150),AELE(150),ALAT(150),ALON(150)
000            DIMENSION ITIME(150),ILIN(150),IELE(150),ILAT(150),ILON(150),ICODE
000           *(150)
000            DIMENSION MSQ(10)
000            DIMENSION MIN(10),MOUT(24)
000            DIMENSION GLEB(672)
000            COMMON/NAVCOM/NAVN,INAV,IYR,IDAY,TOTLIN,DEGLIN,TOTIEL,DEGELE,PICLI
000           *N,PICELE,TMPSCL,IOYR,IODAY,TM,R1X,R1Y,R1Z,R1DX,R1DY,R1DZ,PITCH,ROL
000           *L,YAW,PTIM(3),TMN(3),TMX(3),NLCOEF(2),NRCOEF(2),SCLL0(2),SCLL1(2),
000           *ELCOEF(11,2),SCLR0(2),SCLR1(2),ERCOEF(11,2),NASCEF,SCLAS0,SCLAS1,
000           *IELEMN,IELEMX,ASCOEF(16)
000            DATA PI/3.1415926535/
000            DATA MSQ/6HDEFATD,8*0/
000            DATA ICODE/150*0/
000            DATA MIN/6HATSNAV,8*0/
000            CALL IQ(MIN)
000            NDAY=MIN(1)
000            JOUT=1
000            IF(OPTION(MIN(4),3H  P)) JOUT=2
000            IF(.NOT.OPTION(MIN(2),3H  A).AND..NOT.(MIN(2).EQ.0  ))GO TO 195
000            IPT1=MIN(5)
000            IPT2=MIN(6)
000            IPT3=MIN(7)
000            NAVDAY=MOD(NDAY,100000)
000            ISS=NDAY/100000
000            ISS=(ISS/2)*2
000            KDAY=ISS*100000+NAVDAY
000            RADDEG=PI/180.0
000            IF(MIN(5).NE.3HNEW) GO TO 2
000            DO 3 I=1,203
000      3     ICOM(I)=0
000            CALL WCOM
000      2     CONTINUE
000            CALL GETNAV(KDAY,IEXIST)
000            IF(IPT1.NE.0)PTIM(1)=FTIME(IPT1)
000            IF(IPT2.NE.0)PTIM(2)=FTIME(IPT2)
000            IF(IPT3.NE.0)PTIM(3)=FTIME(IPT3)
000            IF(IEXIST.GE.10)GO TO 990
000            CALL HEDDER(KDAY,GLEB,POS)
000            IF(GLEB(POS+1).NE.KDAY.OR.GLEB(POS+2).LT.0)GO TO 990
000            INDEX=6*GLEB(POS+2)+88
000            NL=0
000         10 CALL SCRA(30,INDEX)
000            CALL READW(30,672,GLEB)
000            DO 30 I=1,661,6
000            KIND=GLEB(I)/100000
```

```
000          IF(KIND.EQ.0) GO TO 40
000          IF(KIND.NE.1) GO TO 30
000          IF(NL.GE.150) GO TO 30
000          NL=NL+1
000          ITIME(NL)=GLEB(I+1)
000          ICODE(NL)=MOD(GLEB(I),100000)
000          ILIN(NL)=GLEB(I+2)
000          IELE(NL)=GLEB(I+3)
000          ILAT(NL)=GLEB(I+4)
000          ILON(NL)=GLEB(I+5)
000    30    CONTINUE
000          JJ=GLEB(672)
000          IF(JJ.LT.0) GO TO 40
000          INDEX=6*JJ+88
000          GO TO 10
000    40    CONTINUE
000          KC=2HDL
000.         IF(.NOT.OPTION(MIN(3),3H  L)) GO TO 51
000          DO 50 I=1,NL
000          ENCODE(132,708,MOUT)KC,NDAY,ITIME(I),ICODE(I),ILIN(I),IELE(I),
000         *ILAT(I),ILON(I)
000          CALL TP(JOUT,MOUT)
000    708   FORMAT(1X,A2,8I9)
000    50    CONTINUE
000    51    CONTINUE
000          ENCODE(132,706,MOUT)IDAY,NL
000          CALL TP(JOUT,MOUT)
000    706   FORMAT(1X,'DAY=',I5,' NUMBER OF LANDMARKS=',I3)
000          NLMK=0
000          DO 100 I=1,NL
000          IF(MOD(ICODE(I)/100,10).NE.0) GO TO 100
000          NLMK=NLMK+1
000          PTIME(NLMK)=FTIME(ITIME(I))
000          ALIN(NLMK)=FLOAT(ILIN(I))
000          AELE(NLMK)=FLOAT(IELE(I))
000          ALAT(NLMK)=FLALO(ILAT(I))
000          ALON(NLMK)=FLALO(ILON(I))
000    100   CONTINUE
000          IF(NLMK.LE.0) GO TO 990
000          CALL ATTTUD(PTIME,ALIN,AELE,ALAT,ALON,NLMK)
000          PD=PITCH/RADDEG
000          RD=ROLL/RADDEG
000          YD=YAW/RADDEG
000          ENCODE(132,701,MOUT)PD,RD,YD
000          CALL TP(JOUT,MOUT)
000    701   FORMAT(1H0,1X,'PITCH=',E16.9,'  ROLL=',E16.9,'  YAW=',E16.9)
000          CALL WCOM
000          MSQ(3)=KDAY
000          MSQ(4)=ILALO(PD)
000          MSQ(5)=ILALO(RD)
000          MSQ(6)=ILALO(YD)
000          CALL SQ(MSQ)
000    195   CONTINUE
000          IF(.NOT.OPTION(MIN(2),3H  R).AND..NOT.(MIN(2).EQ.0  ))GO TO 201
000          CALL GETNAV(KDAY,IEXIST)
000          DO 200 I=1,NL
000          PTM=FTIME(ITIME(I))
```

```
000          XLA=FLALO(ILAT(I))
000          XLO=FLALO(ILON(I))
000          CALL ES(PTM,XLA,XLO,XLIN,XELE)
000          RLIN=ILIN(I)-XLIN
000          RELE=IELE(I)-XELE
000          ENCODE(132,710,MOUT)ITIME(I),ICODE(I),RLIN,RELE
000          CALL TP(JOUT,MOUT)
000    200   CONTINUE
000    201   CONTINUE
000    710   FORMAT(3X,I6,5X,I5,5X,2F10.2)
000          RETURN
000    990   CONTINUE
000          CALL EMESS(3HREQ,NDAY)
000          RETURN
000          END$
000    $FILEMA
000    DELETE ATSNAV,GORP
000    $INCLUDE HEDDER
000    $CATALOG
000    NAME=ATSNAV,5,R,W,D
000    TYPE=FG
000    LIB=ATSFLB,LL
000    BEGIN
000    $EOJ

END ELT.
@HDG,P  ******  ATTTUD  ******
```

****** ATTTUD/MCIDAS ****** DAT

```
@ELT,L AF.ATTTUD/MCIDAS
ELT007 RLIB62 12/22-17:01:41-(0,)
000            SUBROUTINE ATTTUD(PTIME,ALIN,AELE,ALAT,ALON,NLMK)
000       C    SUBROUTINE TO COMPUTE ATS6 ATTITUDE FROM LANDMARKS.
000       C    YAW, PITCH, AND ROLL VALUES FOR A SMALL OFFSET OF THE PICTURE
000       C    FRAME COORDINATES FROM BODY CENTERED COORDINATE SYSTEM.
000       C    7 JUNE 1977     G. C. CHATTERS
000            DIMENSION PTIME(1),ALIN(1),AELE(1),ALAT(1),ALON(1)
000            DIMENSION X(3,150),Y(3,150),TIME(150),PRY(3)
000            COMMON/NAVCOM/NAVN,INAV,IYR,IDAY,TOTLIN,DEGLIN,TOTIEL,DEGELE,PICLI
000           *N,PICELE,TMPSCL,IOYR,IODAY,TM,R1X,R1Y,R1Z,R1DX,R1DY,R1DZ,PITCH,ROL
000           *L,YAW,PTIM(3),TMN(3),TMX(3),NLCOEF(2),NRCOEF(2),SCLL0(2),SCLL1(2),
000           *ELCOEF(11,2),SCLR0(2),SCLR1(2),ERCOEF(11,2),NASCEF,SCLAS0,SCLAS1,
000           *IELEMN,IELEMX,ASCOEF(16)
000            COMMON/MINCOM/X,Y,TIME,NP
000            DIMENSION MOUT(24)
000            DATA PI/3.14159265/
000            DATA LP/6/
000            RDPDG=PI/180.0
000            RADLIN=RDPDG*DEGLIN/TOTLIN
000            RADELE=RDPDG*DEGELE/TOTIEL
000            NP=NLMK
000            DO 100 I=1,NLMK
000            ISCAN=1200-(IFIX(ALIN(I))-1)/2
000            TIME(I)=PTIME(I)+ISCAN*TMPSCL
000            IDIR=1
000       C    CONVERT LAT , LON TO ROTATING EARTH CO-OR
000            CALL ERTOER(ALAT(I),ALON(I),X1ER,X2ER,X3ER,IDIR)
000       C    CONVERT ROTATING EARTH TO INERTIAL EARTH COORDINATES
000            CALL ERTOST(X1ER,X2ER,X3ER,X1,X2,X3,IDIR,TIME(I))
000       C    EARTH INERTIAL TO SATELLITE LOCAL VERTICAL
000            CALL STTOLV(X1,X2,X3,IDIR,TIME(I))
000       C    NOTE:ATTITUDE DATA NOT USED THUS LOCAL VERTICAL SAME AS BODY CENTE
000            X(1,I)=X1
000            X(2,I)=X2
000            X(3,I)=X3
000       100  CONTINUE
000            DO 200 I=1,NLMK
000            ELEANG=(PICELE-AELE(I))*RADELE
000            ALNANG=(PICLIN-ALIN(I))*RADLIN
000            Y(1,I)=-SIN(ELEANG)*COS(ALNANG)
000            Y(2,I)=-SIN(ALNANG)
000            Y(3,I)=COS(ELEANG)*COS(ALNANG)
000       200  CONTINUE
000            PRY(1)=0.
000            PRY(2)=0.
000            PRY(3)=0.
000            CALL MINMIZ(PRY,PRY,GNORM,VALUE,ITER)
000            PITCH=PRY(1)
000            ROLL=PRY(2)
000            YAW=PRY(3)
000            ENCODE(132,702,MOUT)ITER,GNORM,VALUE
000       702  FORMAT(1H0,5X,'CONVERGENCE AT ITERATION',I6,//6X,'GRADIENT NORM=',
000           *E16.9,'   S FINAL VALUE=',E16.9)
000            CALL TQ(MOUT)
000            RETURN
000            END$
```

E-8

A6INT/AOIPS

```
      INTEGER*4 ITIME
      DIMENSION MIN(10),RMIN(10),R1(3),R2(3)
      COMMON/NAVCOM/NAVN,INAV,IYR,IDAY,TOTLIN,DEGLIN,TOTIEL,DEGELE,PICLI
     *N,PICELE,IMPSCL,IOYR,IODAY,IM,R1X,R1Y,R1Z,R1DX,R1DY,R1DZ,PITCH,ROL
     *L,YAW,PTIM(3),IMN(3),IMX(3),NLCOEF(2),NRCOEF(2),SCLL0(2),SCLL1(2),
     *ELCOEF(11,2),SCLR0(2),SCLR1(2),ERCOEF(11,2),NASCEF,SCLAS0,SCLAS1,
     *IELEMN,IELEMX,ASCOEF(16)
      DATA PI/3.1415926535/
      DATA LP/6/
C     PROGRAM TO INITIALIZE ATS 6 NAVCOM
      RADDEG=PI/180.0
      MENU=1
      CALL INCOM(' NAVCOM TO BE PRINTED? (1=YES, DEFAULT=NO)',42,1,1,MIN
     *,MENU,1,1)
      IF(MENU.LT.0) GO TO 999
      JPRT=MIN(1)
      MENU=1
      CALL INCOM(' ENTER DAY NUMBER',17,1,1,MIN,MENU,1,1)
      IF(MENU.LE.0) GO TO 999
      INAV=1111
      TOTLIN=2400.0
      DEGLIN=19.92
      TOTIEL=2400.0
      DEGELE=20.07
      PICLIN=(TOTLIN+1.0)/2.0
      PICELE=(TOTIEL+1.0)/2.0
      IMPSCL=0.02
      IYR=74
      IOYR=74
      IDAY=MIN(1)
      IODAY=IDAY
      MENU=3
      CALL INCOM(' ENTER 3 PICTURE START TIMES - HHMMSS',37,1,1,RMIN,MEN
     *U,3,1)
      IF(MENU.LE.0) GO TO 999
      ITIME=RMIN(1)
      PTIM(1)=FTIME(ITIME)
      ITIME=RMIN(2)
      PTIM(2)=FTIME(ITIME)
      ITIME=RMIN(3)
      PTIM(3)=FTIME(ITIME)
      DO 100 I=1,3
      IMN(I)=0.0
      IMX(I)=0.0
100   CONTINUE
      MENU=4
      CALL INCOM(' ENTER FIRST ORBIT POSITION: T(HHMMSS), X, Y, Z (KM)',
     *52,1,1,RMIN,MENU,3,1)
      IF(MENU.LE.0) GO TO 999
      ITIME=RMIN(1)
      T1=FTIME(ITIME)
      R1(1)=RMIN(2)
      R1(2)=RMIN(3)
```

A6INT/AOIPS

```
      R1(3)=RMIN(4)
      MENU=4
      CALL INCOM(' ENTER SECOND ORBIT POSITION: T(HHMMSS), X, Y, Z (KM)'
     *,53,1,1,RMIN,MENU,3,1)
      IF(MENU.LE.0) GO TO 999
      ITIME=RMIN(1)
      T2=FTIME(ITIME)
      R2(1)=RMIN(2)
      R2(2)=RMIN(3)
      R2(3)=RMIN(4)
      CALL GASORB(R1,T1,R2,T2)
 999  CONTINUE
      IF(JPRT.LE.0) GO TO 1000
      WRITE(LP,761)NAVN,INAV,IYR,IDAY
 761  FORMAT(1H1,'NAVN=',I5,T27,'INAV=',I10,T52,'YEAR=',I2,T77,'DAY=',I3
     *)
      WRITE(LP,762) TOTLIN,DEGLIN,TOTIEL,DEGELE
 762  FORMAT(T2,'TOTLIN=',F10.2,T27,'DEGLIN=',F10.2,T52,'TOTIEL=',F10.2,
     *T77,'DEGELE=',F10.2)
      WRITE(LP,763)PICLIN,PICELE,IMPSCL
 763  FORMAT(T2,'PICLIN=',F10.2,T27,'PICELE=',F10.2,T52,'IMPSCL=',F10.2)
      WRITE(LP,764)IOYR,IODAY
 764  FORMAT(T2,'ORBIT YEAR=',I4,T27,'ORBIT DAY=',I4)
      WRITE(LP,765)TM
 765  FORMAT(T2,'ORBIT TIME, TM=',F15.4)
      WRITE(LP,766)R1X,R1Y,R1Z
 766  FORMAT(T2,'POSITION (KM)',T27,'R1X=',F15.3,T52,'R1Y=',F15.4,T77,'R
     *1Z=',F15.4)
      WRITE(LP,767)R1DX,R1DY,R1DZ
 767  FORMAT(T2,'VELOCITY',T27,'R1DX=',E15.9,T52,'R1DY=',E15.9,T77,'R1DZ
     *=',E15.9)
      PD=PITCH/RADDEG
      RD=ROLL/RADDEG
      YD=YAW/RADDEG
      WRITE(LP,768)PD,RD,YD
 768  FORMAT(T2,'ATTITUDE (DEG)',T27,'PITCH=',E15.9,T52,'ROLL=',E15.9,T7
     *7,'YAW=',E15.9)
      WRITE(LP,769)PTIM
 769  FORMAT(T2,'PICTURE TIMES',T27,3F25.4)
 770  FORMAT(1X,5I25)
 771  FORMAT(1X,5E25.9)
      WRITE(LP,770)NLCOEF,NRCOEF,NASCEF
      WRITE(LP,771)TMN,TMX
      WRITE(LP,770)NLCOEF,NRCOEF
      WRITE(LP,771) SCLL0,SCLL1
      WRITE(LP,771)ELCOEF
      WRITE(LP,771) SCLR0,SCLR1
      WRITE(LP,771) ERCOEF
      WRITE(LP,770)NASCEF,IELEMN,IELEMX
      WRITE(LP,771) SCLAS0,SCLAS1
      WRITE(LP,771) ASCOEF
 1000 CONTINUE
      CALL REQUES(RAD50('ATSF2'))
      END
```

****** BCTOPF/MCIDAS ******

DAT

```
@ELT,L AF.BCTOPF/MCIDAS
ELT007 RLIB62 12/22-17:01:42-(0,)
000            SUBROUTINE BCTOPF(X,Y,Z,IDIR)
000            DIMENSION A(3,3)
000            COMMON/NAVCOM/NAVN,INAV,IYR,IDAY,TOTLIN,DEGLIN,TOTIEL,DEGELE,PICLI
000           *N,PICELE,TMPSCL,IOYR,IODAY,TM,R1X,R1Y,R1Z,R1DX,R1DY,R1DZ,PITCH,ROL
000           *L,YAW,PTIM(3),TMN(3),TMX(3),NLCOEF(2),NRCOEF(2),SCLL0(2),SCLL1(2),
000           *ELCOEF(11,2),SCLR0(2),SCLR1(2),ERCOEF(11,2),NASCEF,SCLAS0,SCLAS1,
000           *IELEMN,IELEMX,ASCOEF(16)
000      C     PT(3) = YAW, PT(2) = ROLL, PT(1) = PITCH
000            CALL UNIT(A)
000            CALL ROTATE(A,YAW,3,1)
000            CALL ROTATE(A,ROLL,1,1)
000            CALL ROTATE(A,PITCH,2,1)
000            IF(IDIR.EQ.2)GO TO 10
000            XT = X*A(1,1)+Y*A(1,2)+Z*A(1,3)
000            YT = X*A(2,1)+Y*A(2,2)+Z*A(2,3)
000            ZT = X*A(3,1)+Y*A(3,2)+Z*A(3,3)
000            X=XT
000            Y=YT
000            Z=ZT
000            RETURN
000      10    XT=X*A(1,1)+Y*A(2,1)+Z*A(3,1)
000            YT = X*A(1,2)+Y*A(2,2)+Z*A(3,2)
000            Z = X*A(1,3)+Y*A(2,3)+Z*A(3,3)
000            X=XT
000            Y=YT
000            RETURN
000            END$

END ELT.
@HDG,P  ******  CNPS/IBMSSP  ******
```

****** EATOST/MCIDAS ****** DAT

```
@ELT,L AF.EATOST/MCIDAS
ELT007 RLIB62 12/22-17:01:43-(0,)
000              SUBROUTINE EATOST(PICTIM,LNS,IES,LA,LO,IUNK,INAV,BETA,BETDOT,AF)
000              PTIME=PICTIM
000              IF(IUNK.EQ.1) GO TO 1
000              IF(IUNK.EQ.2) GO TO 2
000              RETURN
000     1        CONTINUE
000              XLIN=LNS
000              XELE=IES
000              CALL SE(PTIME,XLIN,XELE,XLAT,XLON)
000              LA=ILALO(XLAT)
000              LO=ILALO(XLON)
000              RETURN
000     2        CONTINUE
000              XLAT=FLALO(LA)
000              XLON=FLALO(LO)
000              CALL ES(PTIME,XLAT,XLON,XLIN,XELE)
000              LNS=XLIN
000              IES=XELE
000              RETURN
000              END

END ELT.
@HDG,P  ******  EDGCOR/MCIDAS  ******
```

```
 ******  EDGCOR/MCIDAS  ******                                                                          DAT

@ELT,L AF.EDGCOR/MCIDAS
ELT007 RLIB62 12/22-17:01:44-(0,)
      000              SUBROUTINE EDGCOR(PTIME,ALIN,DELLIN,DELELE)
      000              COMMON/NAVCOM/NAVN,INAV,IYR,IDAY,TOTLIN,DEGLIN,TOTIEL,DEGELE,PICLI
      000             *N,PICELE,TMPSCL,IOYR,IODAY,TM,R1X,R1Y,R1Z,R1DX,R1DY,R1DZ,PITCH,ROL
      000             *L,YAW,PTIM(3),TMN(3),TMX(3),NLCOEF(2),NRCOEF(2),SCLL0(2),SCLL1(2),
      000             *ELCOEF(11,2),SCLR0(2),SCLR1(2),ERCOEF(11,2),NASCEF,SCLAS0,SCLAS1,
      000             *IELEMN,IELEMX,ASCOEF(16)
      000              DATA A/522.07/,LC/1235/
      000              ISCAN=1200-(IFIX(ALIN-1))/2
      000              TIME=PTIME+ISCAN*TMPSCL
      000              DO 20 IC=2,3
      000       20     IF(TMN(IC).LE.TIME.AND.TIME.LE.TMX(IC)) GO TO 100
      000              DELLIN=0.0
      000              DELELE=0.0
      000              RETURN
      000      100     CONTINUE
      000              IX=IC-1
      000              XL=ISCAN*SCLL1(IX)+SCLL0(IX)
      000              CALL CNPS(EL,XL,ELCOEF(1,IX),NLCOEF(IX))
      000              XR=ISCAN*SCLR1(IX)+SCLR0(IX)
      000              CALL CNPS(ER,XR,ERCOEF(1,IX),NRCOEF(IX))
      000              DELELE=(EL+ER)/2.0
      000              MSC=1200-(LC-1)/2
      000              Y=ISCAN-MSC
      000              XX=SQRT(A**2-Y**2)
      000              DELTA=(ER-EL)/2.0
      000              RAD=Y**2-2*XX*DELTA-DELTA**2
      000              DELLIN=SQRT(RAD)-Y
      000              RETURN
      000              END$

END ELT.
@HDG,P  ******  EDGFT/MCIDAS  ******
```

****** EDGFT/MCIDAS ****** DAT

```
@ELT,L AF.EDGFT/MCIDAS
ELT007 RLIB62 12/22-17:01:44-(0,)
       000     $JOB EDGFIT U3200
       000     $OPTION .8,9,13,20
       000     $FORTRAN
       000           SUBROUTINE MAIN
       000           DIMENSION JWIN(11,10)
       000           DIMENSION DATI(601),WORK(231),COEF(21)
       000           DIMENSION ISCNL(300),ISCNR(300),IUL(300),IUR(300),NOUT(17)
       000           DIMENSION MIN(10)
       000           DIMENSION MOUT(44)
       000           COMMON/NAVCOM/NAVN,INAV,IYR,IDAY,TOTLIN,DEGLIN,TOTIEL,DEGELE,PICLI
       000          *N,PICELE,TMPSCL,IOYR,IODAY,TM,R1X,R1Y,R1Z,R1DX,R1DY,R1DZ,PITCH,ROL
       000          *L,YAW,PTIM(3),TMN(3),TMX(3),NLCOEF(2),NRCOEF(2),SCLL0(2),SCLL1(2),
       000          *ELCOEF(11,2),SCLR0(2),SCLR1(2),ERCOEF(11,2),NASCEF,SCLAS0,SCLAS1,
       000          *IELEMN,IELEMX,ASCOEF(16)
       000           DATA TMNL/1440.0/,TMXL/0.0/,TMNR/1440.0/,TMXR/0.0/
       000           DATA LUN/10/,JSECSV/-1/,NUMPER/10/,NUMENT/11/,JOUT/2/
       000           DATA MIN/6HGCCGCC,8*0/
       000           CALL IQ(MIN)
       000           IP=11
       000           CALL RCOM
       000           ENCODE(132,701,MOUT)PTIM
       000     701   FORMAT(5X,3F20.6)
       000           CALL TP(1,MOUT)
       000           CALL VARASG('WIN',LUN)
       000           CALL OPN(LUN)
       000           DO 500 IPAIR=1,2
       000           NL=0
       000           NR=0
       000           TMNL=1440.0
       000           TMXL=0.0
       000           TMNR=1440.0
       000           TMXR=0.0
       000           CALL SCRA(LUN,0)
       000           IK=0
       000     90    CONTINUE
       000     100   CONTINUE
       000           JSEC=IK/NUMPER
       000           JVEC=MOD(IK,NUMPER)+1
       000           IF(JSEC.NE.JSECSV) CALL READW(LUN,110,JWIN)
       000           IF(JSEC.NE.JSECSV) JSECSV=JSEC
       000           IK=IK+1
       000           IF(JWIN(1,JVEC).EQ.0) GO TO 400
       000           IH=MOD(JWIN(4,JVEC),10)
       000           IF(IH.GT.4) GO TO 100
       000           T1=FTIME(JWIN(2,JVEC))
       000           T2=FTIME(JWIN(3,JVEC))
       000           IF(T1.NE.PTIM(1     ).OR.T2.NE.PTIM(IPAIR+1))GO TO 100
       000           IF(MOD(JWIN(4,JVEC)/100,1000).NE.100) GO TO 100
       000           IY1=JWIN(5,JVEC)
       000           IY2=JWIN(7,JVEC)
       000           IF(IY1.NE.IY2) GO TO 100
       000           IX1=JWIN(6,JVEC)
       000           IF(IX1.GE.PICELE) GO TO 200
       000           NL=NL+1
       000           IF(NL.GT.300) GO TO 100
```

```
000        ISCNL(NL)=1200-(IY1-1)/2
000        IUL(NL)=JWIN(9,JVEC)*(T2-T1)/60.0
000        GO TO 100
000   200  CONTINUE
000        NR=NR+1
000        IF(NR.GT.300) GO TO 100
000        ISCNR(NR)=1200-(IY1-1)/2
000        IUR(NR)=JWIN(9,JVEC)*(T2-T1)/60.0
000        GO TO 100
000   400  CONTINUE
000        IF(NL.LT.IP+1) GO TO 459
000        DO 410 I=1,NL
000        TS=PTIM(IPAIR+1)+ISCNL(I)*TMPSCL
000        TMNL=AMIN1(TMNL,TS)
000        TMXL=AMAX1(TMXL,TS)
000        DATI(I)=ISCNL(I)
000   410  DATI(I+NL)=IUL(I)/100.0
000        DATI(2*NL+1)=-1.0
000        CALL APCH(DATI,NL,IP,XD,XO,WORK,IER)
000        IF(IER.LT.0) GO TO 990
000        SCLLO(IPAIR)=XO
000        SCLL1(IPAIR)=XD
000        EPS=1.0E-4
000        IOP=+1
000        ETA=1.0E-3
000        CALL APFS(WORK,IP,IRES,IOP,EPS,ETA,IER)
000        IF(IER.LT.0) GO TO 990
000        NLCOEF(IPAIR)=IRES
000        ENCODE(132,706,MOUT)IRES
000        CALL TQ(MOUT)
000        IX=IRES*(IRES-1)/2
000        DO 420 I=1,IRES
000        COEF(I)=WORK(I+IX)
000        ENCODE(132,705,MOUT)I,COEF(I)
000   705  FORMAT(2X,'COEF(',I2,')=',E20.9)
000        CALL TQ(MOUT)
000   420  ELCOEF(I,IPAIR)=COEF(I)
000   459  CONTINUE
000        IF(NR.LT.IP+1) GO TO 490
000        DO 460 I=1,NR
000        TS=PTIM(IPAIR+1)+ISCNR(I)*TMPSCL
000        TMNR=AMIN1(TMNR,TS)
000        TMXR=AMAX1(TMXR,TS)
000        DATI(I)=ISCNR(I)
000   460  DATI(I+NR)=IUR(I)/100.0
000        DATI(2*NR+1)=-1.0
000        CALL APCH(DATI,NR,IP,XD,XO,WORK,IER)
000        IF(IER.LT.0) GO TO 990
000        SCLRO(IPAIR)=XO
000        SCLR1(IPAIR)=XD
000        EPS=1.0E-4
000        IOP=+1
000        ETA=1.0E-3
000        CALL APFS(WORK,IP,IRES,IOP,EPS,ETA,IER)
000        IF(IER.LT.0) GO TO 990
000        NRCOEF(IPAIR)=IRES
000        ENCODE(132,706,MOUT)IRES
```

****** EDGFT/MCIDAS ****** DAT

```
000   706    FORMAT(2X,'DIMENSION=',I3)
000          CALL TQ(MOUT)
000          IX=IRES*(IRES-1)/2
000          DO 470 I=1,IRES
000          COEF(I)=WORK(I+IX)
000          ENCODE(132,705,MOUT)I,COEF(I)
000          CALL TQ(MOUT)
000   470    ERCOEF(I,IPAIR)=COEF(I)
000   490    CONTINUE
000          TMN(IPAIR+1)=AMAX1(TMNL,TMNR)
000          TMX(IPAIR+1)=AMIN1(TMXL,TMXR)
000   500    CONTINUE
000          CALL WCOM
000          CALL EMESS(3HFIN,0)
000          RETURN
000   990    CONTINUE
000          CALL TQ(72H ERROR RETURN FROM APCH, APFS.                    )
000         *
000          RETURN
000          END$
000   $FILEMA
000   DELETE GCCGCC,GORP
000   $INCLUDE ATSSSP
000   $CATALOG
000   NAME=GCCGCC,5,R,W,D
000   TYPE=FG
000   LIB=ATSFLB,LL
000   BEGIN
000   $EOJ

END ELT.
@HDG,P  ******  ERTOER  ******
```

****** ERTOER/MCIDAS ******

DA

```
@ELT,L AF.ERTOER/MCIDAS
ELT007 RLIB62 12/22-17:01:46-(0,)
000             SUBROUTINE ERTOER(XLAT,XLON,XE,YE,ZE,IDR)
000            PI=3.14159265
000            RDPDG=PI/180.0
000            A=6378.15
000             B=6356.77
000            ESQ=(A-B)*(A+B)/A**2
000             IF(IDR .EQ. 2)GO TO 10
000             XLT=XLAT*RDPDG
000             XLN=XLON*RDPDG
000             CLT=COS(XLT)
000             SLT=SIN(XLT)
000             CLN=COS(XLN)
000             SLN=SIN(XLN)
000             RR=A/SQRT(1.-ESQ*SLT**2)
000             XE=CLT*CLN*RR
000             YE=CLT*SLN*RR
000             ZE=SLT*RR*(1.-ESQ)
000             RETURN
000     10      CONTINUE
000             IF(((XE**2+YE**2+ZE**2) .LT. B**2) .OR. ((XE**2+YE**2+ZE**2)
000          *   .GT. A**2))GO TO 15
000             XLON=ATAN2(YE,XE)/RDPDG
000            XLAT=ATAN(ZE/((1.-ESQ)*SQRT(XE**2+YE**2)))/RDPDG
000             RETURN
000     15      XLAT=100.
000             XLON=200.
000             RETURN
000            END$

END ELT.
@HDG,P  ******  ERTOST  ******
```

```
******  ERTOST/MCIDAS  ******

@ELT,L AF.ERTOST/MCIDAS
ELT007 RLIB62 12/22-17:01:47-(0,)
000          SUBROUTINE ERTOST(XE,YE,ZE,X,Y,Z,IDIR,TIME)
000    C      PARAMETER LIST
000    C          XE,YE,ZE=EARTH COORDS OF LANDMARK
000    C          X,Y,Z=POINTING VEC IN INERTIAL COORDS TO LANDMARK
000    C          IF IDIR=1, EARTH TO POINTING
000    C          IF IDIR=2, POINTING TO EARTH
000    C           ITIME=HHMMSS OF CURRENT POINTING VECTOR
000          COMMON/NAVCOM/NAVN,INAV,IYR,IDAY,TOTLIN,DEGLIN,TOTIEL,DEGELE,PICLI
000         *N,PICELE,TMPSCL,IOYR,IODAY,TM,R1X,R1Y,R1Z,R1DX,R1DY,R1DZ,PITCH,ROL
000         *L,YAW,PTIM(3),TMN(3),TMX(3),NLCOEF(2),NRCOEF(2),SCLLO(2),SCLL1(2),
000         *ELCOEF(11,2),SCLRO(2),SCLR1(2),ERCOEF(11,2),NASCEF,SCLASO,SCLAS1,
000         *IELEMN,IELEMX,ASCOEF(16)
000          DDD=FLOAT(MOD(IDAY,1000))
000           A=6378.15
000           B=6356.77
000           R1=99.59477026
000           R2=.985647336
000           R3=.2506844773
000          PI=3.14159265
000          RDPDG=PI/180.0
000           R=(R1+R2*DDD+R3*TIME)
000           R=AMOD(R,360.)*RDPDG
000           CALL ORBIT(XS,YS,ZS,TIME)
000           CR=COS(R)
000           SR=SIN(R)
000           IF(IDIR .EQ. 2)GO TO 10
000           X1=CR*XE-SR*YE
000           Y1=SR*XE+CR*YE
000           Z1=ZE
000           X=X1-XS
000           Y=Y1-YS
000           Z=Z1-ZS
000          CALL NRMLIZ(X,Y,Z,RNORM)
000           RETURN
000    10     CONTINUE
000          F=B**2/A**2
000          AQ=F+(1.-F)*Z**2
000          BQ=2.*((X*XS+Y*YS)*F+Z*ZS)
000          CQ=(XS**2+YS**2)*F+ZS**2-B**2
000          RAD=BQ**2-4.*AQ*CQ
000          IF(RAD.LT.0.) GO TO 15
000          S=-(BQ+SQRT(RAD))/(2.*AQ)
000           X2=XS+X*S
000           Y2=YS+Y*S
000           Z2=ZS+Z*S
000           XE=CR*X2+SR*Y2
000           YE=-SR*X2+CR*Y2
000           ZE=Z2
000           RETURN
000    15    WRITE(6,1000)
000    1000  FORMAT(1X,'THE SUBROUTINE ERTOST (IDIR=2) HAS RECEIVED BAD
000         * INPUT DATA.')
000           RETURN
000          END$
```

****** ES/MCIDAS ******

DA

```
@ELT,L AF.ES/MCIDAS
ELT007 RLIB62 12/22-17:01:48-(0,)
000          SUBROUTINE ES(PTIME,XLAT,XLON,XLIN,XELE)
000          COMMON/NAVCOM/NAVN,INAV,IYR,IDAY,TOTLIN,DEGLIN,TOTIEL,DEGELE,PICLI
000         *N,PICELE,TMPSCL,IOYR,IODAY,TM,R1X,R1Y,R1Z,R1DX,R1DY,R1DZ,PITCH,ROL
000         *L,YAW,PTIM(3),TMN(3),TMX(3),NLCOEF(2),NRCOEF(2),SCLL0(2),SCLL1(2),
000         *ELCOEF(11,2),SCLR0(2),SCLR1(2),ERCOEF(11,2),NASCEF,SCLAS0,SCLAS1,
000         *IELEMN,IELEMX,ASCOEF(16)
000          DATA INIT/0/
000          ALON=XLON
000          ALAT=XLAT
000          TIME=PTIME
000          IDIR=1
000          ALINSV=0
000          AELESV=0
000          DO 100 II=1,5
000          CALL ERTOER(ALAT,ALON,X1ER,X2ER,X3ER,IDIR)
000          CALL ERTOST(X1ER,X2ER,X3ER,X1,X2,X3,IDIR,TIME)
000          CALL STTOLV(X1,X2,X3,IDIR,TIME)
000   C      BC SAME AS LV HERE.
000          CALL BCTOPF(X1,X2,X3,IDIR)
000          CALL PFTOTC(ALIN,AELE,X1,X2,X3,IDIR,INIT)
000          CALL EDGCOR(PTIME,ALIN,DELLIN,DELELE)
000          AELEC=AELE+DELELE
000          ALINC=ALIN+DELLIN
000          IF(ABS(ALINC-ALINSV).LE.0.5) GO TO 150
000          ALINSV=ALINC
000          AELESV=AELEC
000          ISCAN=1200-(IFIX(ALIN-1))/2
000          TIME=PTIME+ISCAN*TMPSCL
000   100    CONTINUE
000   150    CONTINUE
000          XLIN=ALINC
000          XELE=AELEC
000          RETURN
000          END

END ELT.
@HDG,P  ******  FLALO  ******
```

```
 ******  FLALO/MCIDAS  ******

@ELT,L AF.FLALO/MCIDAS
ELT007 RLIB62 12/22-17:01:49-(0,)
       000           FUNCTION FLALO(M)
       000           IF (M .LT. 0)GO TO 1
       000           N=M
       000           X=1.0
       000           GO TO 2
       000    1      N=-M
       000           X=-1.0
       000    2      FLALO=FLOAT(N/10000)+FLOAT(MOD(N/100,100))/60.+FLOAT(MOD(N,100))/
       000          * 3600.
       000           FLALO=FLALO*X
       000           RETURN
       000          END$

END ELT.
@HDG,P  ******  FLIP  ******
```

****** FLIP/MCIDAS ******

```
@ELT,L AF.FLIP/MCIDAS
ELT007 RLIB62 12/22-17:01:50-(0,)
 000          SUBROUTINE FLIP(A,B,I,N,ALTRET)
 000     C
 000     C   THIS ROUTINE IS CALLED ONLY BY SUBROUTINE INVERT AND IS USED TO
 000     C   PERFORM AN ELEMENTARY ROW OPERATION ON MATRICES A AND B.
 000     C
 000          LOGICAL ALTRET
 000          DIMENSION A(N,N),B(N,N)
 000          ALTRET=.FALSE.
 000          LIM=I+1
 000          DO 2 L=LIM,N
 000          IF( ABS(A(L,I)).LT.1.0E-32)GOTO 2
 000          DO 1 M=1,N
 000          A(I,M)=A(I,M)+A(L,M)
 000     1    B(I,M)=B(I,M)+B(L,M)
 000          RETURN
 000     2    CONTINUE
 000          ALTRET=.TRUE.
 000          RETURN
 000          END$

END ELT.
@HDG,P  ******  FTIME/MCIDAS  ******
```

****** FTIME/MCIDAS ******

DAT

```
@ELT,L AF.FTIME/MCIDAS
ELT007 RLIB62 12/22-17:01:51-(0,)
000001      000          FUNCTION FTIME(ITIME)
000002      000          IH=ITIME/10000
000003      000          HRS=IH*60.
000004      000          AMNS=FLOAT((ITIME-(ITIME/10000)*10000)/100)
000005      000          SECS=FLOAT(MOD(ITIME,100))/60.
000006      000          FTIME=HRS+AMNS+SECS
000007      000          RETURN
000008      000         END$

END ELT.
@HDG,P  ******  GASORB/MCIDAS  ******
```

E-22

```
****** GASORB/MCIDAS ******

@ELT,L AF.GASORB/MCIDAS
ELT007 RLIB62 12/22-17:01:51-(0,)
 000          SUBROUTINE GASORB(R1,T1,R2,T2)
 000          REAL KE,L,M,MU
 000          DIMENSION R1(3),R2(3)
 000          COMMON/NAVCOM/NAVN,INAV,IYR,IDAY,TOTLIN,DEGLIN,TOTIEL,DEGELE,PICLI
 000         *N,PICELE,TMPSCL,IOYR,IODAY,TM,R1X,R1Y,R1Z,R1DX,R1DY,R1DZ,PITCH,ROL
 000         *L,YAW,PTIM(3),TMN(3),TMX(3),NLCOEF(2),NRCOEF(2),SCLL0(2),SCLL1(2),
 000         *ELCOEF(11,2),SCLR0(2),SCLR1(2),ERCOEF(11,2),NASCEF,SCLAS0,SCLAS1,
 000         *IELEMN,IELEMX,ASCOEF(16)
 000          DATA RE,KE,MU/6378.15,0.07436574,1.0/
 000          TM=T1
 000          DO 5 I=1,3
 000          R1(I)=R1(I)/RE
 000          R2(I)=R2(I)/RE
 000        5 CONTINUE
 000          R1X=R1(1)
 000          R1Y=R1(2)
 000          R1Z=R1(3)
 000          R1NORM=SQRT(R1(1)**2+R1(2)**2+R1(3)**2)
 000          R2NORM=SQRT(R2(1)**2+R2(2)**2+R2(3)**2)
 000          TAU=KE*(T2-T1)
 000          R1R2NM=R1NORM*R2NORM
 000          COSANM=(R1(1)*R2(1)+R1(2)*R2(2)+R1(3)*R2(3))/R1R2NM
 000          CSHFAN=SQRT((1+COSANM)/2)
 000          L=(R1NORM+R2NORM)/(4*SQRT(R1R2NM)*CSHFAN)-.5
 000          M=(MU*TAU**2)/(2*SQRT(R1R2NM)*CSHFAN)**3
 000          YU=1
 000       10 YL=YU
 000          XL=M/YL**2-L
 000          CHFEAN=1-2*XL
 000          SHFEAN=SQRT(4*XL*(1-XL))
 000          ECANOM=2*ATAN2(SHFEAN,CHFEAN)
 000          XU=(ECANOM-SIN(ECANOM))/SHFEAN**3
 000          YU=1+XU*(L+XL)
 000          IF(ABS(YU-YL).GE.1.0D-6)GOTO10
 000          YL=YU
 000          A=((TAU*SQRT(MU))/(2*YL*SQRT(R1R2NM)*CSHFAN*SHFEAN))**2
 000          F=1-A/R1NORM*(1-COS(ECANOM))
 000          G=TAU-A**1.5/SQRT(MU)*(ECANOM-SIN(ECANOM))
 000          R1DX=(R2(1)-F*R1(1))/G
 000          R1DY=(R2(2)-F*R1(2))/G
 000          R1DZ=(R2(3)-F*R1(3))/G
 000          RETURN
 000          END

END ELT.
@HDG,P  ****** INVERT/MCIDAS ******
```

****** INVERT/MCIDAS ****** DAT

```
@ELT,L AF.INVERT/MCIDAS
ELT007 RLIB62 12/22-17:01:52-(0,)
000           SUBROUTINE INVERT(AA,B,N,ALTRET)
000     C
000     C     THIS ROUTINE RETURNS IN B THE INVERSE OF THE N-DIMENSIONAL MATRIX AA.
000     C     IF AA IS SINGULAR, A RETURN IS TAKEN THROUGH LABEL $.
000     C
000           LOGICAL ALTRET,LOGTMP
000           DIMENSION AA(N,N),B(N,N),A(3,3)
000           DATA TV/1.0E35/
000           ALTRET=.FALSE.
000           DO 1 I=1,N
000           DO 1 J=1,N
000     1     A(I,J)=AA(I,J)
000           DO 2 I=1,N
000           DO 2 J=1,N
000           B(I,J)=0
000     2     IF(I.EQ.J)B(I,J)=1.0
000           LIM=N-1
000           DO 4 I=1,LIM
000           IF(ABS(A(I,I)).LT.1.0E-30) CALL FLIP(A,B,I,N,ALTRET)
000           LOGTMP=ALTRET
000           ALTRET=.FALSE.
000           IF(LOGTMP) GO TO 101
000           LIM2=I+1
000           DO 4 J=LIM2,N
000           IF(TV*ABS(A(I,I)).LE.ABS(A(J,I))) ALTRET=.TRUE.
000           IF(TV*ABS(A(I,I)).LE.ABS(A(J,I))) RETURN
000           FACTOR=A(J,I)/A(I,I)
000           DO 3 K=LIM2,N
000     3     A(J,K)=A(J,K)-FACTOR*A(I,K)
000           DO 4 K=1,N
000     4     B(J,K)=B(J,K)-FACTOR*B(I,K)
000           DO 5 I=N,2,-1
000           LIM3=I-1
000           DO 5 J=LIM3,1,-1
000           IF(TV*ABS(A(I,I)).LE.ABS(A(J,I))) ALTRET=.TRUE.
000           IF(TV*ABS(A(I,I)).LE.ABS(A(J,I))) RETURN
000           FACTOR=A(J,I)/A(I,I)
000           DO 5 K=1,N
000     5     B(J,K)=B(J,K)-FACTOR*B(I,K)
000           DO 6 I=1,N
000           FACTOR=A(I,I)
000           DO 6 J=1,N
000           IF(TV*ABS(FACTOR).LE.ABS(B(I,J))) ALTRET=.TRUE.
000           IF(TV*ABS(FACTOR).LE.ABS(B(I,J))) RETURN
000     6     B(I,J)=B(I,J)/FACTOR
000           RETURN
000     101   ALTRET=.TRUE.
000           RETURN
000           END$

END ELT.
@HDG,P ****** LATSF/MCIDAS ******
```

LATSF/AOIPS

```
C       PROGRAM TO READ IMAGES FROM ATS-6 TAPES.
        INTEGER*4 ITIMX
        LOGICAL IFVIS,IFIR,OPTION
        DIMENSION MIN(10),RMIN(10),IARRAY(5600)
        EQUIVALENCE (IDIR,IAROW)
        DIMENSION IDIR(256),IAROW(256)
        DIMENSION MODAY(12)
        COMMON/NAVCOM/NAVN,INAV,IYR,IDAY,TOTLIN,DEGLIN,TOTIEL,DEGELE,PICLI
       *N,PICELE,IMPSCL,IOYR,IODAY,IM,R1X,R1Y,R1Z,R1DX,R1DY,R1DZ,PITCH,ROL
       *L,YAW,PTIM(3),TMN(3),TMX(3),NLCOEF(2),NRCOEF(2),SCLL0(2),SCLL1(2),
       *ELCOEF(11,2),SCLR0(2),SCLR1(2),ERCOEF(11,2),NASCEF,SCLAS0,SCLAS1,
       *IELEMN,IELEMX,ASCOEF(16)
        DATA MODAY/0,31,59,90,120,151,181,212,243,273,304,334/
        DATA LUN/4/,LP/6/,ICR/5/
        DATA IT/5/,NELES/512/
        DATA NLINS/512/
C       PICK UP INPUT PARAMETERS
        NWA=5600
        MENU=1
        CALL INCOM(' ENTER PICTURE TIME (HHMMSS)',28,1,1,RMIN,MENU,3,1)
        IF(MENU.LE.0) GO TO 999
        ITIMX=RMIN(1)
        PT=FTIME(ITIMX)
        DO 5 I=1,3
5       IF(PT.EQ.PTIM(I)) GO TO 6
        WRITE(IT,710)ITIMX
710     FORMAT(1X,'NAVCOM NOT INITIALIZED FOR ',I7)
        GO TO 999
6       CONTINUE
        IPIC=I
        MENU=2
        CALL INCOM(' ENTER START LINE AND ELEMENT',29,3,1,MIN,MENU,1,1)
        IF(MENU.LE.0) GO TO 999
        LINE=MIN(1)
        IELE=MIN(2)
        MENU=1
        CALL INCOM(' ENTER ZOOM AREA(1-7)',21,5,1,MIN,MENU,1,1)
        IF(MENU.LE.0) GO TO 999
        IAREA=MIN(1)
        LRNVIS=IAREA+5+(IPIC-1)*7
        LRNIR=LRNVIS+21
        MENU=1
        CALL INCOM(' ENTER TAPE UNIT (0,1)',22,7,1,MIN,MENU,1,1)
        IF(MENU.LE.0) GO TO 999
        IMUNIT=MIN(1)
        IBDF=1
        IFVIS=.TRUE.
        IFIR=.TRUE.
C       COMPUTE UPPER LEFT COORDINATES
        IULINE=LINE
        IUELE=IELE
        IUMS=1200-(IULINE-1)/2
        ILLINE=IULINE+(NLINS-1)*IBDF
```

```
      ILMS=1200-(IDLINE-1)/2
C     ESTABLISH AREA DIRECTORY
      DO 10 I=1,256
10    IDIR(I)=0
      IDIR(1)='AT'
      IDIR(2)='S6'
      IDIR(3)=' V'
      IDIR(4)='IS'
      IDIR(5)='E.'
      IDIR(6)='H.'
      IDIR(7)='T.'
      IDIR(8)='  '
      IDIR(9)='AT'
      IDIR(10)='S6'
      IDIR(11)='  '
      IDIR(12)='E.'
      IDIR(13)='H.'
      IDIR(14)='T.'
      IDIR(15)='  '
      IDIR(16)='  '
      DO 12 I=12,1,-1
12    IF(IDAY.GT.MODAY(I))GO TO 13
13    CONTINUE
      MON=I
      IDIR(19)=IYR*100+MON
      IDIR(20)=(IDAY-MODAY(MON))*100+IFIX(PTIM(IPIC)/60.)
      IM=PTIM(IPIC)
      IS=PTIM(IPIC)*60.-IM*60.
      IS=MOD(IS,60)
      IM=MOD(IM,60)
      IDIR(21)=IM*100+IS
      IDIR(24)=1
      IDIR(25)=1
      IDIR(26)=IPIC
      IDIR(29)='EF'
      IDIR(30)='IC'
      IDIR(31)=1
      IDIR(32)=LRNVIS
      IDIR(33)=512
      IDIR(34)=513
      IDIR(37)=512
      IDIR(38)=512
      IDIR(39)=IDELE
      IDIR(40)=IDLINE
      IDIR(41)=1
      IDIR(42)=1
      IDIR(43)=1
      IDIR(44)=1
      IDIR(45)=1
      IDIR(46)=1
      IDIR(47)=1
      IDIR(48)=1
      IDIR(51)=1000*(IMUINT+1)
      IDIR(53)=1
      NWD=(NELES+1)/2
      CALL OPEN(LRNVIS,NLINS+1,NWD)
      CALL OPEN(LRNIR,NLINS+1,NWD)
      CALL LBLWRT(LRNVIS,IDIR,256)
      IDIR(32)=LRNIR
      IDIR(3)='  '
      IDIR(4)='IR'
      CALL LBLWRT(LRNIR,IDIR,256)
C     GENERATE OFFSET TABLE
      CALL GENOFF(IDELE,NELES)
      CALL ASNLUN(LUN,'MM',IMUNIT)
C     ADVANCE TO FIRST SCAN TO INPUT
```

```
      WRITE(IT,711)
  711 FORMAT(' TAPE SEARCH STARTS')
      CALL IOTPSR(LUN,+2,ISTAT)
      DO 510 I=1,1200
      CALL IOTPIN(LUN,IARRAY,NWA,LE,ISTAT)
      CALL CRKTHR(195,IARRAY,IAROW)
      INMS=IAROW(195)
      IF(INMS.LE.IUMS) GO TO 511
  510 CONTINUE
      GO TO 922
  511 CALL IOTPSR(LUN,-1,ISTAT)
      WRITE(IT,712)
  712 FORMAT(' IMAGE LOAD STARTS')
C     START MAIN LOOP TO READ DATA IN AND STORE
C     READ A SCAN
  601 CALL IOTPIN(LUN,IARRAY,NWA,LE,ISTAT)
      CALL CRKTHR(195,IARRAY,IAROW)
      INMS=IAROW(195)
      IF(INMS.LT.ILMS) GO TO 690
C     MAP SCAN NUMBER TO AREA ROW
      L1=(1200-INMS)*2+(3-2)
      L2=(1200-INMS)*2+(3-1)
      LA1=L1-IULINE
      LA2=L2-IULINE
      LR1=MOD(LA1,IBDF)
      LR2=MOD(LA2,IBDF)
      IF(LR1.NE.0 .AND. LR2.NE.0) GO TO 601
      IROW1=LA1/IBDF
      IROW2=LA2/IBDF
      ISD=MOD(INMS,2)
      ISEC1=NS*IROW1
      ISEC2=NS*IROW2
C     PICK OUT LINE SEGMENT
C     FIRST VISIBLE LINE
      IF(IFVIS.AND.LR1.EQ.0) CALL LINGRB(IARRAY(3745),ISD,IUELE,NELES,IB
     *DF,IAROW)
      IF(IFVIS.AND.LR1.EQ.0) CALL WRITE(LRNVIS,IAROW,0)
C     SECOND VISIBLE LINE
      IF(IFVIS.AND.LR2.EQ.0)CALL LINGRB(IARRAY(1945),ISD,IUELE,NELES,IBD
     *F,IAROW)
      IF(IFVIS.AND.LR2.EQ.0) CALL WRITE(LRNVIS,IAROW,0)
C     INFRARED
      IF(IFIR) CALL LINGRB(IARRAY(145),ISD,IUELE,NELES,IBDF,IAROW)
      IF(IFIR.AND.LR1.EQ.0) CALL WRITE(LRNIR,IAROW,0)
      IF(IFIR.AND.LR2.EQ.0) CALL WRITE(LRNIR,IAROW,0)
      GO TO 601
  690 CONTINUE
  922 CONTINUE
      CALL CLOSEF(LRNVIS)
      CALL CLOSEF(LRNIR)
      CALL IOTPRW(LUN)
      GO TO 999
  999 CONTINUE
      CALL REQUES(RAD50('ATSF2'))
      END
```

****** LATSF/MCIDAS ****** DAT

```
@ELT,L AF.LATSF/MCIDAS
ELT007 RLIB62 12/22-17:01:53-(0,)
000     $JOB LDATSF U3200
000     $OPTION .8,9,13,20
000     $FORTRAN
000           SUBROUTINE MAIN
000     C     PROGRAM TO READ IMAGES FROM ATS-6 TAPES.
000           LOGICAL IFVIS,IFIR,OPTION
000           DIMENSION MIN(12),IARRAY(3732),IAROW(672)
000           DATA LUN/10/,LP/6/,ICR/5/
000           DATA MIN/6HLDATSF,10*0/,NWA/3732/
000     C     PICK UP INPUT PARAMETERS
000           CALL IQ(MIN)
000     C     INPUT:  SSYYDDD  HHMMSS  AREA  LINE  ELEMENT
000           IDAY=MIN(1)
000           ITIME=MIN(2)
000           IAREA=MIN(3)
000           LINE=MIN(4)
000           IELE=MIN(5)
000           IBDF=1
000           IREEL=0
000           KEY=3H
000     C     COMPUTE DERIVED PARAMETERS
000           ISS=IDAY/100000
000           IFILE=1
000           IVSS=(ISS/2)*2
000           IRSS=IVSS+1
000           IFVIS=ISS.EQ.IVSS
000           IFIR=ISS.EQ.IRSS.OR.IAREA.LT.9
000           IRAREA=IAREA
000           IF(IFIR.AND.IFVIS) IRAREA=IAREA+8
000     C     PICK UP AREA SIZE
000           CALL HOWBIG(IAREA,NLINS,NELES)
000           IF(NELES.GT.672)    GO TO 922
000     C     COMPUTE UPPER LEFT COORDINATES
000           IULINE=LINE
000           IUELE=IELE
000           IF(OPTION(KEY,3H  C)) IULINE=LINE-(NLINS/2)*IBDF
000           IF(OPTION(KEY,3H  C)) IUELE=IELE-(NELES/2)*IBDF
000           IUMS=1200-(IULINE-1)/2
000           ILLINE=IULINE+(NLINS-1)*IBDF
000           ILMS=1200-(ILLINE-1)/2
000     C     CREATE IR AREA FOR COMBINED LOAD
000           IF(IFIR.AND.IFVIS) CALL ARASIZ(IAREA+8,NLINS,NELES)
000     C     ESTABLISH AREA DIRECTORY
000           IGE=0
000           CALL ENAREA(IAREA,IREEL,IDAY,ITIME,IULINE,IUELE,IBDF,IBDF,IGE)
000           IRREEL=IRSS*100000+MOD(IREEL,100000)
000           IF(IREEL.EQ.0) IRREEL=0
000           IRDAY=IRSS*100000+MOD(IDAY,100000)
000           IF(IFVIS.AND.IFIR) CALL ENAREA(IAREA+8,IRREEL,IRDAY,ITIME,IULINE,I
000          *UELE,IBDF,IBDF,IGE)
000     C     GENERATE OFFSET TABLE
000           CALL GENOFF(IUELE,NELES)
000           IRR=MAX0(1,MOD(IREEL,10000))
000           CALL GTAP(14,IRR,LUN)
000     C     ADVANCE TO FILE
```

```
000         IFILE=IFILE-1
000         IF(IFILE.LT.1)GO TO 502
000         GO TO 922
000   502   CONTINUE
000   C     ADVANCE TO FIRST SCAN TO INPUT
000         CALL READW(LUN,NWA,IARRAY)
000         CALL READW(LUN,NWA,IARRAY)
000         DO 510 I=1,1200
000         CALL READW(LUN,NWA,IARRAY)
000         CALL CRKTHR(195,IARRAY,IAROW)
000         INMS=IAROW(195)
000         IF(INMS.LE.IUMS) GO TO 511
000   510   CONTINUE
000         GO TO 922
000   511   CALL BSR(LUN)
000   C     START MAIN LOOP TO READ DATA IN AND STORE
000   C     READ A SCAN
000   601   CALL READW(LUN,NWA,IARRAY)
000         CALL CRKTHR(195,IARRAY,IAROW)
000         INMS=IAROW(195)
000         IF(INMS.LT.ILMS) GO TO 690
000   C     MAP SCAN NUMBER TO AREA ROW
000         L1=(1200-INMS)*2+(3-2)
000         L2=(1200-INMS)*2+(3-1)
000         LA1=L1-IULINE
000         LA2=L2-IULINE
000         LR1=MOD(LA1,IBDF)
000         LR2=MOD(LA2,IBDF)
000         IF(LR1.NE.0 .AND. LR2.NE.0) GO TO 601
000         IROW1=LA1/IBDF
000         IROW2=LA2/IBDF
000         NS=NSECL(NELES)
000         ISD=MOD(INMS,2)
000         ISEC1=NS*IROW1
000         ISEC2=NS*IROW2
000   C     PICK OUT LINE SEGMENT
000   C     FIRST VISIBLE LINE
000         IF(IFVIS.AND.LR1.EQ.0) CALL LINGRB(IARRAY(2497),ISD,IUELE,NELES,IB
000        *DF,IAROW)
000         IF(IFVIS.AND.LR1.EQ.0) CALL WRITA(IAREA,ISEC1,NS*112,IAROW)
000   C     SECOND VISIBLE LINE
000         IF(IFVIS.AND.LR2.EQ.0)CALL LINGRB(IARRAY(1297),ISD,IUELE,NELES,IBD
000        *F,IAROW)
000         IF(IFVIS.AND.LR2.EQ.0) CALL WRITA(IAREA,ISEC2,NS*112,IAROW)
000   C     INFRARED
000         IF(IFIR) CALL LINGRB(IARRAY(97),ISD,IUELE,NELES,IBDF,IAROW)
000         IF(IFIR.AND.LR1.EQ.0) CALL WRITA(IRAREA,ISEC1,NS*112,IAROW)
000         IF(IFIR.AND.LR2.EQ.0) CALL WRITA(IRAREA,ISEC2,NS*112,IAROW)
000         GO TO 601
000   690   CONTINUE
000         CALL MARKOK(IAREA)
000         CALL MARKOK(IRAREA)
000         CALL REW(LUN)
000         RETURN
000   910   CONTINUE
000   912   CONTINUE
000   913   CONTINUE
```

```
000        CALL EMESS(3HREQ,0)
000        RETURN
000   921  WRITE(LP,701)
000   701  FORMAT(1X,'E-O-F TERMINATES LOAD')
000        CALL MARKOK(IAREA)
000        CALL MARKOK(IRAREA)
000        CALL REW(LUN)
000        RETURN
000   922  CONTINUE
000        CALL EMESS(3HREQ,0)
000        RETURN
000        END
000        SUBROUTINE LINGRB(INDATA,SDIR,ELE,NELES,BDF,OUTDAT)
000        IMPLICIT INTEGER(A-Z)
000        DIMENSION INDATA(1),OUTDAT(1)
000        DIMENSION TDAT(2406),DLELE(672)
000        COMMON/OFFSET/DLELE
000        DATA WOFF/6/,NE/2406/
000        IF(BDF.NE.1) CALL EMESS(3HREQ,BDF)
000        IF(BDF.NE.1) CALL EXIT
000        IO2=ELE-1
000        CALL CRKATS(NE,INDATA,TDAT)
000        IF(SDIR.EQ.1) GO TO 100
000   C    SHIFT EVEN SCANS
000        DO 50 I=1,NELES
000        DE=I+IO2+DLELE(I)+WOFF
000        OUTDAT(I)=TDAT(DE)
000   50   CONTINUE
000        CALL PACK(NELES,OUTDAT,OUTDAT)
000        RETURN
000   100  CONTINUE
000   C    LOAD ODD SCANS
000        DO 150 I=1,NELES
000        OUTDAT(I)=TDAT(I+IO2+WOFF)
000   150  CONTINUE
000        CALL PACK(NELES,OUTDAT,OUTDAT)
000        RETURN
000        END
000        SUBROUTINE GENOFF(IUELE,NELES)
000        COMMON/NAVCOM/NAVN,INAV,IYR,IDAY,TOTLIN,DEGLIN,TOTIEL,DEGELE,PICLI
000       *N,PICELE,TMPSCL,IOYR,IODAY,TM,R1X,R1Y,R1Z,R1DX,R1DY,R1DZ,PITCH,ROL
000       *L,YAW,PTIM(3),TMN(3),TMX(3),NLCOEF(2),NRCOEF(2),SCLL0(2),SCLL1(2),
000       *ELCOEF(11,2),SCLR0(2),SCLR1(2),ERCOEF(11,2),NASCEF,SCLAS0,SCLAS1,
000       *IELEMN,IELEMX,ASCOEF(16)
000        COMMON/OFFSET/IDLELE(672)
000        DATA NOMOFF/8/
000        CALL RCOM
000        NEND=IUELE+NELES-1
000        DO 100 I=1,NELES
000        IDLELE(I)=NOMOFF
000        IELE=IUELE+I-1
000        IF(IELE.LT.IELEMN.OR.IELEMX.LT.IELE)GO TO 100
000        T=IELE*SCLAS1+SCLAS0
000        CALL CNPS(Y,T,ASCOEF,NASCEF)
000        IDLELE(I)=Y+0.5
000   100  CONTINUE
000        RETURN
```

****** LATSF/MCIDAS ******

DA

```
000170      000          ENDS
000171      000     $FILEMA
000172      000     DELETE LDATSF,GORP
000173      000     $INCLUDE IFLD
000174      000     $CATALOG
000175      000     NAME=LDATSF,5,R,W,D
000176      000     TYPE=FG
000177      000     LIB=ATSFLB,LL
000178      000     BEGIN
000179      000     $EOJ

END ELT.
@HDG,P  ******  LDATSF  ******
```

```
****** LS/MCIDAS ******

@ELT,L AF.LS/MCIDAS
ELT007 RLIB62 12/22-17:01:56-(0,)
       000          SUBROUTINE LS(X,Y,VAL,DD,DIR)
       000    C
       000    C  THIS ROUTINE PERFORMS AN ARMIJO LINE SEARCH FROM THE POINT "X"
       000    C  IN THE DIRECTION "DIR" AND RETURNS THE SELECTED POINT IN "Y"
       000    C  AND THE OBJECTIVE FUNCTION VALUE S(Y) IN "VAL". ON CALL, "DD"
       000    C  IS THE UNNORMALIZED DIRECTIONAL DERIVATIVE <GRAD(S(X)),DIR>.
       000    C  THIS LINE SEARCH ROUTINE RETURNS IN Y THE POINT X+2**(-N)*DIR,
       000    C  WHERE N IS THE LEAST NONNEGATIVE INTEGER SUCH THAT
       000    C  -S(2**(-N)*DIR) REPRESENTS AT LEAST 40% OF THE FUNCTIONAL DROP
       000    C  IN THE LINEARIZATION OF S AT X IN MOVING FROM X TO X+2**(-N)*DIR.
       000    C
       000          DIMENSION X(3),Y(3),DIR(3)
       000          DATA FAC,WALK/5.0E-1,1.0E-5/
       000          RLAM=1.0
       000          TSTVAL=.4*RLAM*ABS(DD)
       000          OLDVAL=S(X)
       000        1 IF(RLAM.LT.WALK)RETURN
       000          DO 2 I=1,3
       000        2 Y(I)=X(I)+RLAM*DIR(I)
       000          VAL=S(Y)
       000          IF(OLDVAL-VAL.GE.TSTVAL)RETURN
       000          TSTVAL=FAC*TSTVAL
       000          RLAM=FAC*RLAM
       000          GOTO 1
       000          END$

END ELT.
@HDG,P ****** MINMIZ/MCIDAS ******
```

```
******  MINMIZ/MCIDAS  ******                                                                DAT

@ELT,L AF.MINMIZ/MCIDAS
ELT007 RLIB62 12/22-17:01:56-(0,)
        000           SUBROUTINE MINMIZ(PTIN,PTOUT,GNORM,VAL,ITN)
        000     C
        000     C  THIS ROUTINE PERFORMS A MODIFIED NEWTON METHOD MINIMIZATION.  IT
        000     C  BEGINS AT THE POINT "PTIN" AND RETURNS IN "PTOUT" THE POINT
        000     C  SELECTED AS THE OPTIMAL POINT, IN "GNORM" THE NORM OF THE GRADIENT
        000     C  OF S AT "PTOUT", AND IN "VAL" THE VALUE OF THE OBJECCTIVE FUNCTION
        000     C  S AT "PTOUT".  A POINT X(K+1) IS DEEMED OPTIMAL WHEN
        000     C  ABS(S(X(K+1))-S(X(K)))<=(10**-10)*ABS(S(X(K))).
        000     C
        000           LOGICAL ALTRET
        000           DIMENSION PTIN(3),PTOUT(3)
        000           DIMENSION HESS(3,3),GRAD(3),PT(3),DIR(3)
        000           DATA CONVRG,ITERAT,EQUALO/1.0E-18,25,1.0E-30/
        000     C
        000           ITN=0
        000           DO 5 I=1,3
        000         5 PT(I)=PTIN(I)
        000           OLDVAL=S(PT)
        000           DO 50 I=1,ITERAT
        000           ITN=ITN+1
        000           CALL PRTIAL(PT,GRAD,HESS)
        000           CALL INVERT(HESS,HESS,3,ALTRET)
        000           IF(ALTRET) GO TO 16
        000           DO 10 J=1,3
        000           DIR(J)=0
        000           DO 10 K=1,3
        000        10 DIR(J)=DIR(J)-HESS(J,K)*GRAD(K)
        000           DDD=0
        000           DO 15 J=1,3
        000        15 DDD=DDD+DIR(J)*GRAD(J)
        000           IF(DDD.LT.-EQUALO)GOTO 25
        000           IF(DDD.GT.+EQUALO)GOTO 19
        000        16 DDD=0
        000           DO 17 J=1,3
        000           DDD=DDD+GRAD(J)**2
        000        17 DIR(J)=-GRAD(J)
        000           GOTO 25
        000        19 DO 20 J=1,3
        000        20 DIR(J)=-DIR(J)
        000        25 CALL LS(PT,PTOUT,VAL,DDD,DIR)
        000           IF(ABS(VAL-OLDVAL).LE.CONVRG*ABS(OLDVAL)) GO TO 60
        000           OLDVAL=VAL
        000           DO 30 J=1,3
        000        30 PT(J)=PTOUT(J)
        000        50 CONTINUE
        000        60 GNORM=0
        000           DO 65 I=1,3
        000        65 GNORM=GNORM+GRAD(I)**2
        000           GNORM=SQRT(GNORM)
        000           RETURN
        000           END$

END ELT.
@HDG,P  ******  NRMLIZ/MCIDAS  ******
```

```
******  NRMLIZ/MCIDAS  ******                                                DAT

@ELT,L AF.NRMLIZ/MCIDAS
ELT007 RLIB62 12/22-17:01:57-(0,)
        000          SUBROUTINE NRMLIZ(VX,VY,VZ,VNORM)
        000          VNORM=SQRT(VX**2+VY**2+VZ**2)
        000          VX=VX/VNORM
        000          VY=VY/VNORM
        000          VZ=VZ/VNORM
        000          RETURN
        000          END$

END ELT.
@HDG,P  ******  OFFSETFIT  ******
```

OFSTF/AOIPS

```
C     PROGRAM TO FIT POLYNOMIAL TO ALTERNATE SCAN OFFSET DATA STORED IN
C     UNIT 10 BY PROGRAM OFFSEIGEN.  INPUT DATA:
C     CARD1: FIRST-POINT-ELEMENT-COORDINATE  LAST-POINT
      LOGICAL OPTION
      DIMENSION ICORLM(2),XSCALE(10),OFSTLM(2),YSCALE(10)
      DIMENSION DATI(690),WORK(231)
      DIMENSION COEF(21),LABEL(12)
      DIMENSION MIN(10),MOUT(37)
      DIMENSION ICOOR(230),OFFSET(230),WATE(230)
      COMMON/BUFFER/WATE,OFFSET
      COMMON/BUFF1/NCORMX,ICOOR
      COMMON/NAVCOM/NAVN,INAV,IYR,IDAY,TOTLIN,DEGLIN,TOTIEL,DEGELE,PICLI
     *N,PICELE,IMPSCL,IOYR,IODAY,IM,R1X,R1Y,R1Z,R1DX,R1DY,R1DZ,PITCH,ROL
     *L,YAW,PTIM(3),TMN(3),TMX(3),NLCOEF(2),NRCOEF(2),SCLL0(2),SCLL1(2),
     *ELCOEF(11,2),SCLR0(2),SCLR1(2),ERCOEF(11,2),NASCEF,SCLAS0,SCLAS1,
     *IELEMN,IELEMX,ASCOEF(16)
      DATA ICORLM/0,2400/,OFSTLM/5.,15./
      DATA LP/6/
      DATA IT/5/
      DATA MINDEG/15/
      MENU=2
      CALL INCOM(' ENTER LEFT, RIGHT ELEMENTS FROM PRINTER LISTING.',49,
     *3,1,MIN,1,1)
      IF(MENU.LE.0) GO TO 999
C     READ LIMITS OF USEFUL DATA (ICOOR LIMITS) FROM CARDS.
      ILE=MIN(1)
      IRE=MIN(2)
C     READ IN OFFSETS AND WEIGHTS AND COORDINATESS(ELEMENT NUMBER)
      DO 10 I=1,NCORMX
      IF(ICOOR(I).EQ.ILE) ILEAE=I
      IF(ICOOR(I).EQ.IRE)IREAE=I
10    CONTINUE
      NPTS=IREAE-ILEAE+1
C     FIT LEAST SQUARES POLYNOMIAL.
      IN=0
      DO 25 I=ILEAE,IREAE
      IN=IN+1
      DATI(IN)=ICOOR(I)
      DATI(IN+NPTS)=OFFSET(I)
      DATI(IN+2*NPTS)=WATE(I)
25    CONTINUE
      CALL APCH(DATI,NPTS,MINDEG,XD,X0,WORK,IER)
      IF(IER.NE.0) GO TO 990
      ETA=1.0E-3
      EPS=1.0E-4
      IOP=-1
      CALL APFS(WORK,MINDEG,M,IOP,EPS,ETA,IER)
      IF(IER)990,28,28
28    CONTINUE
      NASCEF=M
      SCLAS0=X0
      SCLAS1=XD
      IELEMN=ILE
```

OFSTF/AOIPS

```
      IELEMX=IRE
      IX=M*(M-1)/2
      DO 29 I=0,M
      COEF(I+1)=WORK(I+1+IX)
29    ASCOEF(I+1)=COEF(I+1)
C     PRINT COEFFICIENT, DEGREE SCALING
      WRITE(IT,720)M
720   FORMAT(10X,'DEGREE=',I2)
      WRITE(IT,721)X0,XD
721   FORMAT(2X,'SCALING PARAMETERS. X0=',E15.9,' XD=',E15.9)
      DO 30 I=0,M
30    WRITE(IT,722)I,COEF(I+1)
722   FORMAT(10X,'D(',I2,')=',E15.9)
      GO TO 999
990   CONTINUE
      WRITE(IT,730)IER
999   CONTINUE
      CALL REQUES(RAD50('ATSF2'))
730   FORMAT(1X,'ERROR CODE=',I2)
      END
```

****** OFSTF/MCIDAS ****** DA

```
@ELT,L AF.OFSTF/MCIDAS
ELT007 RLIB62 12/22-17:01:58-(0,)
000      $JOB OFSTFT U3200
000      $OPTION .8,9,13,20
000      $FORTRAN
000                SUBROUTINE MAIN
000      C         PROGRAM TO FIT POLYNOMIAL TO ALTERNATE SCAN OFFSET DATA STORED IN
000      C         UNIT 10 BY PROGRAM OFFSETGEN.  INPUT DATA:
000      C         CARD1: FIRST-POINT-ELEMENT-COORDINATE  LAST-POINT
000                LOGICAL OPTION
000                DIMENSION ICORLM(2),XSCALE(10),OFSTLM(2),YSCALE(10)
000                DIMENSION DATI(690),WORK(231)
000                DIMENSION COEF(21),LABEL(12)
000                DIMENSION MIN(10),MOUT(24)
000                COMMON/OFFDAT/NCORMX,ICOOR(230),OFFSET(230),WATE(230)
000                COMMON/NAVCOM/NAVN,INAV,IYR,IDAY,TOTLIN,DEGLIN,TOTIEL,DEGELE,PICLI
000               *N,PICELE,TMPSCL,IOYR,IODAY,TM,R1X,R1Y,R1Z,R1DX,R1DY,R1DZ,PITCH,ROL
000               *L,YAW,PTIM(3),TMN(3),TMX(3),NLCOEF(2),NRCOEF(2),SCLLO(2),SCLL1(2),
000               *ELCOEF(11,2),SCLRO(2),SCLR1(2),ERCOEF(11,2),NASCEF,SCLASO,SCLAS1,
000               *IELEMN,IELEMX,ASCOEF(16)
000                DATA ICORLM/0,2400/,OFSTLM/5.,15./
000                DATA LP/6/
000                DATA MINDEG/15/
000                DATA MIN/6HGCCGCC,8*0/,LUNF/10/
000                CALL IQ(MIN)
000      C         READ LIMITS OF USEFUL DATA (ICOOR LIMITS) FROM CARDS.
000                ILE=MIN(1)
000                IRE=MIN(2)
000                JOUT=1
000                IF(OPTION(MIN(3),3H  P)) JOUT=2
000      C         READ IN OFFSETS AND WEIGHTS AND COORDINATESS(ELEMENT NUMBER)
000                CALL DYNASG('OFFSTD',LUNF)
000                CALL OPN(LUNF)
000                CALL READW(LUNF,1151,NCORMX)
000                DO 10 I=1,NCORMX
000                IF(ICOOR(I).EQ.ILE) ILEAE=I
000                IF(ICOOR(I).EQ.IRE)IREAE=I
000      10        CONTINUE
000                NPTS=IREAE-ILEAE+1
000      C         FIT LEAST SQUARES POLYNOMIAL.
000                IN=0
000                DO 25 I=ILEAE,IREAE
000                IN=IN+1
000                DATI(IN)=ICOOR(I)
000                DATI(IN+NPTS)=OFFSET(I)
000                DATI(IN+2*NPTS)=WATE(I)
000      25        CONTINUE
000                CALL APCH(DATI,NPTS,MINDEG,XD,XO,WORK,IER)
000                IF(IER.NE.0) GO TO 990
000                ETA=1.0E-3
000                EPS=1.0E-4
000                IOP=-1
000                CALL APFS(WORK,MINDEG,M,IOP,EPS,ETA,IER)
000                IF(IER)990,28,28
000      28        CONTINUE
000                CALL RCOM
000                NASCEF=M
```

```
****** OFSTF/MCIDAS ******

000            SCLAS0=X0
000            SCLAS1=XD
000            IELEMN=ILE
000            IELEMX=IRE
000            IX=M*(M-1)/2
000            DO 29 I=0,M
000            COEF(I+1)=WORK(I+1+IX)
000      29    ASCOEF(I+1)=COEF(I+1)
000      C     STORE DATA
000            CALL WCOM
000      C     PRINT COEFFICIENT, DEGREE SCALING
000            ENCODE(132,720,MOUT)M
000            CALL TP(JOUT,MOUT)
000      720   FORMAT(10X,'DEGREE=',I2)
000            ENCODE(132,721,MOUT)X0,XD
000            CALL TP(JOUT,MOUT)
000      721   FORMAT(2X,'SCALING PARAMETERS. X0=',E15.9,' XD=',E15.9)
000            DO 30 I=0,M
000            ENCODE(132,722,MOUT)I,COEF(I+1)
000      30    CALL TP(JOUT,MOUT)
000      722   FORMAT(10X,'D(',I2,')=',E15.9)
000            RETURN
000      990   CONTINUE
000            ENCODE(132,720,MOUT)IER
000            CALL TQ(MOUT)
000            RETURN
000      730   FORMAT(1X,'ERROR CODE=',I2)
000            END$
000      $FILEMA
000      DELETE GCCGCC,GORP
000      $INCLUDE ATSSSP
000      $CATALOG
000      NAME=GCCGCC,5,R,W,D
000      TYPE=FG
000      LIB=ATSFLB,LL
000      BEGIN
000      $EOJ

END ELT.
@HDG,P  ****** OFSTG/MCIDAS  ******
```

OFSTG/AOIPS

```
C       PROGRAM TO GENERATE TABLE
C       OF ALTERNATE SCAN OFFSETS. CORRELATIONS DONE ON IR.
C       ASSIGN TAPE TO UNIT TEN. OUTPUT IS TO PRINTER AND GLOBAL COMMON
        LOGICAL MISSNG,ISLAST
        DIMENSION IODATA(5600),IEDATA(5600)
        EQUIVALENCE(IODATA,IEDATA)
        DIMENSION MIN(10),MOUT(37)
        DIMENSION IOBRT(2598),IEBRT(2598)
        DIMENSION WATE(230),OFFSET(230),ICOOR(230),FLPARY(9)
        COMMON/BUFFER/WATE,OFFSET
        COMMON/BUFF1/NCORMX,ICOOR
        DATA ISTATG/0/,NLAG/9/,NCOR/220/,LPRT/6/
        DATA NW1/147/,NW2/1949/,NW3/5598/
        DATA LUN/10/,MISSNG/.FALSE./
        DATA LP/6/,IT/5/
        DATA IFIRST/800/,INISCN/ 8/,LSTSCN/400/
        DATA IELEST/100/,NISIZ/15/,NOMOFF/ +8/,IELINT/10/
        DATA NBP/2598/,IDOFF/198/
        DATA IDSTRT/67/
C       INITIALIZE SCAN SEARCH
        IFIRST=((IFIRST+1)/2)*2
        INTSCN=(INISCN/2)*2
        LSTSCN=((LSTSCN+1)/2)*2
        IESCAN=IFIRST+INTSCN
        NCORMX=0
        DO 10 I=1,230
        WATE(I)=0.
        OFFSET(I)=0.0
        ICOOR(I)=0
10      CONTINUE
        ENCODE(74,702,MOUT)
702     FORMAT(1X,'MOUNT TAPE. ENTER DRIVE NUMBER')
        MENU=1
        CALL INCOM(MOUT,36,1,1,MIN,MENU,1,1)
        IMUNIT=MIN(1)
        IF(MENU.LT.0) GO TO 999
        IF(MENU.EQ.0)IMUNIT=0
        CALL ASNLUN(LUN,'MM',IMUNIT)
        CALL IOTPRW(LUN)
C       SKIP HEADER RECORDS
        CALL IOTPSR(LUN,+2,ISTAT)
100     IF(MISSNG) IESCAN=IESCAN-2
        IF(.NOT.MISSNG) IESCAN=IESCAN-INTSCN
        IOSCAN=IESCAN-1
        ISLAST=IESCAN.LE.LSTSCN
200     CALL IOTPIN(LUN,IEDATA,NW1,LE,ISTAT)
        CALL CRKIHR(195,IEDATA,IEBRT)
        JESCAN=IEBRT(195)
        MISSNG=JESCAN.LT.IESCAN
        IF(MISSNG)WRITE(IT,701)IESCAN
701     FORMAT(1X,'SCAN NUMBER',I5,' NOT FOUND.')
        IF(MISSNG) GO TO 100
        IF(JESCAN.GT.IESCAN) GO TO 200
```

OFSTG/AOIPS

```
C       EVEN SCAN FOUND. NOW CHECK FOR ODD SCAN.
        CALL IOTPIN(LUN,IODATA,NW1,LE,ISTAT)
        CALL CRKTHR(195,IODATA,IOBRT)
        JOSCAN=IOBRT(195)
        MISSNG=JOSCAN.NE.IOSCAN
        IF(MISSNG)WRITE(IT,701)IOSCAN
        IF(MISSNG) GO TO 100
C       BACK UP TO READ IN DATA.
        CALL IOTPSR(LUN,-2,ISTAT)
C       INPUT PAIR OF SCANS.
        CALL IOTPIN(LUN,IEDATA,NW2,LE,ISTAT)
        CALL CRKTHR(195,IEDATA,IEBRT)
        JESCAN=IEBRT(195)
        CALL CRKAIS(NBP,IEDATA,IEBRT)
        CALL IOTPIN(LUN,IODATA,NW2,LE,ISTAT)
        CALL CRKTHR(195,IODATA,IOBRT)
        JOSCAN=IOBRT(195)
        CALL CRKAIS(NBP,IODATA,IOBRT)
        IF(JESCAN.NE.IESCAN.OR.JOSCAN.NE.IOSCAN) GO TO 990
C
C       DO CORRELATIONS.
C
        DO 600 KCOR=1,NCOR
        IELE=IELEST+(KCOR-1)*IELINT
        ICOOR(KCOR)=IELE
        MAXLAG=(NLAG-1)/2
        MINLAG=-MAXLAG
        DO 500 KLAG=MINLAG,MAXLAG
        MAXBRT=0
        MINBRT=512
        NTSIZ=(NTSIZ-1)/2*2+1
        FLP=0
        DO 400 KVAL=1,NTSIZ
        ICT=IELE+KVAL-1-NTSIZ/2
        ICS=IELE+KVAL-1+KLAG+NOMOFF-NTSIZ/2
        ICTX=ICT+IDOFF
        ICSX=ICS+IDOFF
        IPIXT=IOBRT(ICTX)
        IPIXS=IEBRT(ICSX)
        MAXBRT=MAX0(MAXBRT,IPIXT)
        MINBRT=MIN0(MINBRT,IPIXT)
        FLP=FLP+(IPIXT-IPIXS)**2
400     CONTINUE
C       STORE LP(2) MEASURE VALUE
        FLPARY(KLAG-MINLAG+1)=FLP
500     CONTINUE
C       SEARCH TABLE OF VALUES OF LP(2) MEASURE FOR MIN.
        IOFF=NOMOFF+MINLAG
        FLPMIN=FLPARY(1)
        DO 510 K=2,NLAG
        IF(FLPMIN.GT.FLPARY(K)) IOFF=NOMOFF+(K-1)+MINLAG
        IF(FLPMIN.GT.FLPARY(K)) FLPMIN=FLPARY(K)
510     CONTINUE
C       WEIGHT OFFSETS BY BRIGHTNESS RANGE.
        WT=FLOAT(MAXBRT-MINBRT)/512.
        WATE(KCOR)=WATE(KCOR)+WT
        OFFSET(KCOR)=OFFSET(KCOR)+IOFF*WT
600     CONTINUE
        IF(.NOT.ISLAST) GO TO 100
C       COMPUTE AVARAGE OFFSETS.
        DO 610 KCOR=1,NCOR
        IF(WATE(KCOR).LT.1.0E-3) GO TO 610
        OFFSET(KCOR)=OFFSET(KCOR)/WATE(KCOR)
610     CONTINUE
C       DUMP TABLE VALUES.
        NICOL=(NCOR-1)/5+1
```

OFSTG/AOIPS

```
      JOUT=2
      DO 650 IWW=1,NICOL
      WRITE(LP,710)(ICOOR(IW),OFFSET(IW),WATE(IW),IW=IWW,NCOR,NICOL)
650   CONTINUE
710   FORMAT ((5(5X,I4,1X,F8.3,F8.4)))
      NCORMX=NCOR
      GO TO 999
990   CONTINUE
      WRITE(5,715)
715   FORMAT(2X,'OFFSETGEN ERROR RETURN.')
999   CONTINUE
      CALL IOIPRW(LUN)
      CALL REQUES(RAD50('OFSTF2'))
      END
```

****** OFSTG/MCIDAS ****** DATE

```
@ELT,L AF.OFSTG/MCIDAS
ELT007 RLIB62 12/22-17:01:59-(0,)
000     $JOB ATS6 U3200
000     $OPTION.8,9,20
000     $FORTRAN
000             SUBROUTINE MAIN
000     C       OF ALTERNATE SCAN OFFSETS. CORRELATIONS DONE ON IR.
000     C       ASSIGN TAPE TO UNIT 10. OUTPUT IS TO PRINTER AND UNIT 20.
000             LOGICAL MISSNG,ISLAST
000             DIMENSION IODATA(1299),IEDATA(1299)
000             DIMENSION MIN(10),MOUT(24)
000             DIMENSION IOBRT(2598),IEBRT(2598)
000             DIMENSION WATE(230),OFFSET(230),ICOOR(230),FLPARY(9)
000             COMMON/OFFDAT/NCORMX,ICOOR,OFFSET,WATE
000             DATA ISTATG/0/,NLAG/9/,NCOR/220/,LPRT/6/
000             DATA MIN/6HGCCGCC,8*0/
000             DATA NW1/98/,NW2/1299/,NW3/3732/
000             DATA LUN/10/,MISSNG/.FALSE./
000             DATA LUNF/20/
000             DATA IFIRST/800/,INTSCN/ 8/,LSTSCN/400/
000             DATA IELEST/100/,NTSIZ/15/,NOMOFF/ +8/,IELINT/10/
000             DATA NBP/2598/,IDOFF/198/
000             DATA IDSTRT/67/
000             CALL IQ(MIN)
000             IREEL=0
000     C       INITIALIZE SCAN SEARCH
000             IFIRST=((IFIRST+1)/2)*2
000             INTSCN=(INTSCN/2)*2
000             LSTSCN=((LSTSCN+1)/2)*2
000             IESCAN=IFIRST+INTSCN
000             CALL GTAP(14,IREEL,LUN)
000             CALL REW(LUN)
000     C       SKIP HEADER RECORDS
000             CALL READW(LUN,NW1,IEDATA)
000             CALL READW(LUN,NW1,IEDATA)
000     100     IF(MISSNG) IESCAN=IESCAN-2
000             IF(.NOT.MISSNG) IESCAN=IESCAN-INTSCN
000             IOSCAN=IESCAN-1
000             ISLAST=IESCAN.LE.LSTSCN
000     200     CALL READW(LUN,NW1,IEDATA)
000             CALL CRKTHR(195,IEDATA,IEBRT)
000             JESCAN=IEBRT(195)
000             MISSNG=JESCAN.LT.IESCAN
000             IF(MISSNG) ENCODE(132,701,MOUT)IESCAN
000             IF(MISSNG) CALL TQ(MOUT)
000     701     FORMAT(1X,'SCAN NUMBER',I5,' NOT FOUND.')
000             IF(MISSNG) GO TO 100
000             IF(JESCAN.GT.IESCAN) GO TO 200
000     C       EVEN SCAN FOUND. NOW CHECK FOR ODD SCAN.
000             CALL READW(LUN,NW1,IODATA)
000             CALL CRKTHR(195,IODATA,IOBRT)
000             JOSCAN=IOBRT(195)
000             MISSNG=JOSCAN.NE.IOSCAN
000             IF(MISSNG) ENCODE(132,701,MOUT) IOSCAN
000             IF(MISSNG) CALL TQ(MOUT)
000             IF(MISSNG) GO TO 100
000     C       BACK UP TO READ IN DATA.
```

```
000         CALL BSR(LUN)
000         CALL BSR(LUN)
000   C     INPUT PAIR OF SCANS.
000         CALL READW(LUN,NW2,IEDATA)
000         CALL READW(LUN,NW2,IODATA)
000         CALL CRKTHR(195,IEDATA,IEBRT)
000         JESCAN=IEBRT(195)
000         CALL CRKATS(NBP,IEDATA,IEBRT)
000         CALL CRKTHR(195,IODATA,IOBRT)
000         JOSCAN=IOBRT(195)
000         CALL CRKATS(NBP,IODATA,IOBRT)
000         IF(JESCAN.NE.IESCAN.OR.JOSCAN.NE.IOSCAN) GO TO 990
000   C
000   C     DO CORRELATIONS.
000   C
000         DO 600 KCOR=1,NCOR
000         IELE=IELEST+(KCOR-1)*IELINT
000         ICOOR(KCOR)=IELE
000         MAXLAG=(NLAG-1)/2
000         MINLAG=-MAXLAG
000         DO 500 KLAG=MINLAG,MAXLAG
000         MAXBRT=0
000         MINBRT=512
000         NTSIZ=(NTSIZ-1)/2*2+1
000         FLP=0
000         DO 400 KVAL=1,NTSIZ
000         ICT=IELE+KVAL-1-NTSIZ/2
000         ICS=IELE+KVAL-1+KLAG+NOMOFF-NTSIZ/2
000         ICTX=ICT+IDOFF
000         ICSX=ICS+IDOFF
000         IPIXT=IOBRT(ICTX)
000         IPIXS=IEBRT(ICSX)
000         MAXBRT=MAX0(MAXBRT,IPIXT)
000         MINBRT=MIN0(MINBRT,IPIXT)
000         FLP=FLP+(IPIXT-IPIXS)**2
000   400   CONTINUE
000   C     STORE LP(2) MEASURE VALUE
000         FLPARY(KLAG-MINLAG+1)=FLP
000   500   CONTINUE
000   C     SEARCH TABLE OF VALUES OF LP(2) MEASURE FOR MIN.
000         IOFF=NOMOFF+MINLAG
000         FLPMIN=FLPARY(1)
000         DO 510 K=2,NLAG
000         IF(FLPMIN.GT.FLPARY(K)) IOFF=NOMOFF+(K-1)+MINLAG
000         IF(FLPMIN.GT.FLPARY(K)) FLPMIN=FLPARY(K)
000   510   CONTINUE
000   C     WEIGHT OFFSETS BY BRIGHTNESS RANGE.
000         WT=FLOAT(MAXBRT-MINBRT)/512.
000         WATE(KCOR)=WATE(KCOR)+WT
000         OFFSET(KCOR)=OFFSET(KCOR)+IOFF*WT
000   600   CONTINUE
000         IF(.NOT.ISLAST) GO TO 100
000   C     COMPUTE AVARAGE OFFSETS.
000         DO 610 KCOR=1,NCOR
000         IF(WATE(KCOR).LT.1.0E-3) GO TO 610
000         OFFSET(KCOR)=OFFSET(KCOR)/WATE(KCOR)
000   610   CONTINUE
```

```
******  OFSTG/MCIDAS  ******                                              DAT

000   C       DUMP TABLE VALUES.
000           NICOL=(NCOR-1)/5+1
000           JOUT=2
000           DO 650 IWW=1,NICOL
000           ENCODE(132,710,MOUT)(ICOOR(IW),OFFSET(IW),WATE(IW),IW=IWW,NCOR,
000          *NICOL)
000           CALL TP(JOUT,MOUT)
000   650     CONTINUE
000   710     FORMAT ((5(5X,I4,1X,F8.3,F8.4)))
000           NCORMX=NCOR
000           CALL DYNASG('OFFSTD',LUNF)
000           CALL OPN(LUNF)
000           CALL WRITW(LUNF,1151,NCORMX)
000           RETURN
000   990     CONTINUE
000           CALL TQ(72H OFFSETGEN ERROR RETURN.                              )
000          *
000           RETURN
000           END$
000   $FILEMA
000   DELETE GCCGCC,GORP
000   $INCLUDE IFLD
000   $CATALOG
000   NAME=GCCGCC,5,R,W,D
000   TYPE=FG
000   LIB=ATSFLB,LL
000   BEGIN
000   $EOJ

END ELT.
@HDG,P  ******  OFSTVL  ******
```

****** ORBIT/MCIDAS ****** DA

```
@ELT,L AF.ORBIT/MCIDAS
ELT007 RLIB62 12/22-17:02:01-(0,)
000            SUBROUTINE ORBIT(X,Y,Z,T)
000            REAL KE,MU,MMINMO
000            COMMON/NAVCOM/NAVN,INAV,IYR,IDAY,TOTLIN,DEGLIN,TOTIEL,DEGELE,PICLI
000           *N,PICELE,TMPSCL,IOYR,IODAY,TM,R1X,R1Y,R1Z,R1DX,R1DY,R1DZ,PITCH,ROL
000           *L,YAW,PTIM(3),TMN(3),TMX(3),NLCOEF(2),NRCOEF(2),SCLLO(2),SCLL1(2),
000           *ELCOEF(11,2),SCLRO(2),SCLR1(2),ERCOEF(11,2),NASCEF,SCLASO,SCLAS1,
000           *IELEMN,IELEMX,ASCOEF(16)
000            DATA RE,KE,MU,EPSILN/6378.15,0.07436574,1.0,1.0E-7/
000            SQRTMU=SQRT(MU)
000            RO=SQRT(R1X**2+R1Y**2+R1Z**2)
000            DO=(R1X*R1DX+R1Y*R1DY+R1Z*R1DZ)/SQRTMU
000            VOSQMU=(R1DX**2+R1DY**2+R1DZ**2)/SQRTMU
000            A=RO/(2.-RO*VOSQMU)
000            CE=(A-RO)/A
000            SE=DO/SQRT(A)
000            MMINMO=KE*(T-TM)*SQRTMU/A**1.5
000            GL=.5*MMINMO
000          5 SINGL=SIN(GL)
000            SNCSGL=SINGL*COS(GL)
000            XNUM=GL+SE*SINGL**2-CE*SNCSGL-.5*MMINMO
000            DENOM=1+2*SE*SNCSGL-CE*(1-2*SINGL**2)
000            G=(GL*DENOM-XNUM)/DENOM
000            IF(ABS(GL-G).LT.EPSILN)GOTO10
000            GL=G
000            GOTO 5
000         10 ECANOM=2*G
000            C=A*(1-COS(ECANOM))
000            S=SQRT(A)*SIN(ECANOM)
000            F=(RO-C)/RO
000            G=1/SQRTMU*(RO*S+DO*C)
000            X=(F*R1X+G*R1DX)*RE
000            Y=(F*R1Y+G*R1DY)*RE
000            Z=(F*R1Z+G*R1DZ)*RE
000            RETURN
000            END$

END ELT.
@HDG,P ****** PACK ******
```

****** PFTOTC/MCIDAS ****** DAT

```
@ELT,L AF.PFTOTC/MCIDAS
ELT007 RLIB62 12/22-17:02:02-(0,)
000             SUBROUTINE PFTOTC(XLIN,XELE,X,Y,Z,IDIR,INIT)
000             COMMON/NAVCOM/NAVN,INAV,IYR,IDAY,TOTLIN,DEGLIN,TOTIEL,DEGELE,PICLI
000            *N,PICELE,TMPSCL,IOYR,IODAY,TM,R1X,R1Y,R1Z,R1DX,R1DY,R1DZ,PITCH,ROL
000            *L,YAW,PTIM(3),TMN(3),TMX(3),NLCOEF(2),NRCOEF(2),SCLL0(2),SCLL1(2),
000            *ELCOEF(11,2),SCLR0(2),SCLR1(2),ERCOEF(11,2),NASCEF,SCLAS0,SCLAS1,
000            *IELEMN,IELEMX,ASCOEF(16)
000     C       IF IDIR = 1, X,Y,Z TO LIN, ELE
000     C       IF IDIR = 2, LIN, ELE TO X,Y,Z
000             DATA PI,RE,GRACON/3.14159265,6378.15,.07436574/
000             IF(INIT.EQ.2)GO TO 1
000             INIT = 2
000             RDPDG = PI/180.
000             RADLIN = RDPDG*DEGLIN/TOTLIN
000             RADELE = RDPDG*DEGELE/TOTIEL
000     1       IF(IDIR.EQ.2)GO TO 10
000             ELEANG = ATAN(X/Z)
000             XLNANG = ASIN(Y)
000             XELE=PICELE+ELEANG/RADELE
000             XLIN = PICLIN+XLNANG/RADLIN
000             RETURN
000     10      ELEANG=(PICELE-XELE)*RADELE
000             XLNANG = (PICLIN-XLIN)*RADLIN
000             X=-COS(XLNANG)*SIN(ELEANG)
000             Y=-SIN(XLNANG)
000             Z=COS(XLNANG)*COS(ELEANG)
000             RETURN
000             END$

END ELT.
@HDG,P  ******  PRTIAL/MCIDAS  ******
```

****** PRTIAL/MCIDAS ****** DA

```
@ELT,L AF.PRTIAL/MCIDAS
ELT007 RLIB62 12/22-17:02:03-(0,)
000               SUBROUTINE PRTIAL(PT,GRAD,HESS)
000         C
000         C     THIS ROUTINE RETURNS IN "GRAD" THE GRADIENT OF S AT "PT" AND IN
000         C     "HESS" THE HESSIAN OF S AT "PT".
000         C
000               DIMENSION X(3,150),Y(3,150),TIME(150)
000               COMMON/MINCOM/X,Y,TIME,NP
000               DIMENSIONPT(3),GRAD(3),HESS(3,3),H(3,3),HA(3,3),HB(3,3),HC(3,3),
000              *HAA(3,3),HAB(3,3),HAC(3,3),HBB(3,3),HBC(3,3),HCC(3,3)
000               A=PT(1)
000               B=PT(2)
000               C=PT(3)
000               PSPA=0
000               PSPB=0
000               PSPC=0
000               PSPASQ=0
000               PSPAPB=0
000               PSPAPC=0
000               PSPBSQ=0
000               PSPBPC=0
000               PSPCSQ=0
000               CALL UNIT(H)
000               CALL ROTATE(H,C,3,1)
000               CALL ROTATE(H,B,1,1)
000               CALL ROTATE(H,A,2,1)
000               CALL UNIT(HA)
000               CALL ROTATE(HA,C,3,1)
000               CALL ROTATE(HA,B,1,1)
000               CALL ROTATE(HA,A,2,2)
000               CALL UNIT(HB)
000               CALL ROTATE(HB,C,3,1)
000               CALL ROTATE(HB,B,1,2)
000               CALL ROTATE(HB,A,2,1)
000               CALL UNIT(HC)
000               CALL ROTATE(HC,C,3,2)
000               CALL ROTATE(HC,B,1,1)
000               CALL ROTATE(HC,A,2,1)
000               CALL UNIT(HAA)
000               CALL ROTATE(HAA,C,3,1)
000               CALL ROTATE(HAA,B,1,1)
000               CALL ROTATE(HAA,A,2,3)
000               CALL UNIT(HAB)
000               CALL ROTATE(HAB,C,3,1)
000               CALL ROTATE(HAB,B,1,2)
000               CALL ROTATE(HAB,A,2,2)
000               CALL UNIT(HAC)
000               CALL ROTATE(HAC,C,3,2)
000               CALL ROTATE(HAC,B,1,1)
000               CALL ROTATE(HAC,A,2,2)
000               CALL UNIT(HBB)
000               CALL ROTATE(HBB,C,3,1)
000               CALL ROTATE(HBB,B,1,3)
000               CALL ROTATE(HBB,A,2,1)
000               CALL UNIT(HBC)
000               CALL ROTATE(HBC,C,3,2)
```

****** PRTIAL/MCIDAS ****** DAT

```
000        CALL ROTATE(HBC,B,1,2)
000        CALL ROTATE(HBC,A,2,1)
000        CALL UNIT(HCC)
000        CALL ROTATE(HCC,C,3,3)
000        CALL ROTATE(HCC,B,1,1)
000        CALL ROTATE(HCC,A,2,1)
000        DO10J=1,NP
000        DO10I=1,3
000        T=Y(I,J)-H(I,1)*X(1,J)-H(I,2)*X(2,J)-H(I,3)*X(3,J)
000        TA=HA(I,1)*X(1,J)+HA(I,2)*X(2,J)+HA(I,3)*X(3,J)
000        TB=HB(I,1)*X(1,J)+HB(I,2)*X(2,J)+HB(I,3)*X(3,J)
000        TC=HC(I,1)*X(1,J)+HC(I,2)*X(2,J)+HC(I,3)*X(3,J)
000        TAA=HAA(I,1)*X(1,J)+HAA(I,2)*X(2,J)+HAA(I,3)*X(3,J)
000        TAB=HAB(I,1)*X(1,J)+HAB(I,2)*X(2,J)+HAB(I,3)*X(3,J)
000        TAC=HAC(I,1)*X(1,J)+HAC(I,2)*X(2,J)+HAC(I,3)*X(3,J)
000        TBB=HBB(I,1)*X(1,J)+HBB(I,2)*X(2,J)+HBB(I,3)*X(3,J)
000        TBC=HBC(I,1)*X(1,J)+HBC(I,2)*X(2,J)+HBC(I,3)*X(3,J)
000        TCC=HCC(I,1)*X(1,J)+HCC(I,2)*X(2,J)+HCC(I,3)*X(3,J)
000        PSPA=PSPA-T*TA
000        PSPB=PSPB-T*TB
000        PSPC=PSPC-T*TC
000        PSPASQ=PSPASQ-T*TAA+TA**2
000        PSPAPB=PSPAPB-T*TAB+TA*TB
000        PSPAPC=PSPAPC-T*TAC+TA*TC
000        PSPBSQ=PSPBSQ-T*TBB+TB**2
000        PSPBPC=PSPBPC-T*TBC+TB*TC
000     10 PSPCSQ=PSPCSQ-T*TCC+TC**2
000        GRAD(1)=2*PSPA
000        GRAD(2)=2*PSPB
000        GRAD(3)=2*PSPC
000        HESS(1,1)=2*PSPASQ
000        HESS(1,2)=2*PSPAPB
000        HESS(2,1)=2*PSPAPB
000        HESS(1,3)=2*PSPAPC
000        HESS(3,1)=2*PSPAPC
000        HESS(2,2)=2*PSPBSQ
000        HESS(2,3)=2*PSPBPC
000        HESS(3,2)=2*PSPBPC
000        HESS(3,3)=2*PSPCSQ
000        RETURN
000        END$
```

END ELT.

@HDG,P ****** ROTATE/MCIDAS ******

```
****** ROTATE/MCIDAS ******

@ELT,L AF.ROTATE/MCIDAS
ELT007 RLIB62 12/22-17:02:03-(0,)
000          SUBROUTINE ROTATE(A,R,IR,IDERIV)
000     C
000     C THIS ROUTINE RETURNS IN "A" THE PRODUCT OF THE INPUT MATRIX
000     C "A" AND A MATRIX RM, WHERE, IF "IDERIV"=1, RM REPRESENTS A
000     C ROTATION THROUGH AN ANGLE "R" (IN RADIANS) ABOUT THE AXIS "IR".  IF
000     C IDERIV=2, THE FIRST DERIVATIVE OF RM IS OPERATED ON A, AND IF
000     C IDERIV=3, THE SECOND DERIVATIVE OF RM IS USED.
000     C
000          DIMENSION A(3,3),INDX1(3),INDX2(3)
000          DATA INDX1,INDX2/2,1,1,3,3,2/
000          IR1=INDX1(IR)
000          IR2=INDX2(IR)
000          CR=DCOS(R)
000          SR=DSIN(R)
000          IF(IDERIV.NE.1)GO TO 2
000          DO 1 J=1,3
000          T1=A(IR1,J)
000          T2=A(IR2,J)
000          A(IR1,J)=CR*T1+SR*T2
000          A(IR2,J)=-SR*T1+CR*T2
000        1 CONTINUE
000          RETURN
000        2 IF(IDERIV.NE.2)GO TO 4
000          DO 3 J=1,3
000          A(IR,J)=0.0
000          T1=A(IR1,J)
000          T2=A(IR2,J)
000          A(IR1,J)=-SR*T1+CR*T2
000          A(IR2,J)=-CR*T1-SR*T2
000        3 CONTINUE
000          RETURN
000        4 CONTINUE
000          DO 5 J=1,3
000          A(IR,J)=0.0
000          T1=A(IR1,J)
000          T2=A(IR2,J)
000          A(IR1,J)=-CR*T1-SR*T2
000          A(IR2,J)=SR*T1-CR*T2
000        5 CONTINUE
000          RETURN
000          END$

END ELT.
@HDG,P ****** S/MCIDAS ******
```

```
****** S/MCIDAS ******

@ELT,L AF.S/MCIDAS
ELT007 RLIB62 12/22-17:02:04-(0,)
      000          FUNCTION S(PT)
      000    C
      000    C   THIS FUNCTION RETURNS AS ITS VALUE THE VALUE OF THE OBJECTIVE
      000    C   FUNCTION S AT THE POINT "PT".
      000    C
      000          DIMENSION X(3,150),Y(3,150),TIME(150)
      000          COMMON/MINCOM/X,Y,TIME,NP
      000          DIMENSION PT(3),H(3,3)
      000          SRES=0
      000          CALL UNIT(H)
      000          CALL ROTATE(H,PT(3),3,1)
      000          CALL ROTATE(H,PT(2),1,1)
      000          CALL ROTATE(H,PT(1),2,1)
      000          DO 10 J=1,NP
      000.         DO 10 I=1,3
      000       10 SRES=SRES+(Y(I,J)-H(I,1)*X(1,J)-H(I,2)*X(2,J)-H(I,3)*X(3,J))**2
      000          S=SRES
      000          RETURN
      000          END$

END ELT.
@HDG,P  ******  SATEAR/MCIDAS  ******
```

```
 ******  SATEAR/MCIDAS  ******

@ELT,L AF.SATEAR/MCIDAS
ELT007 RLIB62 12/22-17:02:05-(0,)
000                SUBROUTINE SATEAR(PICTIM,XLIN,XELE,XLAT,XLON,ITYPE,INAV,BETAIN,BET
000               *DOT,ATFRAC)
000                PTIME=PICTIM
000                GO TO (1,2,3,4,5),ITYPE
000       1        CALL SE(PTIME,XLIN,XELE,XLAT,XLON)
000                RETURN
000       2        CALL ES(PTIME,XLAT,XLON,XLIN,XELE)
000                RETURN
000       3        CONTINUE
000                RETURN
000       4        CONTINUE
000                RETURN
000       5        CONTINUE
000                RETURN
000                END

END ELT.
@HDG,P  ******  SE/MCIDAS  ******
```

****** SE/MCIDAS ******

```
@ELT,L AF.SE/MCIDAS
ELT007 RLIB62 12/22-17:02:06-(0,)
000            SUBROUTINE SE(PTIME,XLIN,XELE,ALAT,ALON)
000            COMMON/NAVCOM/NAVN,INAV,IYR,IDAY,TOTLIN,DEGLIN,TOTIEL,DEGELE,PICLI
000           *N,PICELE,TMPSCL,IOYR,IODAY,TM,R1X,R1Y,R1Z,R1DX,R1DY,R1DZ,PITCH,ROL
000           *L,YAW,PTIM(3),TMN(3),TMX(3),NLCOEF(2),NRCOEF(2),SCLLO(2),SCLL1(2),
000           *ELCOEF(11,2),SCLRO(2),SCLR1(2),ERCOEF(11,2),NASCEF,SCLASO,SCLAS1,
000           *IELEMN,IELEMX,ASCOEF(16)
000            DATA INIT/0/
000            CALL EDGCOR(PTIME,XLIN,DELLIN,DELELE)
000            ALIN=XLIN-DELLIN
000            AELE=XELE-DELELE
000            ISCAN=1200-(IFIX(XLIN-1))/2
000            TIME=PTIME+ISCAN*TMPSCL
000            IDIR=2
000            CALL PFTOTC(ALIN,AELE,X1,X2,X3,IDIR,INIT)
000            CALL BCTOPF(X1,X2,X3,IDIR)
000      C     BC SAME AS LV HERE.
000            CALL STTOLV(X1,X2,X3,IDIR,TIME)
000            CALL ERTOST(X1ER,X2ER,X3ER,X1,X2,X3,IDIR,TIME)
000            CALL ERTOER(ALAT,ALON,X1ER,X2ER,X3ER,IDIR)
000            RETURN
000            END
END ELT.
@HDG,P  ******  STTOLV/MCIDAS  ******
```

DA

DA

****** STTOLV/MCIDAS ******

```
@ELT,L AF.STTOLV/MCIDAS
ELT007 RLIB62 12/22-17:02:08-(0,)
000          SUBROUTINE STTOLV (X,Y,Z,IDIR,TIME)
000    C     IF IDIR=1, POINTING VECTOR (X,Y,Z) IS TRANSFORMED
000    C     FROM SAT INERTIAL TO LOCAL VERTICAL FRAME.
000    C     IF IDIR=2, POINTING VECTOR (X,Y,Z) IS TRANSFORMED FROM
000    C     LOCAL VERTICAL TO SAT INERTIAL FRAME.
000          CALL ORBIT(XS,YS,ZS,TIME)
000          CALL NRMLIZ(XS,YS,ZS,XNORM)
000          X1=X
000          Y1=Y
000          Z1=Z
000          D=SQRT(XS**2+YS**2)
000          IF (IDIR.EQ.2) GO TO 10
000          X=(-YS*X1+XS*Y1)/D
000          Y=(XS*ZS*X1+YS*ZS*Y1-Z1*D**2)/D
000          Z=-(XS*X1+YS*Y1+ZS*Z1)
000          RETURN
000    10    X=-YS*X1/D+XS*ZS*Y1/D-XS*Z1
000          Y=XS*X1/D+YS*ZS*Y1/D-YS*Z1
000          Z=-D*Y1-ZS*Z1
000          RETURN
000          END$

END ELT.
@HDG,P  ****** UNIT/MCIDAS ******
```

E-53

****** UNIT/MCIDAS ****** DAT

```
@ELT,L AF.UNIT/MCIDAS
ELT007 RLIB62 12/22-17:02:09-(0,)
000          SUBROUTINE UNIT(A)
000    C  THIS ROUTINE RETURNS IN "A" A 3 X 3 IDENTITY MATRIX.
000    C
000          DIMENSION A(9),B(9)
000          DATA B/1.0,0.0,0.0,
000         *        0.0,1.0,0.0,
000         *        0.0,0.0,1.0/
000          DO 10 I=1,9
000       10 A(I)=B(I)
000          RETURN
000          END$

END ELT.
@HDG,N
@FIN
```