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

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preface</td>
<td>1</td>
</tr>
<tr>
<td>I. The ATS-6 Navigation Problem</td>
<td>I-1</td>
</tr>
<tr>
<td>II. Using the ATS-6 Navigation System</td>
<td>II-1</td>
</tr>
<tr>
<td>III. Computer Program Command Formats</td>
<td>III-1</td>
</tr>
<tr>
<td>A. AOIPS Version of November 1977</td>
<td>III-1</td>
</tr>
<tr>
<td>B. McIDAS Version of November 1977</td>
<td>III-5</td>
</tr>
<tr>
<td>IV. Software Internal Description</td>
<td>IV-1</td>
</tr>
<tr>
<td>A. Description of Procedures Used by Main Programs</td>
<td>IV-2</td>
</tr>
<tr>
<td>B. Entries in Common Block NAVCOM</td>
<td>IV-5</td>
</tr>
<tr>
<td>C. Subroutine Function Descriptions</td>
<td>IV-8</td>
</tr>
<tr>
<td>References</td>
<td></td>
</tr>
<tr>
<td>Appendix A. Basic Navigation Model Applied to ATS-6 Image Data Base</td>
<td>A-1</td>
</tr>
<tr>
<td>Appendix B. Analysis of Method Which Determines ATS-6 SSP Image Coordinate Displacements Between Successive Images Resulting From Attitude Changes</td>
<td>B-1</td>
</tr>
<tr>
<td>Appendix C. Analysis of Algorithm Which Accounts for Relative Attitude Changes in Successive ATS-6 Data Images Using Earth-Edge Shift Measurements</td>
<td>C-1</td>
</tr>
<tr>
<td>Appendix D. Method of Obtaining Alternate Scan Alignment Curves</td>
<td>D-1</td>
</tr>
<tr>
<td>Appendix E. Source Code Listings of Programs and Subroutines</td>
<td>E-1</td>
</tr>
</tbody>
</table>
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 $\Delta E$ 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 APPS. 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 RD0:, then installing the tasks. The main menu driver expects them to have a name ending in 2 (e.g. LATSF2). The installation command file RD0:[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 DB0:[350,62]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 MML.

Input request:
MOUNT TAPE. ENTER DRIVE NUMBER.

Response: After mounting the tape, enter a ø or 1 for MMØ or MML 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 MMD or MML.

Input request:
ENTER PICTURE TIME (HMMSS)

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 (0,1)

Response: Enter 0 or 1 according to which drive the tape is mounted on (MM0 or MML).

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 DB0:[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 Ø NEW

To set picture start times enter:

AN Ø Ø Ø Ø HHMMSS₁ HHMMSS₂ HHMMSS₃

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 X₁ Y₁ Z₁ (decameters)
DQ SECOND SSYYDDD HHMMSS X₂ Y₂ Z₂ (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 HHHMMSS 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 $t_i$ landmarks, from at least three different locations, should have computation code $\emptyset$. Landmarks for later times should use code 3$\emptyset$ and be used as a check on the navigation.

Running program: Enter the keyin and parameters:
AN SSYYDDD $\emptyset$ $\emptyset$ (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 LATSIF 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).
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 BUFFL. Store weights and offsets in common block BUFFER. On MCDAS 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).
RLX   X component of location of time TM (earth radii).
RLY   Y component of satellite location of time TM (earth radii).
RIZ   Z component of satellite location of time TM (earth radii).
RDX   X component of satellite location of time TM (earth radii/minute).
RDY   Y component of satellite velocity at time TM (earth radii/minute).
Rdz   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) ≤ t ≤ TMX(2). Units are minutes, GMT. PTIM(i) ≤ TMN(i) ≤ 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.

SCLLI(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.

ERC0EF(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.

IELM0  Minimum element number for which alternate scan correction applies.
IELEMX

Maximum element number for which alternate scan correction applies.

ASCOEF(16)

Coefficients for 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:

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 experimenters 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, XLO, XLIN, XELE)

Input Parameters:
PTIME - Picture start time (minutes, GMT).
XLAT - Latitude (degrees) of a point on the earth's surface.
XLO - 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 +DDDDMMSS.
(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 HHHMMSS. (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:**


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 OPSTF and on COMMON/NAVCOM/.
**Name:** INVERT

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

**Input Parameters:**
AA - an N x N 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 N x N 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 \( \nabla S(Y) \cdot \text{DIR} \).
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 \( \nabla S \cdot \text{DIR} \). This line search routine returns in \( Y \) the point \( X + 2^{-N} \times \text{DIR} \) where \( N \) is the least nonnegative integer such that \( -S(2^{-N} \times \text{DIR}) \) represents at least 40\% of the functional drop in the linearization of \( S \) at \( X \) in moving from \( X \) to \( X + 2^{-N} \times \text{DIR} \).

References:
See also subroutine 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, POUT, 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 POUT.
(PTOUT is a vector)

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

VAL The value of the objective function at POUT.

ITN Number of iterations done.

Function:

MINMIZ finds the point POUT at which an objective function is a minimum.
In this case POUT 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 ATTUD.
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:
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**: Dummy variable used so call will match SMS version.
- **BETAIN**
- **BETDOT**
- **ATFRAC**

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 also subroutines ERTOST, LVTOBC, and BCTOFF.
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


### APPENDIX A

**BASIC NAVIGATION MODEL APPLIED TO ATS-6 IMAGE DATA BASE**

<table>
<thead>
<tr>
<th>Contents</th>
<th>page</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOMENCLATURE</td>
<td>A-1</td>
</tr>
<tr>
<td>I. PRELIMINARIES</td>
<td></td>
</tr>
<tr>
<td>A. Coordinate Systems</td>
<td>A-2</td>
</tr>
<tr>
<td>B. Convention for Orthogonal Matrices</td>
<td>A-5</td>
</tr>
<tr>
<td>C. Orbit Methods Used in Predicting Satellite Position $r_s(t)$</td>
<td>A-6</td>
</tr>
<tr>
<td>II. THE NAVIGATION MODEL</td>
<td>A-6</td>
</tr>
<tr>
<td>A. Earth $\rightarrow$ Satellite Transformation</td>
<td>A-6</td>
</tr>
<tr>
<td>B. Satellite $\rightarrow$ Earth Transformation</td>
<td>A-11</td>
</tr>
<tr>
<td>C. Navigation Optimization Procedure</td>
<td>A-13</td>
</tr>
</tbody>
</table>
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
    \( k = \{1, 2, 3\} \) in a ccw
    sense by an angle \( \theta \)

Vectors

\( \hat{r} \) = unit vector
\( \hat{r}_S \) = satellite radius vector
\( \hat{r}_L \) = landmark pointing vector
\( \hat{r}_E \) = earth-coordinate, EI C.S.
\( \hat{r}_E \) = 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

\( \theta, \theta, \theta^P \) = attitude yaw, roll,
     pitch angles
\( \theta_L, \phi_L \) = geodetic latitude, longitude
\( \theta_E \) = sidereal time of Greenwich prime meridian
\( \rho_L, \rho_E \) = radians/line, radians/element
\( \lambda \) = mirror step angle
\( \delta \) = mirror sweep angle
\( \alpha_i \) = navigation parameter
\( \alpha_i \) = optimized navigation parameter
LIN, ELE = satellite image coordinates:
    Line, Element
\( L, E \) = picture-center coordinates
    (line, element)
\( e \) = eccentricity of earth oblate
     spheroid 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 \( \hat{z} = -\hat{r}_s(t)/||\hat{r}_s(t)|| \), where \( \hat{r}_s \) 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 \( \mathbf{R} \) represent the orthogonal transformation of the vector \( \mathbf{r} \), (whose components are expressed in the unprimed system) to the vector \( \mathbf{r}' \) whose components are expressed in terms of the primed system, i.e. \( \mathbf{r}' = \mathbf{R}\mathbf{r} \).

The first form \( \mathbf{R} \) can take is, in matrix representation, \( \mathbf{R} = [\hat{x} \hat{y} \hat{z}]^T \), 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,

\[
\mathbf{r}' = [\hat{x} \hat{y} \hat{z}]^T \mathbf{r} = (x' \hat{r}, y' \hat{r}, z' \hat{r})^T .
\]

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

The second form of \( \mathbf{R} \) is expressed in terms of rotation angles where in its simplest form \( \mathbf{R} = \mathbf{R}(\theta,k) \) represents a rotation by an angle \( \theta \) counter-
clockwise 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 $\theta$ counterclockwise relative to the $xyz$ system about the $z$-axis,

$$\mathbf{r}' = R(\theta,3)\mathbf{r} = \begin{bmatrix} \cos \theta & \sin \theta & 0 \\ -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{r}.$$  

[ROTATE]*

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

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

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 $\theta_y$ about the BC z-axis (yaw), followed by a ccw rotation by an angle $\theta_r$ about the BC x-axis (roll), followed by a cw rotation by an angle $\theta_p$ about the BC y-axis (pitch). Thus a vector expressed in the LV system $^r_{LV}$, is transformed to the BC system ($^r_{BC}$) by the operation:

$$^r_{BC} = R(-\theta_p,2) \ R(\theta_r,1) \ R(\theta_y,3) \ ^r_{LV},$$  

[LVTTOBC]

where the angles $\theta_y, \theta_r, \theta_p$ are generally dependent on time.

C. Orbit Methods Used in Predicting Satellite Position, $^r_s(t)$

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 ($^r_s$) and velocity ($^r_s$) 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,

\[
\hat{r}_{s1}(t) = \hat{r}_{s2}(t) \rightarrow \hat{r}_{s1}(t), \hat{r}_{s1}(t), \quad t \neq t_2. \quad \text{[GASORB]}
\]

The vector spread between \(\hat{r}_{s1}(t)\) and \(\hat{r}_{s2}(t)\) must be less than 70\(^\circ\). Once \(\hat{r}_{s1}(t)\), \(\hat{r}_{s2}(t)\) are determined, the satellite's position for any other time is determined by an iterative \(f, g\) computational procedure:

\[
\hat{r}_{s}(t) = f \hat{r}_{s1}(t) + g \hat{r}_{s1}(t). \quad \text{[ORBIT]}
\]

The details of the computational algorithms for these two procedures are given elsewhere.\(^\dagger\) 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

\(^\dagger\)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 \( \mathbf{r}_s \) and the landmark \( \mathbf{r}_2 \) is given by:

\[
\mathbf{r}_1(t) = \mathbf{r}_2(t) - \mathbf{r}_s(t), \quad \text{[ERTOST]} \tag{1}
\]

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

\[
\mathbf{r}_2 = R(-\theta_E, 3) \mathbf{r}_E, \quad \text{[ERTOST]} \tag{2}
\]

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

\[
\rho = \frac{r_{eq}}{\sqrt{1 - e^2 \sin^2 \theta_L}},
\]

\( \theta_L, \phi_L \) = the geodetic latitude and longitude of the landmark,

\( e = \text{eccentricity of the oblate spheroid earth model} = 8.1812 \times 10^{-2} \),

\( r_{eq} = \text{earth's equatorial radius} = 6378.15 \text{ km} \).

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

\[
\theta_E = a_1 + a_2 \times \text{DDD} + a_3 \times t \quad \text{[ERTOST]} \tag{3}
\]

where, \( \theta_E \) 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

\[
a_1 = 99.59477026 \text{ degrees}
\]

\[
a_2 = .985647336 \text{ degrees/DDD}
\]

\[
a_3 = .2506844773 \text{ degrees/decimal minute}.
\]
It is convenient at this stage to transform the pointing vector \( \hat{\mathbf{r}}_1 \) into a unit vector, \( \hat{\mathbf{r}}_1 (t) = \hat{\mathbf{r}}_1 (t)/\| \hat{\mathbf{r}}_1 (t) \| \). Transformation of \( \hat{\mathbf{r}}_1 \) into the LV frame is accomplished by \( R_{LV} \),

\[
\hat{\mathbf{r}}_{LV} = R_{LV} \hat{\mathbf{r}}_1 = [\hat{x} | \hat{y} | \hat{z}]^T \hat{\mathbf{r}}_1 \quad \text{[STTOLV]} \tag{3}
\]

where, from the definition of the LV coordinate system,

\[
\hat{\mathbf{z}} = -\hat{\mathbf{r}}_S (t)/\| \hat{\mathbf{r}}_S (t) \| = -(x_S, y_S, z_S)^T,
\]

\[
\hat{\mathbf{x}} = (\hat{\mathbf{z}} \times \hat{\mathbf{e}}_3)/\| \hat{\mathbf{z}} \times \hat{\mathbf{e}}_3 \| = (-y_S/d, x_S/d, 0)^T,
\]

\[
\hat{\mathbf{y}} = \hat{\mathbf{z}} \times \hat{\mathbf{x}} = (x_S z_S/d, y_S z_S/d, -d^2)^T,
\]

\[
d = \sqrt{x_S^2 + y_S^2},
\]

where \( x_S^2 + y_S^2 + z_S^2 = 1 \).

In matrix form,

\[
R_{LV} = \frac{1}{d} \begin{bmatrix}
-y_S & x_S & 0 \\
x_S z_S & y_S z_S & -d^2 \\
-dx_S & -dy_S & -dz_S
\end{bmatrix}.
\]

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:

\[
\hat{\mathbf{r}}_{BC} = \hat{\mathbf{r}}_{LV} \tag{4}
\]

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

\[
\hat{\mathbf{r}}_{PF} = (x_{PF}, y_{PF}, z_{PF})^T = R_{PF} \hat{\mathbf{r}}_{BC}, \tag{5a}
\]

where \( R_{PF} = R(\varphi_p, 2) R(\varphi_r, 1) R(\varphi_y, 3) \), \( \varphi \) being the attitude angles.
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

\[
\begin{align*}
\text{LIN} &= L_c + \left( \sin^{-1} y_{PF} / \rho_L \right) + \Delta L_e \quad \text{[PPTOTC]} \\
\text{ELE} &= E_c + \left( \tan(x_{PF} / z_{PF}) / \rho_E \right) + \Delta E_e \quad \text{and [EDCCOR]} 
\end{align*}
\]

where,

\( L_c = \) line number,

\( E_c = \) element number,

\( \Delta L_e = \) line correction due to attitude shifts,

\( \Delta E_e = \) 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,

\( \rho_L = \) number of radians/line \{ nominal ATS-6 scanned field is \( 20^\circ \times 20^\circ \) \}

\( \rho_E = \) number of radians/element

Summarizing equations (1) - (5),

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

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

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

\[ \hat{r}_{PF} = (-\cos \lambda \sin \delta, -\sin \lambda, \cos \lambda \cos \delta)^T \]  \hspace{1cm} [PFTOTC]  \hspace{1cm} (8a)

where

\[ \lambda = (L_c - LIN + \Delta L_e) \rho_L = \text{mirror step angle} \]  \hspace{1cm} (8b)
\[ \delta = (E_c - ELE + \Delta E_e) \rho_E = \text{mirror sweep angle} \]  \hspace{1cm} (8c)

From equation (7a),

\[ \hat{r}_{l} = R_L^T R_B^T R_P^T \hat{r}_{PF}. \]  \hspace{1cm} [SE]  \hspace{1cm} (9)

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

Now

\[ \hat{r} = \hat{r}_s + s \hat{r}_l, \]  \hspace{1cm} (10)

where \( s = \| \hat{r}_l(t) \| \) 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,

\[ \left( \frac{x_E^2 + y_E^2}{r_{eq}^2} + \frac{z_E^2}{r_p^2} \right) = 1 \]  \hspace{1cm} (11)

where,

\[ r_{eq} = \text{earth's equatorial radius} = 6378.15 \text{ km}, \]
\[ r_p = \text{earth's polar radius} = 6356.77 \text{ km}, \]
\[ x_E, y_E, z_E \]  are the vector components of the landmark, \( \hat{r}_E \), 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,

$$
\hat{r}_2 \ast = \hat{r}_s \ast + s \hat{r}_1 \ast
$$

(12)

where,

$$
\hat{r}_2 \ast = R(-\theta_E, 3) \hat{r}_E \ast = R(-\theta_E, 3) (x_E / r, y_E / r, z_E / r) ^T,
$$

$$
\hat{r}_1 \ast = (x / r, y / r, z / r) ^T,
$$

$$
\hat{r}_s \ast = (x_s / r, y_s / r, z_s / r) ^T.
$$

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

$$
\| \hat{r}_2 \ast \| = 1 = \| \hat{r}_s \ast + s \hat{r}_1 \ast \|,
$$

(13)

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

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

(14)

where

$$
\begin{align*}
RAD &= B^2 - 4AC, \\
A &= F + (1 - F) z_1^2, \\
B &= 2(x_1 x_s + y_1 y_s) F + 2z_1 z_s, \\
C &= (x_s^2 + y_s^2) F + z_s^2 - r_s^2, \\
F &= r_s^2 / r_{eq}^2.
\end{align*}
$$

[ERTOST]

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

$$
\hat{r}_E = R(\theta_E, 3) (\hat{r}_s \ast + s \hat{r}_1 \ast) = (x_E, y_E, z_E) ^T,
$$

[ERTOST]

(15)

and hence

$$
\theta_L = \tan^{-1}[z_E / ((1 - e^2) \sqrt{x_E^2 + y_E^2})],
$$

[ERTOER]

(16a)
\[ \phi_L = \tan^{-1}\left[ \frac{y_E}{x_E} \right], \quad \text{(16b)} \]

where

\[
\theta_L = \text{geodetic latitude}, \\
\phi_L = \text{longitude} \\
e^2 = \frac{r_{eq}^2 - r_p^2}{r_{eq}^2} = \text{earth's eccentricity squared.}
\]

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

C. Navigation Optimization Procedure (Subroutine ATTUD)

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, \((\theta_L, \phi_L)\), the associated image coordinates, \((\text{LIN, ELE})\), and the time, \(t\), at which the landmark was imaged.

Let

\[ \alpha_i = \text{ith parameter to be optimized}, \]
\[ R_A(\alpha_i) = \text{transformation associated with the } \alpha_i, \]
\[ \hat{r}_k' = \text{unit pointing vector of the } k^{th} \text{ landmark derived from } (\theta_L, \phi_L)_k', \]
\[ \hat{r}_k = \text{unit pointing vector of the } k^{th} \text{ landmark derived from } (\text{LIN, ELE})_k', \]

and

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

which is a sum over all landmark measurements included in the optimization.

The navigation is complete when a set of parameters \(\alpha_i\) are found (call
them \( \bar{\alpha}_1 \) which minimizes \( S \). The optimized values \( \bar{\alpha}_1 \) are then used in the transformation to predict \( (\theta_L, \phi_L) \) from an arbitrary \( (\text{LIN}, \text{ELE}) \).

As an example, consider equation (5a),

\[
\hat{r}_{PF} = R_{PF} \hat{r}_{BC}
\]

[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 \( \theta_y, \theta_r, \theta_p \) 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

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

[S]

and the values \( \bar{\theta}_y, \bar{\theta}_r, \bar{\theta}_p \) 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

\[ \alpha = \{\alpha_i\} \text{ for convenience}, \]

\[ \alpha^n = \text{value of } \alpha \text{ resulting from the } n\text{th iteration}, \]

\[ N = \text{total number of parameters } \alpha_i, \]

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

\[ VS = \text{gradient of } S, (VS)_i = \partial S/\partial \alpha_i. \]

The iterative procedure works as follows:

1) Start with \( m = 0 \) and increase \( m \) by 1 until a value \( M \) is found such that
\[ S(\alpha^n) - S(\alpha^n - 2^{-M} H^{-1} \nabla S(\alpha^n)) \geq (.4) 2^{-M} H^{-1} \nabla S(\alpha^n) \cdot \nabla S(\alpha^n), \quad [LS] \]

2) When the inequality is satisfied, set
\[ \alpha^{n+1} = \alpha^n - 2^{-M} H^{-1} \nabla S(\alpha^n), \quad [\text{MINIMIZ}] \]

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 \( \alpha^n = \alpha^{n+1} \) and go to 1); if yes, then \( \alpha^{n+1} \rightarrow \alpha \) 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 $\Delta E_R$, $\Delta E_L$ 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 T₁-SSP and T₂-SSP. Furthermore, point 5 is the bisect point for the T₂-chord coinciding with line L, and both T₁- and T₂-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

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

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

solving for $\Delta E$,

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

The second identity is

$$\overline{P_4 - P_2} - \Delta E_R = \overline{P_3 - P_1} - \Delta E_L,$$ where \hspace{1cm} (2)
\[
P_4 - P_2 = 2 \sqrt{(P_4 - P_0')^2 - (P_5 - P_0')^2}
\]

\[
= 2 \sqrt{(P_3 - P_0')^2 - (P_5 - P_0')^2}
\]

\[
= 2 \sqrt{(L - L_c)^2 + (E - E_c)^2 - [L - (L_c + L_c)]^2}
\]

\[
, \text{ and}
\]

\[
P_3 - P_1 = 2(E - E_c)
\]

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

\[
\delta = \sqrt{X^2 + Y^2 - (Y - \Delta L)^2} - X, \text{ where}
\]

\[
\delta = (\Delta E_R - \Delta E_L)/2
\]

\[
X = E - E_c
\]

\[
Y = L - L_c;
\]

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

\[
\Delta L = Y \pm \sqrt{\text{RAD}}, \text{ where}
\]

\[
\text{RAD} = Y^2 - 2X\delta - \delta^2.
\]

The choice of sign in (6) is determined by substituting the expression for \(\delta\) in Eq. (5) in \(\text{RAD}\); the result is

\[
\Delta L = Y \pm \sqrt{(Y - \Delta L)^2} = Y + \sqrt{\text{RAD}}, \; Y > \Delta L
\]

\[
= Y - \sqrt{\text{RAD}}, \; Y < \Delta L.
\]

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

\[
\Delta L = Y \pm \sqrt{\text{RAD}}, \; Y > 0,
\]

\[
\text{RAD} = Y^2 - 2X\delta - \delta^2.
\]
For the case where \(|Y|\) approaches \((\Delta L)\) 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 \(\Delta E_R\) vs \(L\) and a \(\Delta E_L\) 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 \( T_1, T_2, \ldots, T_n \),
from which the displacements in earth coordinates of a feature (cloud)
between successive images are to be determined. Let

\[
(L_i, E_i) = \text{image coordinates (Line, Element) of the cloud feature}
\]

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

\[
\hat{r}_{LVi} = \text{unit Local Vertical pointing vector associated with } \hat{r}_{PFI} ;
\]

\[
R_i = \text{3x3 rotation matrix which transforms without error } r_{LVi}
\]

\[
\text{into } \hat{r}_{PFI} \text{ at the time the feature was scanned, i.e.}
\]

\[
\hat{r}_{PFI} = R_i \hat{r}_{LVi} . \tag{1}
\]

1. A Two-Image Sequence

Consider now the first two images \( T_1, T_2 \) in the sequence whose relation-
ship between LV and PF coordinates are given by

\[
\hat{r}_{PFI} = R_1 \hat{r}_{LV1} \tag{2}
\]

\[
\hat{r}_{PFI} = R_2 \hat{r}_{LV2} . \tag{3}
\]
The relationship of the cloud feature in LV coordinates for image $T_2$ can be related to $T_1$ as follows:

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

(4)

where $\Delta \hat{r}_{LV}$ 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 $\Delta T_{12}$ caused by a non-zero angle of inclination in the satellite's orbit (the eccentricity of the ATS-6 orbit was so small (~ $10^{-4}$) 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 antisymmetric matrix whose elements are the small-angle differences between the PF axes at times $T_1$ and $T_2$. Thus
\[ R_2 = (I + E_{12}) R_1 \]  
(5)

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

\[ \hat{r}_{PF2} = R_1 \hat{r}_{LV1} + E_{12} R_1 \hat{r}_{LV1} + R_1 \hat{r}_{LV}(\Delta T_{12}) + E_{12} R_1 \hat{r}_{LV}(\Delta T_{12}) \]  
(6)

The first term on the RHS of (6) is \( \hat{r}_{PF1} \), the last term is negligible (\(-10^4\) 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

\[ \hat{r}_{PF2} = R_1 \hat{r}_{LV2} + E_{12} \hat{r}_{PF1} \]  
(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 \( \Delta L_{12} \) is proportional to a small change in roll \( \Delta R_{12} \) about the PF-X axis and the element shift is proportional to small changes in pitch \( \Delta P_{12} \) about the PF-Y axis, i.e.

\[ \Delta L_{12} = \Delta R_{12} / \rho \]  
(8a)

\[ \Delta E_{12} = \Delta P_{12} / \rho \]  
(8b)

where \( \rho \) is the angular size of a pixel = \( 1.45 \times 10^{-8} \) 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
\[
\begin{align*}
E_{12} \hat{r}_{PF1} &= \begin{bmatrix}
0 & 0 & \Delta P \\
0 & 0 & \Delta R \\
-\Delta P & -\Delta R & 0
\end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \\
&= [\Delta P, \Delta R, -(\Delta PX + \Delta RY)]^T,
\end{align*}
\]

where it is understood that \(\Delta R, \Delta P\) 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

\[
\hat{r}'_{PF2} = \hat{r}_{PF2} - E_{12} \hat{r}_{PF1} = R_1 \hat{r}_{LV2},
\]

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 \(\Delta L_{12}, \Delta E_{12}\) due to relative
attitude changes are computed and subtracted from \(\Delta L^T, \Delta E^T\).
These differences are added to \(L_1, E_1\) and the associated
PF vector computed. That this vector is a good approximation
of \(\hat{r}_{PF2}^{1}\) is shown below.

By definition,

\[
\hat{r}_{PF1} = \hat{r}_{PF}(L_1, E_1) \tag{11}
\]

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

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 \((L_1 + \Delta L^T, \n E_1 + \Delta E^T)\) keeping first order terms, we have

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

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

\[
\hat{r}_{PF} = [-\cos \lambda \sin \delta, -\sin \lambda, \cos \lambda \cos \delta]^T \tag{14}
\]

\[= [X, Y, Z]^T
\]

where

\[\lambda = (L_c - L)\rho = \text{mirror step angle},
\]

\[\delta = (E_c - E)\rho = \text{mirror sweep angle},
\]

\[(L_c, E_c) = \text{image center line and element value},
\]

\[\rho = \text{angular size of a line or element}.
\]
Taking the partials of \( \hat{r}_{PF} \) in (14) with respect to \( L, E \), and retaining first order terms with the partials evaluated at \( (L_2, E_2) \):

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

\[= [0, PZ, -PY]^{T} \Delta L_{12} + [PZ, 0, -PX]^{T} \Delta E_{12}, \]

(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 \( \hat{r}_{PF2} \), we have

\[
\hat{r}_{PF}(L_1 + \Delta L^{T} - \Delta L_{12}(L_2), E_1 + \Delta E^{T} - \Delta E_{12}(L_2))^{*}
\]

\[\hat{r}_{PF2} - E_{12} \hat{r}_{PF} = \hat{r}_{PF2}' \quad \text{by} \]

(16)

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

3. **Transforming PF to LV coordinates**

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

\[
\hat{r}_{PF1} = R_1 \hat{r}_{LV1}
\]

(17)

\[
\hat{r}_{PF2} = R_1 \hat{r}_{LV2} \quad \text{(see (10))}
\]

(18)

Let \( \bar{R} \) 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. \( \bar{R} \) is a constant matrix. \( \bar{R} \) 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 nonlinearity. Similarly, as above, we can relate $\overline{R}$ to $R_1$ by an infinitesimal transformation

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

(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 nonlinearity as a function of image coordinates repeat from image to image.)

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

$$\hat{r}'_{LV_1} = R^T \hat{r}^*_{PF_1} = (I + e) R^T R \hat{r}_{LV_1} = \hat{r}_{LV_1} + \hat{r}_e$$

(20a)

$$\hat{r}'_{LV_2} = R^T \hat{r}^*_{PF_2} = (I + e) R^T R \hat{r}_{LV_2} = \hat{r}_{LV_2} + \hat{r}_e ,$$

(20b)

where

$$\hat{r}_e = e \hat{r}_{LV} .$$

(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 $\hat{r}_e$ 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 $T_1$ can be related to $R_1$ by the image pair relative-attitude transformation matrices $E_{i,i+1}$ where

$$R_n = (I + E_{n-1,n}) \cdots (I + E_{2,3})(I + E_{1,2})R_1$$

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

(22)

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

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

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

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

(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

$$\hat{r}_{LVi}' = \hat{r}_{LVi} + \hat{r}_e \quad i = 1, 2, ..., n.$$  

(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 $\Delta E$ 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 $\Delta E$ 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:

\[ e = \text{element number} \ (1 \leq e \leq 2400) \]

\[ s = \text{scan number} \ (1 \leq s \leq 1200) \]

\[ p_e(s,e) = \text{pixel digital value at a given even-scan, element position} \]
\(p_0(s,e)\) = pixel digital value at a given odd-scan, element position

\(\delta\) = 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:

\[
C(s,e,\delta) = \sum_{e' = e - t}^{e + t} [p_0(s,e') - p_e(s + 1, e' + \delta)]^2 \quad (s \text{ odd})
\]

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

\[
C(s,e,\Delta E(s,e)) \leq C(s,e,\delta) \quad \text{for all values of } \delta
\]

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 \(C\).

\[
w(s,e) = \max(p_0(s,e')) - \min(p_0(s,e'))
\]

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 \( \Delta E \) 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 $\Delta E(e)$ 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///'*** ATSF6 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 EDGE DATA
717 FORMAT(74H 6 RUN NAVIGATION
718 FORMAT(74H 7 EXIT ATSF6 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=5
MENU=1
CALL INCOM('1',O,HU,1,MIN,MENU,1,1)
IF(MENU.LE.0) GO TO 990
I=MIN(1)
GO TO (1,2,3,4,5,6,7),IJ
1 CALL REQUEST(RADSO('R61TH2'))
GO TO 999
2 CALL REQUEST(RADSO('RDFIG2'))
GO TO 999
3 CALL REQUEST(RADSO('RDFST2'))
GO TO 999
4 CALL REQUEST(RADSO('RLATSF2'))
GO TO 999
5 CALL REQUEST(RADSO('LCUGFT2'))
GO TO 999
6 CALL REQUEST(RADSO('LATSNV2'))
GO TO 999
7 CALL REQUEST(RADSO('MET1'))
GO TO 999
990 CONTINUE
CALL REQUEST(RADSO('ME1'))
999 CONTINUE
END
PROGRAM TO NAVIGATE AT656 IMAGES FROM LANDMARKS.

INTEGER I, ILEN, IYD, ITIME
DIMENSION PTIME(150), ALIN(150), AELE(150), ALAT(150), ALON(150)
DIMENSION ILIN(150), ITELE(150), ILAT(150), ILEN(150), ICODE *
(150)
DIMENSION MIN(10), MOUT(24)
COMMON/NAVCOM/NAV, IAV, IYR, IDAY, TOTILN, DEGLN, TOTILE, DEGLE, PICLI
*N, PTIME, ILIN, ILEN, ITIME, IN, KIA, K1, K2, R1D, R1D, R1D, PITCH, ROLL
*N, IAM, PTIME(3), IM(3), LM(3), NLCOEF(2), ARCUEF(2), SCLO(2), SCLL1(2)
*ALCOEF(11, 2), SCLO(2), SCLL1(2), ERCOEF(11, 2), NASCEF, SCLASO, SCLAS1.
*IELLEN, ILELEN, ASCUEF(16)
DATA P, 3, 4, 15926535/
DATA IT/5, UP/6/
RADDEG=PI/180.0
LUN=10
OPEN(UNIT=1, LUN, NAME='DAT011350,621A6LNR5.DAT', TYPE='OLD', READONLY)
2 CONTINUE
10 CONTINUE
70 FORMAT(2X, 1019)
GO TO 10
40 CONTINUE
708 FORMAT(1X, 1019)
50 CONTINUE
706 FORMAT(1X, 'DAY=', I5, ' NUMBER OF LANDMARKS=', I3)
NLNK=0
DU 50 I=1, NL
WRITE(LP, 708) ITIME(1), ICODE(1), ILIN(1), ITELE(1), ILAT(1), ILEN(1)
708 CONTINUE
51 CONTINUE
706 CONTINUE
NLNK=NLNK+1
IF (ICODE(1) .LT. 100, 100, 10)
NLNK=NLNK+1
TIME(NLKNK)=TIME(1)
ALIN(NLKNK)=FLOAT(ILIN(1))
AELE(NLKNK)=FLOAT(ITELE(1))
ALAT(NLKNK)=FLOAT(ILAT(1))
ALON(NLKNK)=FLOAT(ILEN(1))
WRITE(LP, 700) PTIME(NLKNK), ALIN(NLKNK), AELE(NLKNK), ALAT(NLKNK), ALON(NLKNK)
780 CONTINUE
100 CONTINUE
CLOSE(UNIT=LP)
10 IF (NLNK .LT. 100) GO TO 990
CALL ATTILE(PTIME, ALIN, AELE, ALAT, ALON, NLKNK)
P=PI2RADDEG
RD=RADD/100.0
ATSV/AOIPS

ID=IAW/RADDEG
WRITE(11,701)PD,RO,ID
701 FORMAT(1X,'PIEC=','A16.9',' ROLL=','A16.9',' YAW=','A16.9)
195 CONTINUE
DO 200 I=1,NL
PM=PM1ME(I)
XLA=FLALO(I,1A(I))
XLO=FLALO(I,10ON(I))
CALL ES(PM,XLA,XLO,XLIN,XLE)
XLIN=ILIN(I)-XLIN
XLEE=XLEE(I)-XLE
200 CONTINUE
401 CONTINUE
710 FORMAT(3X,'A16.5X,A15.5X,2F10.2)
GO TO 999
999 CONTINUE
CALL REQUE(S(RAD50('ATSF2'))
END
*** *** ATSV/MCIDAS *** ***

SULTAN, AF. ATSV/MCIDAS
ELT007 REL062 12/22-17:01:39-(0)
000001 000 $SJOB ATSVN U3200
000002 000 $OPTION *S*, *9*, 20
000003 000 $FORTRAN
000004 000 $FORTRAN
000006 000 $C PROGRAM TO NAVIGATE AT56 IMAGES FROM LANDMARKS.
000006 000 $LOGICAL OPTION
000006 000 $INTEGER POS, GLEB
000006 000 $EQUIVALENCE (ICON, NAVN)
000006 000 $DIMENSION ICON(1)
000010 000 $DIMENSION PTIME(150), ALIN(150), AELE(150), ALAT(150), ALON(150)
000011 000 $DIMENSION ITIME(150), ILIN(150), IELE(150), ILAT(150), ION(150), ICODE
000012 000 *(150)
000013 000 $DIMENSION MSQ(10)
000014 000 $DIMENSION MIN(10), ROUT(24)
000015 000 $DIMENSION GLEB(672)
000016 000 $COMMON/NAVCON/NAVN, NAV, IYR, IDAY, TOTLIN, DEGLIN, DEGELE, PINI
000017 000 $*N, PICLE, TPSCL, I0YR, I0DAY, TM, RX, R1Y, R1Z, RDY, R1D, PDY, P1D, PITCH, ROL
000018 000 $*L, LNAV, TIM(3), TMX(3), NLCOEF(2), NRECDE(2), SCLD(2), SCLL(2)
000019 000 $*ELCOEF(11,2), SCLR0(2), SCLR1(2), ERCOE F(11,2), NASCEF, SCLAS0, SCLAS1
000020 000 $IELEK, IELEMX, ASCOE F(16)
000021 000 $DATA PI/3.1415926535/
000022 000 $DATA MSQ/6HDEFATD, B*0/
000023 000 $DATA ICODE/150*0/
000024 000 $DATA MIN/6HATSNAV, B*0/
000025 000 CALL I4(MIN)
000026 000 NDAY=MIN(1)
000027 000 JOUT=1
000028 000 IF(OPTION(MIN(4), 3H P)) JOUT=2
000029 000 IF(.NOT.OPTION(MIN(2), 3H A), .AND., .NOT.(MIN(2), 6G.0 )) GO TO 195
000030 000 IPT1=MIN(5)
000031 000 IPT2=MIN(6)
000032 000 IPT3=MIN(7)
000033 000 NAVDAY=MOD(NDAY, 100000)
000034 000 ISS=RDAY/100000
000035 000 ISS=(ISS/2)*2
000036 000 KDAY=ISS*100000*NAVDAY
000037 000 RADDEG=PI/180.0
000038 000 IF(MIN(5), NE.3NEW) GO TO 2
000039 000 DO 3 I=1, 203
000040 000 3 ICOM(I)=0
000041 000 CALL WCOM
000042 000 2 CONTINUE
000043 000 CALL GETNAV(KDAY, IECLASS)
000044 000 IF(IPT1, NE.0) PTIM(1)=FTIME(IPT1)
000045 000 IF(IPT2, NE.0) PTIM(2)=FTIME(IPT2)
000046 000 IF(IPT3, NE.0) PTIM(3)=FTIME(IPT3)
000047 000 IF(IECLASS, GE.10) GO TO 990
000048 000 CALL HEDDER(KDAY, GLEB, POS)
000049 000 IF((GLEM (POS+1), NE. KDAY. OR. GLEB(POS+2), LT. 0)) GO TO 990
000050 000 INDEX=6*GLEM(POS+2)+88
000051 000 NL=0
000052 000 10 CALL SCRA(30, INDEX)
000053 000 CALL READ(30, 672, GLEB)
000054 000 DO 50 I=1, 661, 6
000055 000 KIND=GLEM(1)/100000
000056 000 IF(KIND, EQ.0) GO TO 40
000057 000 IF(KIND, NE.1) GO TO 30
000058 000 IF(NL, GE.150) GO TO 30
000059 000 NL=NL+1
000060 000 ITIME(NL)=GEBL(I+1)
000061 000 ICODE(NL)=MOD(6*GEBL(I+1) , 100000)
000062 000 ILIN(NL)=GEBL(I+2)
000063 000 IELE(NL)=GEBL(I+3)
000064 000 ILAT(NL)=GEBL(I+4)
000065 000 ILON(NL)=GEBL(I+5)
000066 000 30 CONTINUE
000067 000 JJ=GEBL(672)
000068 000 IF(JJ, LT.0) GO TO 40
000069 000 INDEX=6*JJ+88
000070 000 GO TO 10
000071 000 40 CONTINUE
000072 000 KC=2*HDL
000073 000 IF(.NOT.OPTION(MIN(3), 3H L)) GO TO 51
000074 000 DO 50 I=1, NL
000075 000 ENCODE(132, 208, MOUT), KC, NDAY, ITIME(I), ICODE(I), ILIN(I), IELE(I),
000076 000 *ILAT(I), ILON(I)
000077 000 CALL TP(JOUT, MOUT)
000078 000 708 FORMAT(1X, A2, 8I9)
000079 000 50 CONTINUE
000080 000 51 CONTINUE
000081 000 ENCODE(132, 206, MOUT), DAY, NL
000082 000 CALL TP(JOUT, MOUT)
000083 000 706 FORMAT(1X, "DAY=", I5, " NUMBER OF LANDMARKS=", I3)
000084 000 NLMK=0
000085 000 DO 100 I=1, NL
000086 000 IF(NOD(ICODE(I)/100, 10), NE.0) GO TO 100
000087 000 NLMK=NLMK+1
000088 000 PTIME(NLMK)=FTIME(ITIME(I))
000089 000 ALIN(NLMK)=FLOAT(ILIN(I))
000090 000 AELE(NLMK)=FLOAT(IELE(I))
000091 000 ALAT(NLMK)=FLOAT(ILAT(I))
000092 000 ALON(NLMK)=FLOAT(ILON(I))
000093 000 100 CONTINUE
000094 000 IF(NLMK, LE.0) GO TO 990
000095 000 CALL ATTITUDE(PTIME, ALIN, AELE, ALAT, ALON, NLMK)
000096 000 PD=PITCH/RADDEG
000097 000 RD=ROLL/RADDEG
000098 000 YD=YAW/RADDEG
000099 000 ENCODE(132, 701, MOUT), PD, RD, YD
000100 000 CALL TP(JOUT, MOUT)
000101 000 701 FORMAT(1H0, 1X, "PITCH=", E16.9, " ROLL=", E16.9, " YAW=", E16.9)
000102 000 CALL WCOM
000103 000 MSG(3)=KDAY
000104 000 MSG(4)=ILALO(RD)
000105 000 MSG(5)=ILALO(RD)
000106 000 MSG(6)=ILALO(YD)
000107 000 MSG(6)=ILALO(YD)
000108 000 195 CONTINUE
000109 000 IF(.NOT.OPTION(MIN(2), 3H R), AND...NOT.(MIN(2), EQ.0 )) GO TO 201
000110 000 CALL GETNAV(KDAY, IEXIST)
000111 000 DO 200 I=1, NL
000112 000 FTH=FTIME(ITIME(I))
****** ATSNV/MCIDAS ******

000113  000  XLA=FLAL0(ILAT(1))
000114  000  XLO=FLAL0(ILON(1))
000115  000  CALL ES(PTM,XLA,XLO,XLIN,XLEL)
000116  000  RLIN=ILIN(1)-XLIN
000117  000  RELE=ILEL(1)-XLEL
000118  000  ENCODE(132,710,MOUT)ITIME(1),ICODE(1),RLIN,RELE
000119  000  CALL TP(JOUT,MOUT)
000120  000  200 CONTINUE
000121  000  201 CONTINUE
000122  000  710 FORMAT(3X,16,5X,15,5X,2F10.2)
000123  000  RETURN
000124  000  990 CONTINUE
000125  000  CALL EMESS(3HREQ,NDAY)
000126  000  RETURN
000127  000  ENDS
000128  000  $FILENAME
000129  000  DELETE ATSNV,GORP
000130  000  $INCLUDE HEDDER
000131  000  $CATALOG
000132  000  NAME=ATSNV,5,R,W,D
000133  000  TYPE=FG
000134  000  LIB=ATSFNL,LL
000135  000  BEGIN
000136  000  $EOJ

END ELT.

$HDG,P  ***** ATTUD  *****
SUBROUTINE ATTUD MPs, ALN, AEL, ALA, ALI, NLK

SUBROUTINE TO Compute ATSU ATTITUDE FROM LANDMARKS.

YAW, PITCH, AND ROLL VALUES FOR A SMALL OFFSET OF THE PICTURE

FRAME COORDINATES FROM BODY CENTERED COORDINATE SYSTEM.

DIMENSION PTM(1), ALIN(1), AEL(1), ALA(1), ALI(1)

DIMENSION X(3, 150), Y(3, 150), TIME(150), PRY(3)

COMMON/NAVCOM/NAV, INAV, DIR, IDAY, TOTLIN, DEGLIN, TOTIOL, DEGELE, PI

*PI, PICLE, TMPSC, IDIR, IDAY, TM, R1X, R1Y, R1Z, R1DX, R1DY, R1DZ, PITCH, ROLL

*L, YAW, PTM(3), TMX(3), TXM(3), NRCL(2), GRDCOE(2), SCLL(2), SCLL2(2),

*NLXCOE(7, 2), SCLL3(2), SCUR1(2), ERGOC(15, 2), NASC, RSLAS, RSLAS1,

*ELEM, IELEM, ASCOE(7, 2),

COMMON/MINCOM/X, Y, TIME, NP

DIMENSION MOUT(24)

DATA PI/15926/1

DATA LP/6/

RDPDG=PI/180.0

RADLIN=RDPDG*DEGLIN/TOTLIN

RDELE=RDPDG*DEGELE/TOTIOL

NP=NLK

DO 100 I=1, NLK

ISCA=1200-FIX(ALIN(I))-1/2

TIME(I)=PTM(I)+ISCA*TMPSCL

IDIR=1

CONVERT LAT, LONG TO ROTATING EARTH CO-OR

CALL ERTOE(ALT(1), ALON(1), X1ER, X2ER, X3ER, IDIR)

CONVERT ROTATING EARTH TO INERTIAL EARTH COORDINATES

CALL ERTOE(X1ER, X2ER, X3ER, X1, X2, X3, IDIR, TIME(I))

EARTH INERTIAL TO SATELLITE LOCAL VERTICAL

CALL STOLV(X1, X2, X3, IDIR, TIME(I))

NOTE: ATTITUDE DATA NOT USED THUS LOCAL VERTICAL SAME AS BODY CENTE

X1(I, J)=X1

X2(I, J)=X2

X3(I, J)=X3

CONTINUE

DO 200 I=1, NLK

ELEANG=PI*CLE-AEL(I))*RDELE

ALANG=PIPICLN-AEL(I))*RADLIN

Y(1, J)=SINC(ELEANG)*COS(ALANG)

Y(2, J)=SINC(ALANG)

Y(3, J)=COS(ELEANG)*COS(ALANG)

CONTINUE

PTRY1)=0

PTRY2)=0

PTRY3)=0

CALL MINIMOR(PTR, PTRY, GNRV, VALUE, ITER)

PITCH=PTRY1)

ROLL=PTRY2)

YAW=PTRY3)

ENCODE(150, 702, MOUT) ITER, GNRV, VALUE

FORMAT(1CH, 5X, "CONVERGENCE AT ITERATION", 16, //X, "GRADIENT NORM=",

*E16.9, 8 FINAL VALUE=", E16.9)

CALL T(TOUT(MOUT))

RETURN

END
A6INT/AOIPS

INTEGER*4 ITIME
DIMENSION RIN(10), KMIN(10), R1(3), R2(3)
COMMON/NAVCOM/NAV, IHAV, IYR, IDAY, ITOLL, DELL, DTOLL, UGEL, PIGLI
*4, PICEH, IMPSC, AYR, IDUAY, LM, MX1, MX2, MX3, MX4, RYI, RIZ, RXI, RLY, RIZI, PICH, ROL
*4, IYN, PL9, IMA(4), IMAX, IMUCOF(4), NRPEOF(4), SCLU(2), SCLL(2)
*4, ERPEOF(11, 12), SCLR(2), SCLA(12), ERPEOF(11, 2), NASCF, SCLAS0, SCLAS1,
*4, IEMAX, IEMAX, ASCOF(16)
DATA PL1/3, 1415926535/
DATA UP7/6/
PREADM TO INITIALIZE ATS 6 NAVCOM
RADEG=PL1/180.0
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
JPRF=MIN(1)
MENU=1
CALL INCOM(' ENTER DAY NUMBER ', 17, 1, 1, MIN, MENU, 1, 1)
IF(MENU.LE.0) GO TO 999
IDAY=1111
TOLL=2400.0
DELL=19.92
TOLL=2400.0
UGEL=20.07
PICEH=(TOLL+1.0)/2.0
PICEH=(TOLL+1.0)/2.0
IMPSC=0.02
IXI=74
IYI=74
IDAY=MIN(1)
RADEG=IDAY
MENU=3
CALL INCOM(' ENTER 3 PICTURE START TIMES - HHMMSS ', 37, 1, 1, RMIN, REN
*, 3, 1)
IF(MENU.LE.0) GO TO 999
ITIME=RMIN(1)
PIII(1)=ITIME+ITIME
ITIME=RMIN(2)
PIII(2)=ITIME+ITIME
ITIME=RMIN(3)
PIII(3)=ITIME+ITIME
DO 100 I=1, 3
IMI(1)=0.0
IMA(1)=0.0
100 CONTINUE
MENU=4
CALL INCOM(' ENTER FIRST ORBIT POSITION: HHMMSS, X, Y, Z (KM) ', 52, 1, 1, RMIN, MENU, 3, 1)
IF(MENU.LE.0) GO TO 999
ITIME=RMIN(1)
1=ITIME(1)
R1(1)=RMIN(2)
R1(2)=RMIN(3)
A6INT/AOIPS

K(1:3)=RMIN(4)

IF(MENU=4)

CALL INCOM(' ENTER SECOND ORBIT POSITION: T(mm:ss), X, Y, Z (km)
+3.1.1, RMIN, MENU, 3, 1')

IF(MENU.LE.0) GO TO 999

TIME=RMIN(1)
T2=TIME(TIME)
K(1)=RMIN(2)
K(2)=RMIN(3)
K(3)=RMIN(4)

CALL GASORB(R1,T1,R2,T2)

CONTINUE

IF(JPRI.EQ.0) GO TO 1000

WRITE(LP,761) NAV, INAV, 11R, IDAY

761 FORMAT(I10, 'NAV=', I5, 'I2=', 'I10, I152, 'YEAR=', I2, 'T77, 'DAY=', I4)


764 FORMAT(T2, 'ORBIT DATE=', I4, 'I2=', 'ORBIT DATE=', I4)

765 FORMAT(T2, 'POSITION TIME: TM=', F15.4)

766 FORMAT(T2, 'POSITION (km): T2, X1=', F15.3, 'T2, Y1=', F15.3, 'T2, Z1=', F15.3)

767 FORMAT(T2, 'VELOCITIY: T2, D1=', F15.3, 'T2, D1=', F15.3, 'T2, D1=', F15.3)

768 FORMAT(T2, 'ANGLE (deg): T2, D1=', F15.3, 'T2, D1=', F15.3, 'T2, D1=', F15.3)

769 FORMAT(T2, 'PICTURE TIMES: I2, S2=', F25.4)

770 FORMAT(T2, 'I2, S2=', F25.4)

771 FORMAT(T2, 'S2=', F25.4)

772 FORMAT(T2, 'S2=', F25.4)

773 FORMAT(T2, 'S2=', F25.4)

774 FORMAT(T2, 'S2=', F25.4)

775 CALL REQUEST(RAD50('AFSF2'))
SUBROUTINE BCTOPF(X, Y, Z, IDIR)

COMMON/NAVCOM/NAV, INAV, IYR, IDAY, TOTLIN, DEGLIN, TOTIEL, DEGELE, PICLEI

*N, PICELE, TMPSCF, 10YR, 10DAY, TM, R1X, R1Y, R1Z, R1DX, R1DY, R1DZ, PITCH, ROLL

*L, YAW, PTIM(3), TMN(3), TMK(3), NLCOEF(2), NRCOEF(2), SCLLO(2), SCLL1(2),

CALL COEF(11, 2), SCLR0(2), SCLR1(2), ERCOEF(11, 2), NASCEF, SCLAS0, SCLAS1,

C PT(3) = YAW, PT(2) = ROLL, PT(1) = PITCH

CALL UNIT(A)

CALL ROTATE(A, YAW, 3, 1)

CALL ROTATE(A, ROLL, 1, 1)

CALL ROTATE(A, PITCH, 2, 1)

IF(IDIR.EQ.2) GO TO 10

XT = X*A(1,1) + Y*A(1,2) + Z*A(1,3)

YT = X*A(2,1) + Y*A(2,2) + Z*A(2,3)

ZT = X*A(3,1) + Y*A(3,2) + Z*A(3,3)

X = XT

Y = YT

Z = ZT

RETURN

X = XT

Y = YT

RETURN

END
***** EATOST/MCIDAS *****

DELT,L AF,EATOST/MCIDAS
ELT007 RL1862 12/22-17:01:43-(0,)
000001  000  SUBROUTINE EATOST(PICTIM,LNS,IES,LA,LO,IUNK,INAV,BETA,BETDOT,AF)
000002  000     PTIME=PICTIM
000003  000           IF(IUNK.EQ.1) GO TO 1
000004  000           IF(IUNK.EQ.2) GO TO 2
000005  000           RETURN
000006  000  1     CONTINUE
000007  000     XLIN=LNS
000008  000     XLEL=IES
000009  000           CALL SEL(PTIME,XLIN,XLEL,XLAT,XLON)
000010  000     LA=ILALO(XLAT)
000011  000     LO=ILALO(XLON)
000012  000           RETURN
000013  000  2     CONTINUE
000014  000           XLAT=FLALO(LA)
000015  000           XLON=FLALO(LO)
000016  000           CALL SEL(PTIME,XLAT,XLON,XLIN,XLEL)
000017  000     LNS=XLIN
000018  000     IES=XLEL
000019  000           RETURN
000020  000           END

END ELT.

@HDGE,P ***** EDGECOR/MCIDAS *****
***** EDGTC/MIADAS *****

BELT, L AF.EDGTC/MIADAS
ELTOG7 RL1662 12/22-17:01:44-00,
000001 000 SUBROUTINE EDGTC(P1ME, ALIN, DELLIN, DELELE)
000002 000 COMMON/NAVCOM/NAV, INAV, IYR, IDAY, TOTLIN, DELLIN, TOTIEL, DEGELE, PICLI
000003 000 *N, PILE, TMSCL, I0YR, I0DAY, TM, RIX, R1Y, R1Z, R1X, R1Z, P1CH, R1L
000004 000 *L, YA, PTIM, TMX, TMX, MRCOE(2), HRRCOE(2), SCLLO(2), SCLL1(2),
000005 000 *ELCOEF(11,2), SCLLO(2), SCLL1(2), ERCOEF(11,2), NASCEF, SCLASO, SCLAS1,
000006 000 *ELEMN, IELEMX, ASCOE(16)
000007 000 DATA A/52.07/, LC/1235/
000008 000 ISCAN=1200-(IFIX(ALIN-1))/2
000009 000 TIME=PTIME+ISCAN*TMSCL
000010 000 DO 20 TC=2,3
000011 000 20 IF(TIMX(1C).LE.TIME.AND.TIMX(1C).LE.TIMX(1C)) GO TO 100
000012 000 DELLIN=0.0
000013 000 DELELE=0.0
000014 000 RETURN
000015 000 100 CONTINUE
000016 000 IX=IC-1
000017 000 XL=ISCAN*SCLL1(IX)+SCLLO(IX)
000018 000 CALL CNPS(EL, XL,ELCOEF(1,IX), MRCOE(IX))
000019 000 XR=ISCAN*SCLL1(1A)+SCLLO(1A)
000020 000 CALL CNPS(ER, XR, ERCOEF(1,1A), HRRCOE(1A))
000021 000 DELELE=(EL-ER)/2.0
000022 000 MSC=1200-(LC-1)/2
000023 000 Y=ISCAN-MSC
000024 000 XX=SQR(A**2-Y**2)
000025 000 DELTA=(ER-EL)/2.0
000026 000 RAD=Y**2-2*XX*DELTA-DELTA**2
000027 000 DELLIN=SQR(RAD)-Y
000028 000 RETURN
000029 000 ENDS

END ELT.
@WG6/8 ***** EDGFT/MIADAS *****
***** EDGFT/MCIDAS *****

BELT,L AF,EDGFT/MCIDAS
ELT007 RLIB62 12/22-17:01:44-(0,)
000001 000 SJDB EDGFT U3200
000002 000 SOPRITION +6,9,15,20
000003 000 $FORTRAN
000004 000 SUBROUTINE MAIN
000005 000 DIMENSION JWIN(11,10)
000006 000 DIMENSION DATI(601),WORK(231),COEF(21)
000007 000 DIMENSION ISCN(300),ISENR(300),IUL(300),IUR(300),NOUT(17)
000008 000 DIMENSION MIN(10)
000009 000 DIMENSION MOUT(44)
000010 000 COMMON/NVCOM/NAV,NAV,INAC,TR,DAY,DTGL1,DEGL1,TOTEL,DEGELE,PCL1
000011 000 *N,PICL,TRPSCL,ITQR,ITQR,ITQR,ITQR,ITQR,ITQR,ITQR,ITQR,ITQR,ITQR,ITQR
000012 000 *L,YAW,PTIM(1),TMA(3),TMA(3),NLCOF(2),NLCOF(2),NLCOF(2),NLCOF(2),NLCOF(2)
000013 000 *L,COEF(11,2),SCALR(2),SCLL(2),SCLL(2),SCLL(2),SCLL(2),SCLL(2)
000014 000 *S,JELEM,JELEM,JELEM,JELEM
000015 000 DATA TMNL/1440.0/,TMXL/0.0/,TMMN/1440.0/,TMMX/0.0/
000016 000 DATA LUN/10/,NECSV/-1/,NUMER/10/,NUMER/11/,JOUT/2/
000017 000 DATA MIN/460CCGCL,8*0/
000018 000 CALL I4Q(MIN)
000019 000 IP=11
000020 000 CALL ROM
000021 000 ENCODE(132,701,MOUT)PTIM
000022 000 701 FORMAT(5X,3F20.6)
000023 000 CALL TP(1,MOUT)
000024 000 CALL VARASG("WIN",LUN)
000025 000 CALL OFN(LUN)
000026 000 DO 500 IPAIR=1,2
000027 000 NL=0
000028 000 NR=0
000029 000 TMNL=1440.0
000030 000 TMXL=0.0
000031 000 TMMN=1440.0
000032 000 TMMX=0.0
000033 000 CALL SCRA(LUN,0)
000034 000 IK=0
000035 000 90 CONTINUE
000036 000 100 CONTINUE
000037 000 JSEC=IK/NUMER
000038 000 JVEC=MOD(IK,NUMER)+1
000039 000 IF(JSEC.NE.JSECV) CALL READV(LUN,110,JWIN)
000040 000 IF(JSEC.NE.JSECV) JSECV=JSEC
000041 000 IK=IK+1
000042 000 IF(JWIN(1,JVEC).LE.0) GO TO 400
000043 000 IH=MOD(JWIN(4,JVEC),10)
000044 000 IF(IH.GT.4) GO TO 100
000045 000 T1=FTIME(JWIN(2,JVEC))
000046 000 T2=FTIME(JWIN(3,JVEC))
000047 000 IF((T1,NE,PTIM(1),OR,T2,NE,PTIM(IPAIR+1))GO TO 100
000048 000 IF(KND(JWIN(4,JVEC)/100,1000),NE,100) GO TO 100
000049 000 IY=JWIN(5,JVEC)
000050 000 IY=JWIN(7,JVEC)
000051 000 IF(IY.IE.12) GO TO 100
000052 000 IX=JWIN(6,JVEC)
000053 000 IF(IX1.GE.PICL) GO TO 200
000054 000 NL=NL+1
000055 000 IF(NL.GT.300) GO TO 100
000056  000  ISCNL(NL)=1200-(1Y1-1)/2
000057  000  IUL(NL)=JWIN(9,JVEC)*((T2-T1)/60.0
000058  000  GO TO 100
000059  000  200  CONTINUE
000060  000  NR=NR+1
000061  000  IF(NR.GT.300) GO TO 100
000062  000  ISCNR(NR)=1200-(1Y1-1)/2
000063  000  IUR(NR)=JWIN(9,JVEC)*((T2-T1)/60.0
000064  000  GO TO 100
000065  000  400  CONTINUE
000066  000  IF(NL.LT.IP+1) GO TO 459
000067  000  DO 410 I=1,NL
000068  000  TS=PTIM(IPAIR+1)+ISCNL(I)*TMONTH
000069  000  TMNL=AMAX1(TMNL,TS)
000070  000  TMXL=AMAX1(TMXL,TS)
000071  000  DAT(I)=TSMAX(TMNL,TS)
000072  000  410  DAT(I+NL)=IUL(I)/100.0
000073  000  DAT(I+2*NL+1)=1.0
000074  000  CALL APCH(DATI,NL,IP,XD,X0,WORK,IER)
000075  000  IF(IER.LT.0) GO TO 990
000076  000  SCLR0(IPAIR)=XD
000077  000  SCLR1(IPAIR)=XD
000078  000  EPS=1.0E-4
000079  000  IOP=+1
000080  000  ETA=1.0E-3
000081  000  CALL APFS(WORK,IP,ieres,1op,eps,eta,ier)
000082  000  IF(IER.LT.0) GO TO 990
000083  000  NLCOEF(IPAIR)=IRES
000084  000  ENCODE(132,706,MOUT)*IRES
000085  000  CALL TG(MOUT)
000086  000  Ix=IRES+CIRES-1/2
000087  000  DO 420 I=1,IRES
000088  000  COEF(I)=WORK(I+Ix)
000089  000  ENCODE(132,705,MOUT)*COEF(I)
000090  000  705  FORMAT(2x,"COEF(”,12x,”)="",E20.9)
000091  000  CALL TG(MOUT)
000092  000  420  ELCOEF(I,IPAIR)=COEF(I)
000093  000  CONTINUE
000094  000  IF(NR.LT.IP+1) GO TO 490
000095  000  DO 460 I=1,NR
000096  000  TS=PTIM(IPAIR+1)+ISCNR(I)*TMONTH
000097  000  TMNR=AMAX1(TMNR,TS)
000098  000  TMXR=AMAX1(TMXR,TS)
000099  000  DAT(I)=ISCNR(I)
000100  000  460  DAT(I+NR)=IUR(I)/100.0
000101  000  DAT(I+2*NR+1)=1.0
000102  000  CALL APCH(DATI,NR,IP,XD,X0,WORK,IER)
000103  000  IF(IER.LT.0) GO TO 990
000104  000  SCLR0(IPAIR)=XD
000105  000  SCLR1(IPAIR)=XD
000106  000  EPS=1.0E-4
000107  000  IOP=+1
000108  000  ETA=1.0E-3
000109  000  CALL APFS(WORK,IP,ieres,1op,eps,eta,ier)
000110  000  IF(IER.LT.0) GO TO 990
000111  000  NLCOEF(IPAIR)=IRES
000112  000  ENCODE(132,706,MOUT)*IRES
***** EDGFT/MCIDAS *****
000113 000  706  FORMAT(2X, "DIMENSION=", I3)
000114 000  CALL TQ(MOUT)
000115 000  IX=IRES*(IRES-1)/2
000116 000  DO 470  I=1,IRES
000117 000  COEF(I)=WORK(I+IX)
000118 000  ENCODE(132,70S,MOUT),COEF(I)
000119 000  CALL TQ(MOUT)
000120 000 470  ERCOEF(I,IPAIR)=COEF(I)
000121 000  CONTINUE
000122 000  TMN(IPAIR+1)=AMAX1(TMNL,TMNR)
000123 000  TMX(IPAIR+1)=AMIN1(TMXL,TMXR)
000124 000 500  CONTINUE
000125 000  CALL WCOM
000126 000  CALL EMES3(3HFIN,0)
000127 000  RETURN
000128 000 990  CONTINUE
000129 000  CALL TQ(72H ERROR RETURN FROM APCH, APFS)
000130 000  RETURN
000131 000  END$  
000132 000  $FILEMA
000133 000  DELETE GCCGCC,GORP
000134 000  INCLUDE ATSSSP
000135 000  SCATALOG
000136 000  NAME=GCCGCC,5,R,Y,D
000137 000  TYPE=FG
000138 000  LIB=ATSLB,LL
000139 000  BEGIN
000140 000  SEQJ

END ELT.
$HDG,P ***** ERTOER *****
****** ERTOER/MEIDAS ******

BELT, L AF  ERTOER/MEIDAS
ELTO07 RLIB62 12/22-17:01:46-(00)
000001 000 SUBROUTINE ERTOER(XLAT,XLON,XE,YE,ZE,IDL)
000002 000 PI=3.14159265
000003 000 RDPDG=PI/180.0
000004 000 A=6378.15
000005 000 B=6356.77
000006 000 ESQ=(A-B)*(A+B)/A**2
000007 000 IF(IDK .EQ. 2) GO TO 10
000008 000 XLAT=XLAT*RDPDG
000009 000 XLON=XLON*RDPDG
000010 000 CLT=COS(XLT)
000011 000 SLT=SIN(XLT)
000012 000 CLN=COS(XLN)
000013 000 SLN=SIN(XLN)
000014 000 RR=A/SQRT(1.-ESQ*SLT**2)
000015 000 XE=CLT*CLN*RR
000016 000 YE=CLT*SLN*RR
000017 000 ZE=SLT*RR*(1.-ESQ)
000018 000 RETURN
000019 000 10 CONTINUE
000020 000 IF(((XE**2+YE**2+ZE**2) LT. B**2)) OR. ((XE**2+YE**2+ZE**2) OR. ((GE**2+YE**2+ZE**2)) GO TO 15
000021 000 XLON=ATAN2(YE,XE)/RDPDG
000022 000 XLAT=ATAN(ZE/((1.-ESQ)*SQRT(XE**2+YE**2+ZE**2)))/RDPDG
000023 000 RETURN
000024 000 XLAT=100.
000025 000 XLON=200.
000026 000 RETURN
000027 000 ENDS

END ELT.

DHG,P ****** ERTOST ******
SUBROUTINE ERTOST(XE, YE, ZE, X, Y, Z, IDIR, TIME)
PARAMETER LIST
XE, YE, ZE = EARTH COORDS OF LANDMARK
X, Y, Z = POINTING VEC IN INERTIAL COORDS TO LANDMARK
IF IDIR = 1, EARTH TO POINTING
IF IDIR = 2, POINTING TO EARTH
TIME = HHMMSS OF CURRENT POINTING VECTOR
COMMON/NAVCON/NAVN, INAV, IYR, IDAY, TH, RH, R1X, R1Y, R1Z, R1D, R1D, R1D, PITCH, ROL
*LEN, PI, ELE, TPSCL, TPSCL, TPSCL, NLCOEFF, R, NCOEFF, SCLLO, SCLLO, SCLLO, SCLLO,
*LEN, ELE, TPSCL, NCOEFF, SCLLO, SCLLO, SCLLO, SCLLO
A = 6378.15
B = 6356.77
R1 = 59477026
R2 = 95649356
R3 = 250684473
PI = 14159265
RDPDG = PI / 180.0
R = (R1 + R2 + R3 * TIME)
R = MOD(R1, 360.0) * RDPDG
CALL ORBIT(XS, YS, ZS, TIME)
CR = COS(R)
SR = SIN(R)
IF (IDIR = 2) GO TO 10
X1 = CR * XE + SR * YE
Y1 = SR * XE + CR * YE
Z1 = ZE
X = X1 - XS
Y = Y1 - YS
Z = Z1 - ZS
CALL NMLIZ(X, Y, Z, RNRNM) RETURN
CONTINUE
F = B**2 / (A**2)
AG = F + (1.0 - F)*Z**2
BG = 2.0*SQRT((X*X + Y*Y)*F + Z**2)
CG = (X*X + Y*Y)*F + Z**2 - B**2
RA = B**2 - 4.0*A*G
IF (RA < 0) GO TO 15
S = (-EQ + SQRT(RA)) / (2.0 * AQ)
X = X + S*X
Y = Y + S*Y
Z = Z + S*Z
X = CR * X + SR * Y
Y = SR * X + CR * Y
Z = Z
RETURN
WRITE(*, 1000)
1000 FORMAT(1X, 'THE SUBROUTINE ERTOST (IDIR=2) HAS RECEIVED BAD DATA.')
RETURN
ENDS
***** ES/MCIDAS *****

DATA

SUBROUTINE ES(PTIME, XLAT, XLONG, XLIN, XELE)

* COMMON/NAVCOM/NAVY, INAV, IVK, IDAT, TOTLIN, DFLIN, TOTIEL, DEGELE, PICLI

* L, YAV, PTIM(3), TMH(3), TMX(3), NLCOEF(2), HHCOEF(2), SCL0(2), SCL1(2),

* ELCOEF(11,2), SCLR(2), SCLR(2), ERCEOFC(11,2), NASF, SCLAS0, SCLAS1,

* IELEMN, IIELEMX, ASCEF(16)

DATA INIT/0/

ALON=XLONG

ALAT=XLAT

TIME=PTIME

IDIR=1

ALINSV=0

AELESV=0

DO 100 II=1,5

CALL ERTOER(ALAT, ALON, X1ER, X2ER, X3ER, IDIR)

CALL ERTOST(X1ER, X2ER, X3ER, X1, X2, X3, IDIR, TIME)

CALL STGTV(X1, X2, X3, IDIR, TIME)

C BC SAME AS LV HERE.

CALL GETOFIX(X1, X2, X3, IDIR)

CALL PFTOC(ALIN, AELE, X1, X2, X3, IDIR, INIT)

CALL EDGORG(PTIME, ALIN, DELLIN, DELELE)

AELEC=AELE+DELELE

ALINC=ALIN+DELLIN

IF(ALINC/ALINSV, LE, 0.5) GO TO 150

ALINSV=ALINC

AELESV=AELEC

ISCAN=1200-(IFIX(ALIN-1))/2

TIME=PTIME+ISCAN+TMSCL

100 CONTINUE

150 CONTINUE

XLIN=ALINC

XELE=AELEC

RETURN

END ELT.

END ERT007

%%%% % ***** FLATO *****
**** FLALO/MCIDAS ****

SELT,L AF,FLALO/MCIDAS
ELT007 RLIB62 12/22-17:01:49-(0,)

000001  000  FUNCTION FLALO(N)
000002  000  IF (M .LT. 0) GO TO 1
000003  000  N=M
000004  000  X=1.0
000005  000  GO TO 2
000006  000  1
000007  000  N=-M
000008  000  2
000009  000  FLALO=FLOAT(N/10000)+FLOAT(MOD(N/100,100))/60.+FLOAT(MOD(N,100))/
000010  000  * 3600.*
000011  000  FLALO=FLALO*X
000012  000  RETURN

END ELT*

@HKG  P ***** FLIP *****
*** FLIP/MCIDA9 ***

DELT, L, A, FLIP/MCIDA9
ELT007 RL1D62 12/22/17:01:50-(0)
000001 000 SUBROUTINE FLIP(A,B,I,N,ALTRET)
000002 000 C
000003 000 C THIS ROUTINE IS CALLED ONLY BY SUBROUTINE INVERT AND IS USED TO
000004 000 C PERFORM AN ELEMENTARY ROW OPERATION ON MATRICES A AND B.
000005 000 C
000006 000 C LOGICAL ALTRET
000007 000 DIMENSION A(N,N), B(N,N)
000008 000 ALTRET=.FALSE.*
000009 000 LIM=1+1
000010 000 DO 2 L=LIM,N
000011 000 IF(AABS(A(L,1))<LT.1.0E-32)GOTO 2
000012 000 DO 1 M=1,N
000013 000 A(I,M) = A(I,M) + A(L,K)
000014 000 B(I,M) = B(I,M) + B(L,M)
000015 000 RETURN
000016 000 CONTINUE
000017 000 ALTRET=.TRUE.*
000018 000 RETURN
000019 000 ENDS

END ELT.
0HDG,P *** FTIME/MCIDA9 ***
***** FTIME/MCIDAS *****

RELT,L AF.FTIME/MCIDAS
ELT007 RL1B62 12/22-17:01:51-00,
000001  000  FUNCTION FTIME(ITIME)
000002  000  IH=ITIME/10000
000003  000  HRS=IH*60.
000004  000  AMS=FLOAT((ITIME-(ITIME/10000)*10000)/100)
000005  000  SECS=FLOAT(MOD(ITIME,100))/60.
000006  000  FTIME=HRS+AMS+SECS
000007  000  RETURN
000008  000  ENDS

END ELT.
@HSG,P ***** GASORG/MCIDAS *****
*** GASORB/MCIDAS ***

DELTA,L AF,GASORB/MCIDAS
ELT007 RL662 12/22-17:01:51-(0,)

000001 000 SUBROUTINE GASORB(R1,T1,R2,T2)
000002 000 REAL KE,L,M,N
000003 000 DIMENSION R(3),R2(3)
000004 000 COMMON/NAVCON/NAV,NAV,IVR,IDR,TOTL,DEGL,I,TOTI,DEGE,L,PIL
000005 000 *N,FICELE,TMPSL,1OYR,1OYD,TM,1RX,R1Y,R1Z,1RX,1RY,1RZ,1RX,1RY,1RZ,PITCH,ROL
000006 000 *L,YAN,PTIM(3),TMN(3),TMX(3),NLCOEF(2),NRCEOF(2),SCLLU(2),SCLLI(2),
000007 000 *ELCOEF(11,2),SCLRR(2),SCLRR(2),ERCOEF(11,2),NASCE,F,SCLASO,SCLASL,
000008 000 *IELEN,M,IELEN,M,ASCEOF(16)
000009 000 DATA R(1)=R1(I)/RE
000010 000 R(2)=R2(I)/RE
000011 000 DO 5 I=1,3
000012 000 R1(I)=R1(I)/RE
000013 000 R2(I)=R2(I)/RE
000014 000 CONTINUE
000015 000 R1X=R1(1)
000016 000 R1Y=R1(2)
000017 000 R1Z=R1(3)
000018 000 R2NORM=SGRT(R1(1)**2+R1(2)**2+R1(3)**2)
000019 000 R2NORM=SGRT(R2(1)**2+R2(2)**2+R2(3)**2)
000020 000 TAU=KE*(T2-T1)
000021 000 R1R2N=R1NORM*R2NORM
000022 000 COSANM=(R1(1)*R2(1)+R1(2)*R2(2)+R1(3)*R2(3))/R1R2NM
000023 000 CSFAN=SGRT(1+COSANM)**2
000024 000 L=(R1NORM*R2NORM)/(4*SGRT(R1R2NM)*CSFAN)**5
000025 000 M=(MU*TAU**2)/(2*SGRT(R1R2NM)*CSFAN)**2
000026 000 YU=1
000027 000 10 YL=YU
000028 000 XL=M/YL**2**L
000029 000 CHFEN=1-2*XL
000030 000 SHFEN=SGRT(4*XL*(1-XL))
000031 000 ECANOM=2*ATAN2(SHFEN,CHFEN)
000032 000 XU=(ECANOM-SIN(ECANOM))/SHFEN**3
000033 000 YU=1/XU*(L+XL)
000034 000 IF(AES(YU-YL).GT.1.0066)GOTO10
000035 000 YL=YU
000036 000 A=(TAU*SGRT(MU))/(2*YL*SGRT(R1R2NM)*CSFAN*SHFEN)**2
000037 000 F=1-A/NORM*(1-COS(ECANOM))
000038 000 G=TAU-A**1.5*SGRT(MU)*(ECANOM-SIN(ECANOM))
000039 000 RDX=R(2,1)-F*R(1,1)
000040 000 R1Y=(R(2,2)-F*R(1,2))/6
000041 000 R1Z=(R(2,3)-F*R(1,3))/6
000042 000 RETURN
000043 000 END

END ELT.
SHDG,P ***** INVERT/MCIDAS *****
**INVERT/MCIDAS**

```
DELT, LAF INVERT/MCIDAS
ELT007 RL1662 12/22/71:01:52-(O,)

000001 000 SUBROUTINE INVERT(AA,B,N,ALTRET)
000002 000
000003 000 C THIS ROUTINE RETURNS IN B THE INVERSE OF THE N-DIMENSIONAL MATRIX AA.
000004 000 C IF AA IS SINGULAR, A RETURN IS TAKEN THROUGH LABEL $%.
000005 000 C
000006 000 C LOGICAL ALTRET, LOGTMP
000007 000 DIMENSION AA(N,N),B(N,N),A(3,3)
000008 000 DATA TV/1.0E35/
000009 000 ALTRET=.FALSE.
000010 000 DO 1 I=1,N
000011 000 DO 1 J=1,N
000012 000 1 A(I,J)=AA(I,J)
000013 000 DO 2 I=1,N
000014 000 DO 2 J=1,N
000015 000 B(I,J)=0
000016 000 2 IF(I.EQ.J)B(I,J)=1.0
000017 000 LIM=N-1
000018 000 DO 4 I=1,LIM
000019 000 IF(ABS(A(I,I))&LT;1.0E-30) CALL FLIP(A,B,I,N,ALTRET)
000020 000 LOGTMP=.FALSE.
000021 000 IF(LOGTMP) GO TO 101
000022 000 LIM2=I+1
000023 000 DO 4 J=LIM2,N
000024 000 4 IF(TV*ABS(A(I,J))&LE;ABS(A(J,I))) ALTRET=.TRUE.
000025 000 IF(TV*ABS(A(I,J))&LE;ABS(A(J,I))) RETURN
000026 000 FACTOR=A(J,I)/A(I,I)
000027 000 DO 3 K=LIM2,N
000028 000 3 A(J,K)=A(J,K)-FACTOR*A(I,K)
000029 000 DO 4 K=1,N
000030 000 4 B(J,K)=B(J,K)-FACTOR*B(I,K)
000031 000 DO 5 I=N,2,-1
000032 000 5 LIM3=I-1
000033 000 DO 5 J=LIM3,1,-1
000034 000 5 IF(TV*ABS(A(I,J))&LE;ABS(A(J,I))) ALTRET=.TRUE.
000035 000 IF(TV*ABS(A(I,J))&LE;ABS(A(J,I))) RETURN
000036 000 FACTOR=A(J,I)/A(I,I)
000037 000 DO 6 K=1,N
000038 000 6 B(J,K)=B(J,K)-FACTOR*B(I,K)
000039 000 DO 6 I=N,1,-1
000040 000 6 FACTOR=A(I,I)
000041 000 DO 6 J=1,N
000042 000 6 IF(TV*ABS(FACTOR)&LE;ABS(B(I,J))) ALTRET=.TRUE.
000043 000 IF(TV*ABS(FACTOR)&LE;ABS(B(I,J))) RETURN
000044 000 6 B(I,J)=B(I,J)/FACTOR
000045 000 6 RETURN
000046 000 6 ALTRET=.TRUE.
000047 000 101 RETURN
000048 000 101 END
```

END ELT.

@MDGP ***** LATSF/MCIDAS *****
PROGRAM TO READ IMAGES FROM ATS-6 TAPES.

INTEGER*4 ITIMX
LOGICAL INVIS,IFIR,OPTION
DIMENSION MIN(10),MAIN(10),ARRAY(5600)

DIMENSION TOTL(256),LROW(456)

DIMENSION MUAY(12)

COMMON/HNAVCOM/NAV,INAV,1YR,1DAY,TOTL,TOTL1,DEGLN,DEGL1,DEGE1,PICL1

PICLE1,LPSCL,LU1K,1DAY1,T4,R4A,R41,R12,R1A,K12,K1A,K1C,PITCH,PITCH1

R4,THAT,TH113,TH113,TH113,M113,THICF2,THICF2,THICF2,THICF2,THICF2

E7,THCFCF112,THCFCF112,THCFCF112,THCFCF112,THCFCF112,THCFCF112

IELEIM,IELEMX,ASCORE(10)

DATA MUDAY/0,31,30,30,120,120,151,181,212,243,273,304,334/

DATA LUN/4,16,6,1CR/5/

DATA 11/5/,NELEIS/512/

DATA RNINS/512/

C PICK UP INPUT PARAMETERS

RNIN=5600

MENU=1

CALL INCOM(' ENTER PICTURE TIME (HHMMSS) ',28+1,1,FMN,MENU+3,1)

IF(MENU=LE.0) GO TO 999

11IMX=MIN(1)

PT=TIME(11IMX)

DO 5 I=1,3

IF(PT.EQ.PTIM(1)) GO TO 6

WRITE(11,710)PTIM

710 FORMAT(1X,'NAVCOM NOT INITIALIZED FOR ',L1)

GO TO 999

CONTINUE

IFIC=1

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 AREA (1-7)',21,5,1,MIN,MENU,1,1)

IF(MENU.LE.0) GO TO 999

AAREA=MIN(1)

LNRVS=LAREA+5+(IFIC-1)*7

LNRK=LNRVS+21

MENU=1

CALL INCOM(' ENTER AREA UNIT (0,1)',22,7,1,MIN,MENU,1,1)

IF(MENU.LE.0) GO TO 999

AUNIT=M1N(1)

IFDF=1

IFVIS=.TRUE.

IFIR=.TRUE.

C COMPUTE UPPER LEFT COORDINATES

JLINES=LINE

JUELE=16C

JUMS=1260-(JLINES-1)/2

JULINE=JLINES+(JLINES-1)*IFDF
LATSF/AOIPS

ILNW=100-1ULINE-1)/2

ESTABLISH AREA DIRECTORY

DO 10 1=1,256

10 DDIR(1)=0

DDIR(1)=IAT

DDIR(2)=S6

DDIR(3)=V

DDIR(4)=1S

DDIR(5)=E

DDIR(6)=W

DDIR(7)=1

DDIR(8)=

DDIR(9)=IAT

DDIR(10)=S6

DDIR(11)=

DDIR(12)=E

DDIR(13)=H

DDIR(14)=T

DDIR(15)=

DDIR(16)=

DO 12 1=12,1,-1

12 IF (IDAY.GT. MODAY(1)) GO TO 13

CONTINUE

MON=1:

DDIR(19)=1IR*100+MON

DDIR(20)=1IR+MODAT(MON)*100+IFIX(E+TIM(IPIC)/60.)

IK=TIM(IPIC)

IS=TIM(IPIC)*60.-1M*60.

IM=MOD(IS,60)

IM=MOD(IM+60)

DDIR(21)=1M+100+IS

DDIR(24)=

DDIR(25)=

DDIR(26)=IPIC

DDIR(27)=EF

DDIR(28)=IC

DDIR(31)=

DDIR(32)=LNRVIS

DDIR(33)=512

DDIR(34)=513

DDIR(37)=512

DDIR(38)=512

DDIR(39)=1UBE

DDIR(40)=1ULINE

DDIR(41)=

DDIR(42)=

DDIR(43)=

DDIR(44)=

DDIR(45)=

DDIR(46)=

DDIR(47)=

DDIR(48)=

DDIR(51)=1000*(IMJIN+1)

DDIR(52)=

NWD=(MELES+1)/2

CALL OPER(LRNRVIS,ULINE+1,NWD)

CALL OPER(LRNIR,ALIN+1,NWD)

CALL ULINAT(LRNVIS,ULIR,256)

DDIR(32)=LNRIR

DDIR(4)=

DDIR(4)=1IR

CALL LNRWR(LRIR,DIR,256)

C

C GENERATE OFFSET TABLE

CALL GEN0FF(IME,MELES)

CALL ASINLUN(LUN,'%',IMUNIT)

C

C ADVANCE TO FIRST SCAN TO INPUT
071 WRITE(11,711)
    711 FORMAT(' TAPE SEARCH STARTS')
    CALL IOIPM(LUO,*1,ISTAT)
    UD 510 I=1,1200
    CALL IOIPM(LU0,IAKRAY,NM,LE,ISTAT)
    CALL CCHAIN(195,IAKRAY,IAROW)
    INMS=IAKROW(195)
    IF(INMS.LE.LUNS) GO TO 511
510 CONTINUE
    GO TO 922
711 CALL IOIPS(LUO,-1,ISTAT)
    WRITE(11,712)
    712 FORMAT(' IMAGE LOAD SIARIS')
    C START MAIN LOOP TO READ DATA IN AND STORE
    C READ A SCAN
601 CALL IOIPM(LU0,IAKRAY,NM,LE,ISTAT)
    CALL CCHAIN(195,IAKRAY,IAROW)
    INMS=IAKROW(195)
    IF(INMS.LT.1LUNS) GO TO 690
    C MAP SCAN NUMBER TO AREA ROW
    L1=(1200-INMS)*24+3-2
    L2=(1200-INMS)*24+3-1
    LA=L1-1LUNE
    LA2=LA-L1LUNE
    LR1=MOD(LA1,1BDF)
    LR2=MOD(LA2,1BDF)
    IF(LR1.NE.0 .AND. LR2.NE.0) GO TO 601
    IRWO1=LA1/1BDF
    IRWO2=LA2/1BDF
    ISD=MOD(INMS,2)
    ISEC=KS1KROW1
    ISEC2=NSI1KROW1
    C PICK OUT LINE SEGMENT
    C FIRST VISIBLE LINE
    IF(IFVIS.AND.LR1.EQ.0) CALL LINGRB(IARRAY(3745),ISD,1UELE,NELES,1BDF,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,1UELE,NELES,1BDF,IAROW)
    IF(IFVIS.AND.LR2.EQ.0) CALL WRITE(LRNVIS,IAROW,0)
    C INFRARED
    IF(IFIR) CALL LINGRB(IARRAY(195),ISD,1UELE,NELES,1BDF,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
722 CONTINUE
    CALL TABLE(LRNVIS)
    CALL TABLE(LRNIR)
799 CONTINUE
    CALL TABLE(RAD50('AISF2'))
END
***** LATSF/MCIDAS *****

DENSITY AF LATSF/MCIDAS

000001 000 SJUB LATSF US200
000002 000 $OPTION .6,9,13,20
000003 000 $FORTTRAN
000004 000 SUBROUTINE MAIN
000005 000 C PROGRAM TO READ IMAGES FROM ATS-6 TAPES.
000006 000 LOGICAL IFVIS,IFIR,OPTION
000007 000 DIMENSION MIN(12),IARRAY(3732),IAROW(672)
000008 000 DATA LUN/10/,LP/0/,ICR/5/
000009 000 DATA MIN/0/,LDATSF,10*0/,NWA/3732/
000010 000 C PICK UP INPUT PARAMETERS
000011 000 CALL IQR(MIN)
000012 000 C INPUT: SSSYDDD HHHHSS AREA LINE ELEMENT
000013 000 IDAY=MIN(1)
000014 000 ITIME=MIN(2)
000015 000 IAREA=MIN(3)
000016 000 ILINE=MIN(4)
000017 000 JILE=MIN(5)
000018 000 IBDF=1
000019 000 IREEL=0
000020 000 KEY=3H
000021 000 C COMPUTE DERIVED PARAMETERS
000022 000 ISS=IDAY/100000
000023 000 IFILE=1
000024 000 JVSS=(ISS/2)+2
000025 000 JRSS=JVSS+1
000026 000 IFVIS=ISS.EQ.IRSS.OR.IAREA.LT.9
000027 000 IFIR=ISS.EQ.IRSS.OR.IAREA.LT.9
000028 000 IRAREA=IAREA
000029 000 IF(IFIR.AND.IFVIS) IRAREA=IAREA+8
000030 000 C PICK UP AREA SIZE
000031 000 CALL HOMBIG(IAREA,NLINS,NELES)
000032 000 IF(NELES.GT.672) GO TO 922
000033 000 C COMPUTE UPPER LEFT COORDINATES
000034 000 IULINE=ILINE
000035 000 JUZE=JILE
000036 000 IF(OPTION.EQ.3H) IULINE=ILINE-(NLINS/2)*IBDF
000037 000 IF(OPTION.EQ.3H) JUZE=JUZE-(NELES/2)*IBDF
000038 000 IUMS=1200-(JUZE-1)/2
000039 000 ILINE=ILINE+(NLINS-1)*IBDF
000040 000 ILMS=1200-(ILINE-1)/2
000041 000 C CREATE AREA FOR COMBINED LOAD
000042 000 IF(IFIR.AND.IFVIS) CALL AREA8(IAREA+B,NLINS,NELES)
000043 000 C ESTABLISH AREA DIRECTORY
000044 000 IG=0
000045 000 CALL ENAREA(IAREA,IREEL,IDAY,ITIME,ILINE,JUZE,IBDF,IG)
000046 000 IRREEL=ISS+100000+MOD(IREEL,100000)
000047 000 IF(IREEL.EQ.0) IRREEL=0
000048 000 IDAY=ISS+100000+MOD(IDAY,100000)
000049 000 IF(IFVIS.AND.IFIR) CALL ENAREA(IAREA+B,IRREEL,IDAY,ITIME,ILINE,1
000050 000 *UELL,IBDF,IG)
000051 000 C GENERATE OFFSET TABLE
000052 000 CALL GENOFF(JUZE,NELES)
000053 000 IRR=MAXO(1,MOD(IREEL,10000))
000054 000 CALL GIAP(14,IRR,LUN)
000055 000 C ADVANCE TO FILE
**** LATSF/MCIDAS ****

000056 000  IFFILE=FILE-1
000057 000  IF(FILE1,LT,1)GO TO 502
000058 000  GO TO 922
000059 000  502 CONTINUE
000060 000  C  ADVANCE TO FIRST SCAN TO INPUT
000061 000  CALL READW(LUN,NWA,ARRAY)
000062 000  CALL READW(LUN,NWA,ARRAY)
000063 000  DO 510 I=1,1200
000064 000  CALL READW(LUN,NWA,ARRAY)
000065 000  CALL CRKTHR(195,ARRAY,ARRAY)
000066 000  IAMS=IAROW(195)
000067 000  IF(INMS.LE.IUMS) GO TO 511
000068 000  510 CONTINUE
000069 000  GO TO 922
000070 000  511 CALL BSR(LUN)
000071 000  C  START MAIN LOOP TO READ DATA IN AND STORE
000072 000  C  READ A SCAN
000073 000  601 CALL READW(LUN,NWA,ARRAY)
000074 000  CALL CRKTHR(195,ARRAY,ARRAY)
000075 000  IAMS=IAROW(195)
000076 000  C  MAP SCAN NUMBER TO AREA ROW
000077 000  L1=(1200-INMS)*2*(3-2)
000078 000  L2=(1200-INMS)*2*(3-1)
000079 000  LA1=L1+IULINE
000080 000  LA2=L2+IULINE
000081 000  LR1=MOD(LA1,IBDF)
000082 000  LR2=MOD(LA2,IBDF)
000083 000  IF(LR1,NE,0 .AND. LR2,NE,0) GO TO 601
000084 000  IROW1=LA1/IBDF
000085 000  IROW2=LA2/IBDF
000086 000  NS=NSEC1(NELES)
000087 000  ISD=MOD(INMS,2)
000088 000  ISEC1=NS*IROW1
000089 000  ISEC2=NS*IROW2
000090 000  C  PICK OUT LINE SEGMENT
000091 000  C  FIRST VISIBLE LINE
000092 000  IF(IFVIS .AND. LR1,EO,0) CALL LINGRB(IARRAY(2497),ISD,INUEL,NELES,IBDF,IAROW)
000093 000  * ISD,IAROW
000094 000  IF(IFVIS .AND. LR1,EO,0) CALL WRITA(IAREA,ISEC1,NS+112,IAROW)
000095 000  C  SECOND VISIBLE LINE
000096 000  IF(IFVIS .AND. LR2,EO,0) CALL LINGRB(IARRAY(1297),ISD,INUEL,NELES,IBDF)
000097 000  * ISD,NELES,IBDF
000098 000  IF(IFVIS .AND. LR2,EO,0) CALL WRITA(IAREA,ISEC2,NS+112,IAROW)
000099 000  C  INFRARED
000100 000  IF(IFIR) CALL LINGRB(IARRAY(97),ISD,INUEL,NELES,IBDF,IAROW)
000101 000  IF(IFIR) CALL LINGRB(IARRAY(97),ISD,INUEL,NELES,IBDF,IAROW)
000102 000  IF(IFIR) CALL WRITA(IAREA,ISEC1,NS+112,IAROW)
000103 000  IF(IFIR) CALL WRITA(IAREA,ISEC2,NS+112,IAROW)
000104 000  GO TO 601
000105 000  690 CONTINUE
000106 000  CALL MARKOK(IAREA)
000107 000  CALL MARKOK(IAREA)
000108 000  CALL REW(LUN)
000109 000  RETURN
000110 000  910 CONTINUE
000111 000  912 CONTINUE
000112 000  913 CONTINUE
CALL EMESS(3HREQ,0)
RETURN
WRITE(LP,701)
FORMAT(1X,"E-O-F TERMINATES LOAD")
CALL MARKOK(1AREA)
CALL MARKOK(1RARLE)
CALL REW(LUN)
RETURN
922 CONTINUE
CALL EMESS(3HREQ,0)
RETURN
END
SUBROUTINE LINGRP(INDATA,SDIR,ELE,NELES,BDF,OUTDAT)
IMPLICIT INTEGER(A-Z)
DIMENSION INDATA(1),OUTDAT(1)
DIMENSION TDAT(2406),DLELE(672)
COMMON/OFFSET/DLELE
DATA WOFF/16,NL/2406/
IF(EDF,NE,x) CALL EMESS(3HREQ,BDF)
IF(EDF,NE,x) CALL EXIT
102=ELE-1
CALL CRKATS(NE,INDATA,TDAT)
IF(SDIR, REQ,1) GO TO 100
C SHIFT EVEN SCANS
DO 50 I=1,NELES
DE=I+102+DLELE(I)+WOFF
OUTDAT(I)=TDAT(DE)
50 CONTINUE
CALL PACK(NELES,OUTDAT,TDAT)
RETURN
100 CONTINUE
C LOAD ODD SCANS
DO 150 I=1,NELES
OUTDAT(I)=TDAT(I+102+WOFF)
150 CONTINUE
CALL PACK(NELES,OUTDAT,TDAT)
RETURN
END
SUBROUTINE GENOFF(JUELE,NELES)
COMMON/NAVCON/NAVN,INAV,INR,NDAY,TOTLIN,DEGLIN,TOTIEL,DEGELE,PICL
/NL,PIELE,TMPSCL,IOYP,IODAY,TRX,R1Y,R1Z,R1D,R1D,RP1P,RO1P,RO1Z
/L1,TTM,TMM,TMX,TM(3),NLCOEF(2),NRCEO(2),SCLD(2),SCLL(2),
/ELCOEF(11,2),SCLR1(2),SCLR2(2),E2ROF(11,2),NASCE,ASCASO,E32AS1,
/IJELEMN,IELEMX,ASCE,16)
COMMON/OFFSET/IDLELE(672)
DATA Nomeff/8/
CALL ROM
NEND=JUELE+NELES-1
DO 100 I=1,NELES
IDLELE(I)=NOMOF
JUELE=JUELE+1
100 IDLELE(1)=0.5
100 CONTINUE
CALL CNPS(Y,T,ASCAS,NASCE)
RETURN
<table>
<thead>
<tr>
<th>Line</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>000170</td>
<td>000</td>
</tr>
<tr>
<td>000171</td>
<td>000</td>
</tr>
<tr>
<td>000172</td>
<td>000</td>
</tr>
<tr>
<td>000173</td>
<td>000</td>
</tr>
<tr>
<td>000174</td>
<td>000</td>
</tr>
<tr>
<td>000175</td>
<td>000</td>
</tr>
<tr>
<td>000176</td>
<td>000</td>
</tr>
<tr>
<td>000177</td>
<td>000</td>
</tr>
<tr>
<td>000178</td>
<td>000</td>
</tr>
<tr>
<td>000179</td>
<td>000</td>
</tr>
</tbody>
</table>

END ELT

*HDG,F P***** LDATS,F *****
***** LS/MCIDAS *****

DELT, L AF = LS/MCIDAS
FMT07 RLIB62 12/22-17:01:56-(0,)

000001 000 000 SUBROUTINE LS(X,Y,VAL,DD,DIR)
000002 000 000 C THIS ROUTINE PERFORMS AN ARMijo LINE SEARCH FROM THE POINT "X"
000003 000 000 C IN THE DIRECTION "DIR" AND RETURNS THE SELECTED POINT IN "Y"
000004 000 000 C AND THE OBJECTIVE FUNCTION VALUE S(Y) IN "VAL". ON CALL, "DD"
000005 000 000 C IS THE UNNORMALIZED DIRECTIONAL DERIVATIVE <GRAD(S(X)),DIR>.
000006 000 000 C THIS LINE SEARCH ROUTINE RETURNS IN Y THE POINT X+2**(-N)*DIR,
000007 000 000 C WHERE N IS THE LEAST NONNEGATIVE INTEGER SUCH THAT
000008 000 000 C -S(2**(-N)*DIR) REPRESENTS AT LEAST 40% OF THE FUNCTIONAL DROP.
000009 000 000 C IN THE LINEARIZATION OF S AT X IN MOVING FROM X TO X+2**(-N)*DIR.
000010 000 000 C
000011 000 000 C
000012 000 000 DIMENSION X(3),Y(3),DIR(3)
000013 000 000 DATA FAC,WALK/5.0E-1,1.0E-5/
000014 000 000 RLM=1.0
000015 000 000 TSTVAL=-4*RLAM*ABS(DD)
000016 000 000 OLDVAL=S(X)
000017 000 000 1 IF(RLAM. LT. WALK) RETURN
000018 000 000 DO 2 I=1,3
000019 000 000 2 Y(I)=X(I)+RLAM*DIR(I)
000020 000 000 VAL=S(Y)
000021 000 000 IF(OLDVAL-VAL. GE. TSTVAL) RETURN
000022 000 000 TSTVAL=FAC*TSTVAL
000023 000 000 RLAM=FAC*RLAM
000024 000 000 GOTO 1
000025 000 000 ENDS

END ELT.

ENDG,P ***** MINIZ/MCIDAS *****
***** MINMIZ/MCIDAS *****

BELT.L AF.MINMIZ/MCIDAS

ELTQ7 RLIB62 12/22-17:01:56-(O,)

000001 000 SUBROUTINE MINMIZ(PMIN,POUT,GNORM,VAL,ITN)

000002 000 C

000003 000 C THIS ROUTINE PERFORMS A MODIFIED NEWTON METHOD MINIMIZATION. IT

000004 000 C BEGINS AT THE POINT "PMIN" AND RETURNS IN "POUT" THE POINT

000005 000 C SELECTED AS THE OPTIMAL POINT, IN "GNORM" THE NORM OF THE GRADIENT

000006 000 C OF S AT "POUT", AND IN "VAL" THE VALUE OF THE OBJECTIVE FUNCTION

000007 000 C S AT "POUT". A POINT X(K+1) IS DEEMED OPTIMAL WHEN

000008 000 C ABS(S(X(K+1))-S(X(K)))<=(10**-10)*ABS(S(X(K))))

000009 000 C

000010 000 C LOGICAL ALTRET

000011 000 DIMENSION PMIN(3),POUT(3)

000012 000 DIMENSION HESS(3,3),GRAD(3),PT(3),DIR(3)

000013 000 DATA CONVRG,ITERAT,EQUALU/1.0E-18,25,1.0E-30/

000014 000 C

000015 000 ITN=0

000016 000 DO 5 I=1,3

000017 000 5 PT(I)=PMIN(I)

000018 000 OLDVAL=S(PT)

000019 000 DO 20 J=1,ITERAT

000020 000 JTH=ITH+1

000021 000 CALL PRTRAL(PT,GRAD,HESS)

000022 000 CALL INVERT(HESS,GRAD,ALTRET)

000023 000 IF(ALTRET) GO TO 16

000024 000 DO 10 J=1,3

000025 000 DIR(J)=0

000026 000 DO 10 K=1,3

000027 000 10 DIR(J)=DIR(J)-HESS(J,K)*GRAD(K)

000028 000 DDD=0

000029 000 DO 15 J=1,3

000030 000 15 DDD=DDD+DIR(J)*GRAD(J)

000031 000 IF(DDD.LT.-EQUALU)GOTO 25

000032 000 IF(DDD.GT.+EQUALU)GOTO 19

000033 000 16 DDD=0

000034 000 DO 17 J=1,3

000035 000 DDD=DDD+GRAD(J)**2

000036 000 17 DIR(J)=-GRAD(J)

000037 000 GOTO 25

000038 000 DO 20 J=1,3

000039 000 20 DIR(J)=0

000040 000 CALL LS(PT,POUT,VAL,DDD,DIR)

000041 000 IF(ABS(VAL-OLDVAL).LE..CONVRG*ABS(OLDVAL)) GO TO 60

000042 000 OLDVAL=VAL

000043 000 DO 30 J=1,3

000044 000 30 PT(J)=POUT(J)

000045 000 50 CONTINUE

000046 000 G Norman=0

000047 000 DO 65 I=1,3

000048 000 65 GNorm=GNorm+GRAD(I)**2

000049 000 GNorm=SQRT(GNorm)

000050 000 RETURN

000051 000 END$

@MD6,P ***** NRALIZ/MCIDAS *****
*****  NRMLIZ/MCIDAS  *****

DELT,L AF,NRMLIZ/MCIDAS
ELT007 RL1862 12/22-17:01:57-(0,)
000001  000  SUBROUTINE NRMLIZ(VX,VY,VZ,VNORM)
000002  000  VNORM=SQRT(VX**2+VY**2+VZ**2)
000003  000  VX=VX/VNORM
000004  000  VY=VY/VNORM
000005  000  VZ=VZ/VNORM
000006  000  RETURN
000007  000  ENDS

END ELT.

@HDG,P  *****  OFFSETFIT  *****
PROGRAM TO FIT POLYNOMIAL TO ALTERNATE SCAN OFFSET DATA STORED IN
UNIT 10 BY PROGRAM OFFSETDM. INPUT DATA:
CARD 1: FIRST-POINT-ELEMENT-COORDINATE LAST-POINT
LOGICAL OPTION
DIMENSION ICOORD(2), ASCALE(10), OFSILM(2), YSCALE(10)
DIMENSION XAIL(690), WORK(231)
DIMENSION COEF(21), LABEL(12)
DIMENSION MIN(10), MOUT(3)
DIMENSION ICOORD(230), OFFSET(230), RATE(230)
COMMON/BUFFER/MATE, OFFSET
COMMON/BUFP1/SCORAX, ICOORD
COMMON/BUFP2/NAYM, NAYN, INAY, IDAY, TOTLIN, DECLIN, TOTIEL, DEGELE, PICL
*RP, RICE, LMPSC, IDTH, IDTHA, IM, RIX, RIXA, RIZ, RIZA, H1D, H1D2, P1CH, ROL
*L, LMAEN, IFLM(3), FMT(3), IMA(3), ELCOLY(2), NACOF(2), SCLU(2), SCLU(2),
*ELCOTY(11,2), SCASH(2), SCLU(2), SCLM(11,2), EXCOTF(11,2), NASCF, SCLAS0, SCLAS1.
*ELEMA, ELEMX, ASCOEFF(16)
DATA ICOORD(0, 2400), OFSILM(5, 15, 6)
DATA I15/5/150
DATA MINDEG/15/
Menu=2
CALL INCOM(' ENTER LEFT, RIGHT ELEMENTS FROM PREVIOUS LISTING ', 495)
*3,1,MIN,1,11)
IF(MENU.LE.0) GO TO 999
READ LIMITS OF USEFUL DATA (ICoord LIMITS) FROM CARDS.
ILE=MIN(1)
IRE=MAX(1)
READ IN OFFSETS AND WEIGHTS AND COORDINATES (ELEMENT NUMBERS)
DO 10 1=1, NCOORD
IF(1, COORD1, 2, ILE) ILEA=1
IF(1, ICOORD1, 2, IRE) IREA=1
CONTINUE
NPIS=1, ILEA=1, IREA+1
FIT LEAST SQUARES POLYNOMIAL.
IN=0
DO 25 1=ILEA, IREA
IN=IN+1
WRITE(IN+1,1) ICOORD(1)
WRITE(IN+NPIS,1) OFFSET(1)
WRITE(IN+NPIS,1) RATE(1)
CONTINUE
CALL APCH(DAIL, NPIS, MINDEG, XD, XO, WORK, IER)
IF(IER.NE.0) GO TO 990
ETA=1.0E+3
EPS=1.0E-4
LUP=1
CALL AIFS(WORK, MINDEG, M, LUP, EPS, ETA, IER)
IF(IER.990, 28, 28
CONTINUE
NASCF=M
SCLAS0=X0
SCLAS1=X0
ELEMA=ILE

OELEM=1HE
IA=M*(M-1)/2
DO 29 I=0,M
COEF(I+1)=WORK(I+1+IA)
29 ASCOEF(I+1)=COEF(I+1)
C PRINT COEFFICIENT, DEGREE SCALING
WRITE(II,720)M
720 FORMAT(10X,'DEGREE=',I2)
WRITE(II,721)X0,XD
721 FORMAT(2X,'SCALING PARAMETERS. X0=',E15.9,' XD=',E15.9)
DO 30 I=0,M
30 WRITE(II,722)1,COEF(I+1)
722 FORMAT(10X,'D(1,12)=',E15.9)
GO TO 999
990 CONTINUE
WRITE(II,730)1IER
999 CONTINUE
CALL REQUEST(RAD50('ATSF2'))
730 FORMAT(1X,'ERROR CODE=',I2)
END
***** OFSTF/MCIDAS *****

DELT, L AF, OFSTF/MCIDAS
ELT007 RLIB62 12/22-17:01:58-(0,)
000001 000 $JOB OFSTF U3200
000002 000 $OPTION 8, 9, 13, 20
000003 000 $FORTRAN
000004 000 SUBROUTINE MAIN
000005 000 C PROGRAM TO FIT POLYNOMIAL TO ALTERNATE SCAN OFFSET DATA STORED IN
000006 000 C UNIT 10 BY PROGRAM OFFSETGEN. INPUT DATA:
000007 000 C CARD 1: FIRST-POINT-ELEMENT-COORDINATE LAST-POINT
000008 000 C LOGICAL OPTION
000009 000 C DIMENSION ICORLM(2), YSCALE(10), OFSTLM(2), YSCALE(10)
000010 000 C DIMENSION DATI(400), WORK(231)
000011 000 C DIMENSION COEF(21), LABEL(12)
000012 000 C DIMENSION MIN(10), MOUT(24)
000013 000 C COMMON/OFFDAT/, ICORX, ICOORD(230), OFFSET(230), WATE(230)
000014 000 C COMMON/NAVCOM/, NAVN, INVX, IYR, IDAY, TOTLIN, DEGILN, TOTIEL, DEGLE, PICTI
000015 000 C *NP, PICELE, MPSC, CTY, JDAY, TDY, RIV, RTX, RITX, RIT, RTDX, RTY, RITDX, PITCH, ROL
000016 000 C *L, YAW, KXIK(3), XM(3), YMX(3), NLCOE(2), NRHOE(2), SCLLI(2), SCLLJ(2),
000017 000 C *ELCOE(11, 2), SCLOI(2), SLRI(2), ERICOE(11, 2), NASCEF, SCLASO, SCLA1, SCLA2, SCLA3,
000018 000 C *IELEMN, IELEMX, ASCOE(16)
000019 000 C DATA ICORLM/0, 2400/, OFSTLM/5, 15/*
000020 000 C DATA LP/6/
000021 000 C DATA MINDEG/15/
000022 000 C DATA MIN/6HGC/6CC, 8*/
000023 000 C CALL I6(MIN)
000024 000 C READ LIMITS OF USEFUL DATA (ICOR LIMITS) FROM CARDS.
000025 000 C ILE=MIN(1)
000026 000 C IRE=MIN(2)
000027 000 C JOUT=1
000028 000 C IF(OPTION(MIN(3), 3H P)) JOUT=2
000029 000 C READ IN OFFSETS AND WEIGHTS AND COORDINATES(ELEMENT NUMBER)
000030 000 C CALL DYNAGF"OFFST", LUNF)
000031 000 C CALL OPN(LUNF)
000032 000 C CALL READ(LUNF, 1151, NCORMX)
000033 000 C DO 10 i=1, NCORMX
000034 000 C IF(ICOR(I)=ELE) ILEAE=I
000035 000 C IF(ICOR(I)=ELE) IREAE=I
000036 000 C CONTINUE
000037 000 C NPTS=IREAE-ILEAE+1
000038 000 C FIT LEAST SQUARES POLYNOMIAL.
000039 000 C IN=0
000040 000 C DO 25 I=ILEAE, IREAE
000041 000 C IN=IN+1
000042 000 C DATI(IN)=ICOR(I)
000043 000 C DATI(IN+NPTS)=OFFSET(I)
000044 000 C DATI(IN+2*NPTS)=WATE(I)
000045 000 C CONTINUE
000046 000 C CALL APCH(DATI, NPTS, MINDEG, XD, XD, WORK, IER)
000047 000 C IF(IER.NE.0) GO TO 990
000048 000 C ETA=1, UE=3
000049 000 C EPS=1, UE=4
000050 000 C IO=1
000051 000 C CALL APFS(WORK, MINDEG, M, IO, EPS, ETA, IER)
000052 000 C IF(IER)990, 28, 28
000053 000 C CONTINUE
000054 000 C CALL RCOM
000055 000 C NASCEF=M
**** OFSTF/MCIDAS *****

000056 000 SCLASQ=XQ
000057 000 SCLAS1=XB
000058 000 IELEMN=ILE
000059 000 IELEMX=IRE
000060 000 IX=M*(M-1)/2
000061 000 DO 29 I=0,M
000062 000 COEF(I+1)=WORK(I+1+IX)
000063 000 29 ASCOEF(I+1)=COEF(I+1)
000064 000 C STORE DATA
000065 000 CALL WC0M
000066 000 C PRINT COEFFICIENT, DEGREE SCALING
000067 000 ENCODE(132,720,MOU)M
000068 000 CALL TP(JOUT,MOU)
000069 000 720 FORMAT(10X,"DEGREE=",I2)
000070 000 ENCODE(132,721,MOU)XOXD
000071 000 CALL TP(JOUT,MOU)
000072 000 721 FORMAT(2X,SCALING PARAMETERS X0="E15.9", XD="E15.9")
000073 000 DO 30 I=0,M
000074 000 ENCODE(132,722,MOU)I,COEF(I+1)
000075 000 30 CALL TP(JOUT,MOU)
000076 000 722 FORMAT(10X,"D(",12,")=",E15.9)
000077 000 RETURN
000078 000 990 CONTINUE
000079 000 ENCODE(132,720,MOU)IER
000080 000 CALL TG(MOUT)
000081 000 RETURN
000082 000 730 FORMAT(1X,"ERROR CODE=",I2)
000083 000 ENDS
000084 000 SFILM1A
000085 000 DELETE GCCGGC6G0RP
000086 000 $INCLUDE ATSSSP
000087 000 $CATALOG
000088 000 NAME=GCGGCGS,R,W,D
000089 000 TYPE=FG
000090 000 LIB=ATSFLB,LL
000091 000 BEGIN
000092 000 $EOJ

END ELT.

@MBG,P **** OFSTG/MCIDAS *****
PROGRAM TO GENERATE TABLE
OF ALTERNATE SCAN OFFSETS. CORRELATIONS DONE ON IR.
ASSIGN TAPE TO UNIT TEN. OUTPUT IS TO PRINTER AND GLOBAL COMMON
LOGICAL MISSING, ISLAST
DIMENSION 100ATA(5600), 1EDATA(5600)
EQUIVALENCE(100ATA, 1EDATA)
DIMENSION MINT(10), MOUT(17)
DIMENSION NTBST(2598), NTBRAT(2598)
DIMENSION MA1(230), OFFSET(230), ICOUNT(230), FLWORD(9)
COMMON /BUF1/ A1, OFFSET
COMMON /BUFF1/ NCORNA, ICOUN
DATA 1A1AG/07, NLAG/9/, NCOR/220/, LPR1/6/
DATA N1/197/, N2/199/, N3/5598/
DATA LUN/10/, MISSING/, RANGE/
DATA DP/6/, 11/5/
DATA IFIRST/800/, INICN/8/, LSTSCN/400/
DATA INEX/100/, N1SIZ/15/, NOOFFSET/6/, ISELECT/10/
DATA NOP/2598/, IDDVF/198/
DATA LSTTXT/6/
C
INITIALIZE SCAN SEARCH
IFIRST= (IFIRST+1)/2+2
INICN=(INICN/2)*2
LSTSCN=(LSTSCN+60)/2+2
ISCAN=IFIRST+INICN
NCORNA=0
DO 10 I=1, 230
A1(I)=0.
OFFSET(I)=0.0
ICOUNT(I)=0
10 CONTINUE
LNCODE(74, 702, MOUT)
702 FORMAT(I4, ' MOUNT TAPE. ENTER DRIVE NUMBER')
MENU=1
CALL INCOM(MOUT, 30, 1, 1, MN1, MENU, 1, 1)
IMUNIT=MN1(1)
IF(MENU.LT.0) GO TO 999
IF(MENU.EQ.0) IMUNIT=0
CALL ASHUN(MUN, 'MN', IMUNIT)
CALL IUPLNM(MUN)
C
SKIP HEADER RECORDS
CALL IUFSRM(MUN, '+2, ISTAT)
100 IF(MISSN=1) IESCAN=1ESCAN+2
IF(MISSN=0, MISSN) IESCAN=1ESCAN-LISTCN
IESCAN=IESCAN-1
ISLAS=IESCAN, LE, LSTSCN
200 CALL IUFSRM(MUN, 'MDATA, N1, LE, ISTAT)
CALL CH1LNM(19, 1EDATA, LEBRT)
IESCAN=LEBRT(195)
MISSN=IESCAN, LE, IESCAN
IF(MISSN=NAIL, I1, 701) IESCAN
FORTAT(I4, ' SCAN NUMBER', I5, ' NOT FOUND.')
IF(MISSN=1) GO TO 100
IF(IESCAN.GT.1ESCAN) GO TO 200
C   EVEN SCAN Found, NOW CHECK FOR UDD Scan.
CALL I0UPIN(LUN, IODATA, #1, LE, ISTAT)
CALL CKAHK(195, IODATA, I0UBRK)
J0SCAN=I0UBRK(195)
MISSAG=J0SCAN, N, I0SCAN
IF(MISSAG) WRITE(11, 701) I0SCAN
IF(MISSAG) GO TO 100
C   BACK UP TO READ UDD DATA.
CALL I0UPSR(LUN, 2, ISTAT)
C   INPUT PATH OF SCANS.
CALL I0UPIN(LUN, IODATA, #2, LE, ISTAT)
CALL CKTHK(195, I0DATA, I0BRK)
J0SCAN=I0BRK(195)
CALL CKRAIS(NBR, I0DATA, I0BRK)
CALL I0UPIN(LUN, I0DATA, #2, LE, ISTAT)
CALL CKTHK(195, I0DATA, I0BRK)
J0SCAN=I0BRK(195)
CALL CKRAIS(NBR, I0DATA, I0BRK)
IF(J0SCAN & #4, I0DATA, I0BRK) GO TO 990
C
C   DO CORRELATIONS.
C   DO 600 KCUR=1, NCRK
ILEE=IELST+(KCUR-1)*IELINT
ICUR=KCUR+1
MALEAG=(NLAG-1)/2
MINLAG=MALEAG
DU 500 KLAG=MINLAG, MALEAG
MAXBR=0
MINBR=512
NTSZ=(NTSZ-1)/2+1
FLP=0
DO 400 KVAL=1, NTSZ
ICT=1*ILEE+KVAL-1=NTSZ/2
ICS=1*ILEE+KVAL-1+KLAG+NUMOFF-NTSZ/2
ICTX=ICT+10OFF
ICSX=ICS+10OFF
IPIAT=ICTX(ICTX)
IPIAS=ICSX(ICTX)
MAXBR=MAX(MAXBR, IPIAT)
MINBR=MIND(MINBR, IPIAT)
FLP=FLP+IPIAT-IPIAS+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.
1OFF=NUMOFF+MINLAG
FLP#1=FLPARY(1)
DO 510 K=2, NLAG
IF(FLP#<1.GT.FLPARY(K)) 1OFF=NUMOFF+(K-1)+MINLAG
IF(FLP#<1.GT.FLPARY(K)) FLP#1=FLPARY(K)
510 CONTINUE
C   WEIGHT OFFSETS BY BRIGHTNESS RANGE.
W=FUCAI(MAXBR-MINBR)/512.
WAIE(KCUR)=WAIE(KCUR)+W
OFFSET(KCUR)=OFFSET(KCUR)+1OFE+W
600 CONTINUE
C   IF(.NOT.ISLAST) GO TO 100
C   COMPUTE AVERAGE OFFSETS.
DO 610 KCUR=1, NCUR
IF(.NOT.ISLAST) GO TO 610
OFFSET(KCUR)=OFFSET(KCUR)/WAIE(KCUR)
610 CONTINUE
C   DUMP TABLE VALUES.
N2COL=(NCUR+1)/51
OFSTG/GOIPS

DO 650 IWW=1,NICOL
WRITE(LP,710)(ICUOR(IW),OFFSET(IW),MAIE(IW),IWW=IWW,NCOR,NICOL)
650 CONTINUE
710 FORMAT((5(5X,14,1X,F8.3,F8.4))))
NCORMA=NCOR
GO TO 999
990 CONTINUE
WRITE(5,715)
715 FORMAT(2X,'OFFSETGEN ERROR RETURN. ')
999 CONTINUE
CALL LUFRM(LUN)
CALL REQUEST(RAD50('OFSTF2'))
END
BELT,L AF,OSTG/MCIDAS
ELD07 RL1662 12/22-17:01:50-9(0)
000001 000 $JOB ATS6 U3200
000002 000 $OPTION,8,9,20
000003 000 $FORTRAN

000004 000 C SUBROUTINE MAIN
000005 000 C OF ALTERNATE SCAN OFFSETS, CORRELATIONS DONE ON IR.
000006 000 C ASSIGN TAPE TO UNIT 10, OUTPUT IS TO PRINTER AND UNIT 20.
000007 000 C LOGICAL MISSNG, IALAST
000008 000 C DIMENSION IOCDATA(1299), IEBDATA(1299)
000009 000 C DIMENSION MIN(70), MOUT(24)
000010 000 C DIMENSION ID(2598), IEBRT(2598)
000011 000 C DIMENSION WATE(230), OFFSET(230), ICOORD(230), FLPAHRY(9)
000012 000 C COMMON/OFFDAT, NCURM, ICOORD, OFFSET, WATE
000013 000 C DATA ISTATG/0/, NLAG/9/, NCOORD/220/, LPRT/6/
000014 000 C DATA MIN/6, HC1CC/6, R=0/
000015 000 C DATA NH/98/, NW2/1299/, NW3/3732/
000016 000 C DATA LUN/10/, MISSNG/1, FALSE, L
000017 000 C DATA LUNF/20/
000018 000 C DATA IFIRST/600/, INTSCN/ 5/, LOSTCN/400/
000019 000 C DATA ILEST/100/, NTS2T/15/, NOMOFF/ 8/, ILEINT/10/
000020 000 C DATA NHB/2598/, ITDFF/198/
000021 000 C DATA IUBRT/67/;
000022 000 C CALL IQ(MIN)
000023 000 C IREEL=0
000024 000 C C INITIALIZE SCAN SEARCH
000025 000 C IFIRST=(IFIRST+1)/2+2
000026 000 C INTSCN=(INTSCN/2)*2
000027 000 C LOSTCN=(LSTCN+1)/2+2
000028 000 C IESC=IFIRST+INTSCN
000029 000 C CALL GTAP(14, IREEL, LUN)
000030 000 C CALL RELU(LUN)
000031 000 C C SKIP HEADER RECORDS
000032 000 C CALL READW(LUN,NW, IEDATA)
000033 000 C CALL READW(LUN, NW1, IEDATA)
000034 000 C 100 IF(MISSNG) IESC=IESCAN-2
000035 000 C IF(NOT,MISSNG) IESC=IESCAN-INTSCN
000036 000 C IESC=IESCAN-1
000037 000 C ISLAST=IESCAN, IESC=INTSCN
000038 000 C CALL READW(LUN, NW, IEDATA)
000039 000 C CALL CRKTHR(195, IEDATA, IEBRT)
000040 000 C JESC=IEBRT(195)
000041 000 C MISSNG=JESCAN, LTR, IESCAN
000042 000 C IF(MISSNG) ENCODE(132, 701, MOUT) IESCAN
000043 000 C IF(MISSNG) CALL TQ(MOUT)
000044 000 C 701 FORMAT(1x, "SCAN NUMBER", 15, " NOT FOUND.")
000045 000 C IF(MISSNG) GO TO 100
000046 000 C IF(JESCAN, GT, IESCAN) GO TO 200
000047 000 C EVEN SCAN FOUND, NOW CHECK FOR ODD SCAN.
000048 000 C CALL READW(LUN, NW, IEDATA)
000049 000 C CALL CRKTHR(195, IEDATA, IEBRT)
000050 000 C JESC=IEBRT(195)
000051 000 C MISSNG=JESCAN, LTR, IESCAN
000052 000 C IF(MISSNG) ENCODE(132, 701, MOUT) IESCAN
000053 000 C IF(MISSNG) CALL TQ(MOUT)
000054 000 C IF(MISSNG) GO TO 100
000055 000 C BACK UP TO READ IN DATA.
CALL BSRLUN
CALL BSRLUN
C INPUT PAIR OF SCANS
CALL READLUN, NW2, IEDATA
CALL READLUN, NW2, IODDATA
CALL CRKTHR(195, IEDATA, IEBRT)
JESCAN=IEBRT(195)
CALL CRKATS(NBP, IEDATA, IEBRT)
CALL CRKTHR(195, IODDATA, IOBRT)
JOSCAN=IOBRT(195)
CALL CRKATS(NBP, IODDATA, IOBRT)
C IF(JESCAN .NE. JOSCAN .OR. JOSCAN .NE. IOSCAN) GO TO 990
C DO CORRELATIONS.
DO 600 KCOR=1, NCOR
IELE=IELEST+(KCOR-1)XIELINT
ICOR(KCOR)=IELE
MAXLAG=NLAG-1)/2
MINLAG=-MAXLAG
DO 500 KLAG-MINLAG, MAXLAG
MAXRT=0
MINBRT=512
NTSZ1=(NTSZ1-1)/2+2+1
FLP=0
DO 400 KVAL=1, NTSZ
ICT=IELE+KVAL-1-NTSZ/2
ICS=IELE+KVAL-1-KLAG+NOFF-NTSZ/2
ICTX=ICT+IDOFF
ICSX=ICS+IDOFF
IPIXT=IOBRT(ICTX)
IPIXS=IEBRT(ICSX)
MAXBRT=MAXD(MAXBRT, IPIXT)
MINBRT=MIND(MINBRT, IPIXT)
FLP=FLP+(IPIXT-IPIXS)X2
CONTINUE
DO 400 CONTINUE
C STORE LP(2) MEASURE VALUE
FLPARY(KLAG-MINLAG+1)=FLP
CONTINUE
C SEARCH TABLE OF VALUES OF LP(2) MEASURE FOR MIN
IOFF=NOFF+MINLAG
FLPMIN=FLPARY(1)
DO 510 K=2, NLAG
IF(FLPMIN.GT.FLPARY(K)) IOFF=NOFF+(K-1)+MINLAG
FLPMIN=FLPARY(K)
CONTINUE
WEIGHT OFFSETS BY BRIGHTNESS RANGE.
WT=FLOAT(MAXBRT-MINBRT)/512.
WATE(KCOR)=WATE(KCOR)+WT
OFFSET(KCOR)=OFFSET(KCOR)+IOFF*WT
CONTINUE
IF (.NOT. ILAST) GO TO 100
CONTINUE
C COMPUTE AVERAGE OFFSETS
DO 610 KCOR=1, NCOR
IF(WATE(KCOR).LT.1.0E-3) GO TO 610
CONTINUE
OFFSET(KCOR)=OFFSET(KCOR)/WATE(KCOR)
CONTINUE
***** OFSTG/MCIDAS *****

000113 000  C  DUMP TABLE VALUES
000114 000  NCOL=(NCOR-1)/5+1
000115 000  JOUT=2
000116 000  DO 650 IWW=1,NCOL
000117 000  ENCODE(132,710,MOUT)(ICOR(IW),OFFSET(IW),WATE(IW),IW=IWW,NCOR,
000118 000  *NCOL)
000119 000  CALL TP(JOUT,MOUT)
000120 000  650 CONTINUE
000121 000  710 FORMAT ((5(5X,14,1X,FB,3,FB,4))
000122 000  NCORMX=NCOR
000123 000  CALL DYNAG("OFFSTD",LUNF)
000124 000  CALL OPN(LUNF)
000125 000  CALL WRITW(LUNF,1151,NCORMX)
000126 000  RETURN
000127 000  990 CONTINUE
000128 000  CALL TG(72H OFFSETGEN ERROR RETURN.
000129 000  *
000130 000  RETURN
000131 000  ENDS
000132 000  $FILEMA
000133 000  DELETE GCCGCC,GORP
000134 000  $INCLUDE IFLD
000135 000  $CATALOG
000136 000  NAME=GCCGCC,5,R,W,D
000137 000  TYPE=FG
000138 000  LIB=ATSFLB,LL
000139 000  BEGIN
000140 000  $EOJ

END ELT
@HDG,P ***** OFSTVL *****
****** PFTOTC/MCIDAS ******

BELT, L AF*PFTOTC/MCIDAS
ELT007 RLI662 12/22-17:02:02-(0)

000001 000 SUBROUTINE PFTOTC(XLIN, XELE, X, Y, Z, IDIR, INIT)
000002 000 COMMON/NAVCOM/NAV, INA, IY, IDAY, TOTLIN, DEGLIN, TOTIEL, DEGELE, PICLI
000003 000 *N, PICLEL, TMPSCALE, IDIRP, IDAYP, TR, R1X, R1Y, R1Z, R1DX, R1DY, R1DZ, PITCH, ROL
000004 000 *L, YAWP, TTMPSCALE, TMX(3), TMX(3), TMX(2), TMX(2), TMLINE(2), TMLINE(2),
000005 000 *E, LCOEF(1, 2), SCLR0(2), SCLR1(2), ERCOEF(1, 2), NASCEF, SCLASGY, SCLAST,
000006 000 *IELEMN, IELEMX, ASCOEF(16)
000007 000 C IF IDIR = 1, X, Y, Z TO LIN, ELE
000008 000 C IF IDIR = 2, LIN, ELE TO X, Y, Z
000009 000 DATA PI, RE, ERACON/3.14159265, 6.378.15, 074365743/
000010 000 IF(INIT, EQ.2)GO TO 1
000011 000 INIT = 2
000012 000 RDPDG = PI/180.
000013 000 RADLIN = RDPDG*DEGLIN/TOTLIN
000014 000 RADELE = RDPDG*DEGELE/TOTIEL
000015 000 1 IF(INIT, EQ.2)GO TO 10
000016 000 ELEANG = ATAN(X/Z)
000017 000 XLNANG = ASIN(Y)
000018 000 XELE = PICLEL+XLEANG/RADELE
000019 000 XLIN = PICLIN+XLNANG/RADELIN
000020 000 RETURN
000021 000 10 ELEANG = (PICLEL-XELE)*RADELE
000022 000 XLNANG = (PICLIN-XLIN)*RADLIN
000023 000 X = COS(XLNANG)*SIN(ELEANG)
000024 000 Y = -SIN(XLNANG)
000025 000 Z = COS(XLNANG)*COS(ELEANG)
000026 000 RETURN
000027 000 ENDS

END ELT*

@HDG,P ****** PTIAL/MCIDAS ******

E-45
***** PRTIAL/MCIDAS *****

CALL, AF,PRTIAL/MCIDAS
ELT007 RLIB62 12/22-17:02:03-(0,)
000001  000  SUBROUTINE PRTIAL(PT,GRAD,HESS)
000002  000  C
000003  000  C  THIS ROUTINE RETURNS IN "GRAD" THE GRADIENT OF S AT "PT" AND IN
000004  000  C  "HESS" THE HESSIAN OF S AT "PT".
000005  000  C
000006  000  DIMENSION X(3,150),Y(3,150),TIME(150)
000007  000  COMMON/MINC/X,Y,TIME,NP
000008  000  DIMENSION PT(3),GRAD(3),HESS(3,3),HA(3,3),HB(3,3),HC(3,3),
000009  000  *HAA(3,3),HAB(3,3),HAC(3,3),HBB(3,3),HBC(3,3),HCC(3,3),
000010  000  A=PT(1)
000011  000  B=PT(2)
000012  000  C=PT(3)
000013  000  PSFA=0
000014  000  PSPB=0
000015  000  PSC=0
000016  000,  PSPASQ=0
000017  000  PSPAP=0
000018  000  PSPAPC=0
000019  000  PSPAPSQ=0
000020  000  PSPBPC=0
000021  000  PSPCPSQ=0
000022  000  CALL UNIT(H)
000023  000  CALL ROTATE(H,C,3,1)
000024  000  CALL ROTATE(H,B,1,1)
000025  000  CALL ROTATE(H,A,2,1)
000026  000  CALL UNIT(HA)
000027  000  CALL ROTATE(HA,C,3,1)
000028  000  CALL ROTATE(HA,B,1,1)
000029  000  CALL ROTATE(HA,A,2,1)
000030  000  CALL UNIT(HB)
000031  000  CALL ROTATE(HB,C,3,1)
000032  000  CALL ROTATE(HB,B,1,2)
000033  000  CALL ROTATE(HB,A,2,1)
000034  000  CALL UNIT(HC)
000035  000  CALL ROTATE(HC,C,3,2)
000036  000  CALL ROTATE(HC,B,1,1)
000037  000  CALL ROTATE(HC,A,2,1)
000038  000  CALL UNIT(HAA)
000039  000  CALL ROTATE(HAA,C,3,1)
000040  000  CALL ROTATE(HAA,B,1,1)
000041  000  CALL ROTATE(HAA,A,2,1)
000042  000  CALL UNIT(HAD)
000043  000  CALL ROTATE(HAD,C,3,1)
000044  000  CALL ROTATE(HAD,B,1,2)
000045  000  CALL ROTATE(HAD,A,2,2)
000046  000  CALL UNIT(HAC)
000047  000  CALL ROTATE(HAC,C,3,2)
000048  000  CALL ROTATE(HAC,B,1,1)
000049  000  CALL ROTATE(HAC,A,2,2)
000050  000  CALL UNIT(HBB)
000051  000  CALL ROTATE(HBB,C,3,1)
000052  000  CALL ROTATE(HBB,B,1,1)
000053  000  CALL ROTATE(HBB,A,2,1)
000054  000  CALL UNIT(HBC)
000055  000  CALL ROTATE(HBC,C,3,2)
CALL ROTATE(HBC, A\_1, 2)
CALL ROTATE(HBC\_A, A\_2, 1)
CALL UNIT(HCC)
CALL ROTATE(HCC\_C, A\_3, 3)
CALL ROTATE(HCC\_A, A\_1, 1)
CALL ROTATE(HCC\_A\_1, A\_2, 1)
D0 10 J=1,NP
D0 10 I=1,3
T=Y(I, J)-H(I, 1)*X(I, J)-H(I, 2)*X(2, J)-H(I, 3)*X(3, J)
T A=HA(A, 1)*X(I, J)+HA(A, 2)*X(2, J)+HA(A, 3)*X(3, J)
T B=HB(I, 1)*X(I, J)+HB(I, 2)*X(2, J)+HB(I, 3)*X(3, J)
T C=HC(I, 1)*X(I, J)+HC(I, 2)*X(2, J)+HC(I, 3)*X(3, J)
T A A=HAA(A, 1)*X(I, J)+HAA(A, 2)*X(2, J)+HAA(A, 3)*X(3, J)
T A B=HAB(I, 1)*X(I, J)+HAB(I, 2)*X(2, J)+HAB(I, 3)*X(3, J)
T A C=HAC(I, 1)*X(I, J)+HAC(I, 2)*X(2, J)+HAC(I, 3)*X(3, J)
T B A=HBA(I, 1)*X(I, J)+HBA(I, 2)*X(2, J)+HBA(I, 3)*X(3, J)
T B B=HBB(I, 1)*X(I, J)+HBB(I, 2)*X(2, J)+HBB(I, 3)*X(3, J)
T B C=HBC(I, 1)*X(I, J)+HBC(I, 2)*X(2, J)+HBC(I, 3)*X(3, J)
T C A=HCA(I, 1)*X(I, J)+HCA(I, 2)*X(2, J)+HCA(I, 3)*X(3, J)
T C B=HCB(I, 1)*X(I, J)+HCB(I, 2)*X(2, J)+HCB(I, 3)*X(3, J)
T C C=HCC(I, 1)*X(I, J)+HCC(I, 2)*X(2, J)+HCC(I, 3)*X(3, J)
PS A=PS A-T*TA
PS B=PS B-T*TB
PS C=PS C-T*TC
PS A S Q=PS A S Q-T*TA A+TA**2
PS A P B=PS A P B-T*TA B+TA*TB
PS A P C=PS A P C-T*TA C+TA*TC
P S B S Q=PS B S Q-T*T B B+TB**2
P S P C P C=PS P C P C-T*T B C+TB*TC
P S P C S Q=PS P C S Q-T*T C C+TC**2
G R A D (1)=2*PS A
G R A D (2)=2*PS B
G R A D (3)=2*PS C
H E S S (1, 1)=2*PS A S Q
H E S S (1, 2)=2*PS A P B
H E S S (1, 3)=2*PS A P C
H E S S (2, 1)=2*PS P A B
H E S S (2, 2)=2*PS P A C
H E S S (3, 1)=2*PS P B C
H E S S (2, 2)=2*PS P B Q
H E S S (3, 2)=2*PS P B C
H E S S (3, 3)=2*PS P C S Q
R E T U R N
E N D S

END ELT.
@HGD,P ***** ROTATE/MCIDAS *****
***ROTAPE/MCIDAS***

@ELT, L AF,ROTAPE/MCIDAS
ELT007 RL1663 12/22-17:02:03-(0)
000001 000 SUBROUTINE ROTATE(A,R,IR,IDERV)
000002 000 C
000003 000 C THIS ROUTINE RETURNS IN "A" THE PRODUCT OF THE INPUT MATRIX
000004 000 C "A" AND A MATRIX RM, WHERE, IF "IDERV"=1, RM REPRENS A
000005 000 C ROTATION THROUGH AN ANGLE "R" (IN RADIANS) ABOUT THE AXIS "IR". IF
000006 000 C IDERV=2, THE FIRST DERIVATIVE OF RM IS OPERATED ON A; AND IF
000007 000 C IDERV=3, THE SECOND DERIVATIVE OF RM IS USED.
000008 000 C
000009 000 DIMENSION A(3,3),INDX1(3),INDX2(3)
000010 000 DATA INDX1,INDX2/1,1,3,1,3,2/
000011 000 IR1=INDX1(IR)
000012 000 IR2=INDX2(IR)
000013 000 CR=DCOS(R)
000014 000 SR=DSIN(R)
000015 000 IF(IDERV.NE.1)GO TO 2
000016 000 DO 1 J=1,3
000017 000 T1=A(IR1,J)
000018 000 T2=A(IR2,J)
000019 000 A(IR1,J)=CR*T1+SR*T2
000020 000 A(IR2,J)=-SR*T1+CR*T2
000021 000 1 CONTINUE
000022 000 RETURN
000023 000 2 IF(IDERV.NE.2)GO TO 4
000024 000 DO 3 J=1,3
000025 000 A(IR1,J)=0.0
000026 000 T1=A(IR1,J)
000027 000 T2=A(IR2,J)
000028 000 A(IR1,J)=-SR*T1+CR*T2
000029 000 A(IR2,J)=-CR*T1-SR*T2
000030 000 3 CONTINUE
000031 000 RETURN
000032 000 4 CONTINUE
000033 000 DO 5 J=1,3
000034 000 A(IR1,J)=0.0
000035 000 T1=A(IR1,J)
000036 000 T2=A(IR2,J)
000037 000 A(IR1,J)=-CR*T1-SR*T2
000038 000 A(IR2,J)=SR*T1-CR*T2
000039 000 5 CONTINUE
000040 000 RETURN
000041 000 END$
****** S/MCIDAS ******

@ELT,L,AF,S/MCIDAS
ELT007 RL1862 12/22-17:02:04-(0,)
000001  000  FUNCTION S(PT)
000002  000  C  THIS FUNCTION RETURNS AS ITS VALUE THE VALUE OF THE OBJECTIVE
000003  000  C  function S AT THE POINT "PT".
000004  000  C
000005  000  C
000006  000  DIMENSION X(3,150),Y(3,150),TIME(150)
000007  000  COMMON/MINCOM/X,Y,TIME,NP
000008  000  DIMENSION PT(3),H(3,3)
000009  000  SRES=0
000010  000  CALL UNIT(H)
000011  000  CALL ROTATE(H,PT(3),3,1)
000012  000  CALL ROTATE(H,PT(2),1,1)
000013  000  CALL ROTATE(H,PT(1),2,1)
000014  000  DO 10 J=1,NP
000015  000  DO 10 I=1,3
000016  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
000017  000  S=SRES
000018  000  RETURN
000019  000  ENDS

END ELT.

S/MCIDAS ****** SATCAP/MCIDAS ******
***** SATEAR/MCIDAS *****

GELT, L AF, SATEAR/MCIDAS
ELT007 RL1862 12/22-17:02:05-(0,)
000001 000 SUBROUTINE SATEAR(PICTIM, XLIN, XELE, XLAT, XLON, ITYPE, INAV, GETAIN, BET
000002 000 *DOT, ATFRAC
000003 000 PTIME=PICTIM
000004 000 GO TO (1,2,3,4,5), ITYPE
000005 000 1 CALL SE(PTIME, XLIN, XELE, XLAT, XLON)
000006 000 RETURN
000007 000 2 CALL ES(PTIME, XLAT, XLON, XLIN, XELE)
000008 000 RETURN
000009 000 3 CONTINUE
000010 000 RETURN
000011 000 4 CONTINUE
000012 000 RETURN
000013 000 5 CONTINUE
000014 000 RETURN
000015 000 END

END ELT.
SHGD,P ***** SE/MCIDAS *****
Delt, LAF, SE/MCIDAS
ELT007 RLIB62 12/22-17:02:06-(0)
000001 000 SUBROUTINE SE (PTIME, XLIN, XELE, ALAT, ALON)
000002 000 COMMON/NAVCOM/NAV, INAV, IYR, IDAY, TOTLIN, DEGLIN, TOTIEL, DEGELE, PICTL
000003 000 *N, PICELE, TMSPL, I0YR, I0DAY, TM, R1X, R1Y, R1Z, R1DX, R1DY, R1DZ, PITCH, ROL
000004 000 *L, YAW, PTIM(3), TMN(3), TMX(3), NLCOEF(2), NRCOEF(2), SCLLO(2), SCLL1(2),
000005 000 *ELOCEF(11, 2), SCL0(2), SCLR1(2), ERCOEF(11, 2), NASCEF, SCLASD, SCLAS1,
000006 000 *JELEMN, JIELEMX, ASCOE(16)
000007 000 DATA INIT/D/
000008 000 CALL EDGECOR (PTIME, XLIN, DELLIN, DELELE)
000009 000 ALIN=XLIN-DELLIN
000010 000 AELE=XLIN-DELELE
000011 000 ISCANT=1200-(1IFIX(XLIN-1))/2
000012 000 TIME=PTIME+ISCAN*TMSPL
000013 000 IDIR=2
000014 000 CALL PTOTC (ALIN, AELE, X1, X2, X3, IDIR, INIT)
000015 000 CALL OCTPFL (X1, X2, X3, IDIR)
000016 000 C BC SAME AS LV HERE.
000017 000 CALL STTOLV (X1, X2, X3, IDIR, TIME)
000018 000 CALL ERTOST (X1ER, X2ER, X3ER, X1, X2, X3, IDIR, TIME)
000019 000 CALL ERTOGER (ALAT, ALON, X1ER, X2ER, X3ER, IDIR)
000020 000 RETURN
000021 000 END

END ELT

**HDG,** P **STTOLV/MCIDAS**
SUBROUTINE STTOLV (X,Y,Z,DIR,TIME)
C IF DIR=1, POINTING VECTOR (X,Y,Z) IS TRANSFORMED
C FROM SAT INERTIAL TO LOCAL VERTICAL FRAME.
C IF DIR=2, POINTING VECTOR (X,Y,Z) IS TRANSFORMED FROM
C LOCAL VERTICAL TO SAT INERTIAL FRAME.
CALL ORBIT(XS,YS,ZS,TIME)
CALL NAMLZ(XS,YS,ZS,XNORM)
X1=X
Y1=Y
Z1=Z
D=SQRT(XS**2+YS**2)
IF (DIR.EQ.2) GO TO 10
X=(YS*X1+XS*Y1)/D
Y=(XS*ZS+YS*YS*Y1-Z1*D**2)/D
Z=-(XS*X1+YS*Y1+ZS*Z1)
RETURN
10 X=-YS*X1/D+XS*ZS*Y1/D-XS*Z1
Y=XS*X1/D+YS*ZS*Y1/D-YS*Z1
Z=-D*Y1-ZS*Z1
RETURN
END
UNIT/MCIDAS

** **

DATA ** **

*EKT, L AF*UNIT/MCIDAS
ELTO07 RL1862 12/22-17:02:09-(O,)

000001 000 SUBROUTINE UNIT(A)
000002 000 C THIS ROUTINE RETURNS IN "A" A 3 X 3 IDENTITY MATRIX
000003 000 C
000004 000 DIMENSION A(9), B(9)
000005 000 DATA B/1.0, 0.0, 0.0, 1.0/
000006 000 * 0.0, 1.0, 0.0, 0.0/*
000007 000 * 0.0, 0.0, 1.0/*
000008 000 DO 10 I=1, 9
000009 000 10 A(I)=B(I)
000010 000 RETURN
000011 000 ENDS

END ELT
ENDhind
END