A METHOD OF PHOTO DATA REDUCTION WITH DESIGN CONSIDERATIONS FOR THE NOVA. (U) NAVAL BIODYNAMICS LAB NEW ORLEANS LA J J LAMBERT JAN 84 NBDL-84R001
A METHOD OF PHOTO DATA REDUCTION, WITH DESIGN CONSIDERATIONS FOR THE NOVA 800® AND UNIVAC 1100/83® COMPUTERS

James J. Lambert

NAVAL BIODYNAMICS LABORATORY
New Orleans, Louisiana

Approved for public release. Distribution unlimited.
<table>
<thead>
<tr>
<th>REPORT NUMBER</th>
<th>GOVT ACCESSION NO.</th>
<th>RECIPIENT'S CATALOG NUMBER</th>
</tr>
</thead>
<tbody>
<tr>
<td>NBDL-84001</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TITLE</th>
<th>TYPE OF REPORT &amp; PERIOD COVERED</th>
</tr>
</thead>
<tbody>
<tr>
<td>A Method of Photo Data Reduction, with Design Considerations for the NOVA 800® and UNIVAC 1100/83® Computers</td>
<td>Research Report</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AUTHOR(s)</th>
<th>CONTRACT OR GRANT NUMBER(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>James J. Lambert</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PERFORMING ORGANIZATION NAME AND ADDRESS</th>
<th>PROGRAM ELEMENT, PROJECT, TASK AREA &amp; WORK UNIT NUMBERS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naval Biodynamics Laboratory</td>
<td>M0097PN001-5001</td>
</tr>
<tr>
<td>P. O. Box 29407</td>
<td></td>
</tr>
<tr>
<td>New Orleans, LA 70189</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CONTROLLING OFFICE NAME AND ADDRESS</th>
<th>REPORT DATE</th>
<th>NUMBER OF PAGES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naval Medical Research &amp; Development Command</td>
<td>January 1984</td>
<td>86</td>
</tr>
<tr>
<td>Bethesda, MD 20814</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DISTRIBUTION STATEMENT (of this Report)</th>
<th>DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Approved for public release; distribution unlimited</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SUPPLEMENTARY NOTES</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

| KEY WORDS (Continue on reverse side if necessary and identify by block number) |
| data analysis, computer program, photo data reduction, photo digitizing system (PDS), high speed camera, biodynamic experiments, inertial environments, impact injury models |

| ABSTRACT (Continue on reverse side if necessary and identify by block number) |
| Digitized photo data acquired during impact experiments must be scaled and converted to a format suitable for analysis. Once converted, this data requires comprehensive graphical presentation for efficient interpretation. This report presents a detailed description of the software developed to accomplish such tasks in both a production and an interactive environment. Procedures utilizing the design presented have been instituted at NBDL and found to be effective. |
A METHOD OF PHOTO DATA REDUCTION, WITH DESIGN CONSIDERATIONS

FOR THE NOVA 800® AND UNIVAC 1100/83® COMPUTERS

James J. Lambert

January 1984

Naval Medical Research and Development Command
Research Work Unit No. MD097PN001-5001

Approved by

J. C. Guignard
Chairman, Editorial Review Board

Released by

Captain L. E. WILLIAMS MC USN
Commanding Officer

Naval Biodynamics Laboratory
P. O. Box 29407
New Orleans, LA  70189

Opinions or conclusions contained in this report are those of the author(s) and do not necessarily reflect the views or the endorsement of the Department of the Navy. Approved for public release; distribution unlimited. Reproduction in whole or in part is permitted for any purpose of the United States Government.
THE PROBLEM

Digitized photo data acquired during impact experiments must be scaled and converted to a format suitable for analysis. Once converted, these data require comprehensive graphical presentation for efficient interpretation. This report presents a detailed description of the software developed to accomplish such tasks in both a production and an interactive environment.

FINDINGS

Procedures utilizing the design presented have been instituted at NBDL and found to be effective.

RECOMMENDATIONS

In an environment requiring data reduction processing of photo data, the design presented herein should be considered.

ACKNOWLEDGEMENT

This research was sponsored by the Naval Medical Research and Development Command and was performed under Navy Work Unit No. M0097PN001-5001. The author acknowledges Mr. William R. Anderson for valuable technical collaboration proving useful in the presentation of findings of this research. Miss Judy B. Johnson expended much time and patience in creating and maintaining the report text using NBDL word processing equipment. Additionally, Mr. Arthur M. Prell and Miss Margaret M. Harbeson were responsible for the creation of photographic reproductions (of illustrations) and the manuscript editing functions, respectively.

Trade names of materials or products of commercial or nongovernment organizations are cited only where essential for precision in describing research procedures or evaluation of results. Their use does not constitute official endorsement or approval of the use of such commercial hardware or software.
# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1. PHOTO DATA CONVERSION PROGRAM</td>
<td>2</td>
</tr>
<tr>
<td>A. INTRODUCTION</td>
<td>2</td>
</tr>
<tr>
<td>B. DISCUSSION</td>
<td>2</td>
</tr>
<tr>
<td>C. MAIN PROGRAM AND SUBROUTINE DESCRIPTIONS</td>
<td>5</td>
</tr>
<tr>
<td>2. PLOT PROGRAM FOR PHOTO DATA</td>
<td>6</td>
</tr>
<tr>
<td>A. INTRODUCTION</td>
<td>6</td>
</tr>
<tr>
<td>B. DISCUSSION</td>
<td>6</td>
</tr>
<tr>
<td>C. MAIN PROGRAM AND SUBROUTINE DESCRIPTIONS</td>
<td>9</td>
</tr>
<tr>
<td>3. REFORMATTING (SIGN CHANGE) PROGRAM FOR PHOTO DATA</td>
<td>10</td>
</tr>
<tr>
<td>A. INTRODUCTION</td>
<td>10</td>
</tr>
<tr>
<td>B. DISCUSSION</td>
<td>10</td>
</tr>
<tr>
<td>C. MAIN PROGRAM DESCRIPTION</td>
<td>11</td>
</tr>
<tr>
<td>4. OPERATION OF PHOTO DATA REDUCTION SYSTEM</td>
<td>11</td>
</tr>
<tr>
<td>A. INTRODUCTION</td>
<td>11</td>
</tr>
<tr>
<td>B. DISCUSSION</td>
<td>11</td>
</tr>
<tr>
<td>C. PRODUCTION RUNSTREAM</td>
<td>15</td>
</tr>
<tr>
<td>Conclusion</td>
<td>17</td>
</tr>
<tr>
<td>Figures</td>
<td>18</td>
</tr>
<tr>
<td>References</td>
<td>39</td>
</tr>
<tr>
<td>Appendices</td>
<td></td>
</tr>
</tbody>
</table>
A METHOD OF PHOTO DATA REDUCTION, WITH DESIGN CONSIDERATIONS FOR THE NOVA 800® AND UNIVAC 1100/83® COMPUTERS

INTRODUCTION

The Naval Biodynamics Laboratory (NBDL), located in New Orleans, LA, is an internationally recognized laboratory which performs experimental research to determine the effects of aircraft crashes, ship motion, vibration, aircraft ejection and parachute opening forces on the health and performance of Navy personnel. On-going research programs use high speed instrumentation cameras to record the motion of test subjects during biodynamic experiments. The films are digitized and the 3-dimensional motion is reconstructed and analyzed.

The procedures and programs used to convert the digitized photo data into a format compatible with a large-scale general purpose computer (UNIVAC 1100®), to plot the data for identification of data errors, and to prepare the data for subsequent processing have recently undergone major revision. The objectives of the revision were to:

1. Modify existing software to be compatible with announced changes in the UNIVAC operating system.

2. Use new graphics support capabilities to improve error detection procedures, reduce turn-around time of analysis, and produce output compatible with all available graphics devices.

3. Improve efficiency of operations personnel.

4. Standardize data formats and I/O access methods to be compatible with existing archival data.

5. Develop a modular organization of software which provides greater operational flexibility.

This report documents the new procedures and programs.

DESCRIPTION OF SOFTWARE

Preparation of digitized photo data for subsequent reduction and analysis requires a process consisting of several tasks performed under computer control. Software supporting each task conforms to the principles of modular design so that task sequences may be selected to meet specific needs of the data being processed.

In order to process digitized photo data through a data reduction flow, major tasks are selected and merged in a single runstream. The three available programs are:

1. Convert photo data from PDS (Photo Digitizing System) NOVA 800® format to UNIVAC 1100/83® format for subsequent analysis.
2. Produce film frame plots and x-y trajectory plots of target data in order to provide graphical presentation for error analysis.

3. Convert data digitized manually into the format of data tracked under computer control.

Each of these components is documented in detail in the body of this presentation. It is assumed that the reader is familiar with the operating system of the UNIVAC 1100/83® computer (see reference 3) and with the UNIVAC ASCII FORTRAN programming language (see reference 4). Although the software presented herein is intended for specific computers, plotting devices and software support packages, it has been designed to minimize the conversion effort for other systems.

1. Photo Data Conversion Program

   A. Introduction. This section describes a UNIVAC 1100/83® program which converts photo data from PDS (Photo Digitizing System) data tapes to UNIVAC 1100/83® format. In converting data to a format compatible with a large-scale general purpose computer (UNIVAC 1100/83®), the photo data conversion program provides a suitable environment for efficient data reduction and graphical presentation.

   B. Discussion. Input to the conversion program is a PDS data tape. It contains digitized photographic data from at least one experiment, or run. For a particular run, as many as three cameras are used to photograph targets on anatomical mounts strapped to the experimental subject. Each reel of film generated from these cameras during the run is digitized (on the NOVA 800® system), with the output being written to a PDS data tape. Data for several runs may be written to a single PDS tape. This tape contains four record types. The first is a header record, which contains identification information for film from a particular camera used during a run. This record includes such items as run number, PDS record type, camera site number (1, 2 or 3), julian date the film was digitized and time-of-day for the first film frame (all character data on the PDS tape is in ASCII format and numeric data is in twos complement integer format). The second record type is the time record, which identifies time-of-day for each film frame, or data point. The time data values originate from LED displays on the film frames themselves. There are 250 data points on a time record (the first point being the time at first motion of the subject). Identification information for the reel of film the time record describes is also included in the record, i.e., run number, PDS record type and camera site number. The final two record types are the X and Y data records, respectively. Each contains 250 data points measuring film plane displacement (inches) from a stationary origin on the film. An X and Y data record exists for every photo target tracked during the run. Each data record contains (in addition to displacement values) run number, PDS record type, camera site number and a target number. The target number contains information describing the anatomical mount type on which it is located. For instance, there are head mounts, neck mounts, mouth mounts and pelvic mounts - all designed to track motion of a particular segment of the body.
Data on a PDS output tape is organized by reels of film. That is, all data for a particular reel of film (with its unique header, time and data records) are grouped together into a file. The order in which the records appear is always the same: header record, time record, target 1 (x values), target 1 (y values) ••• target N (x values), target N (y values), where N = number targets digitized on film. In some experiments more than the standard 250 frames are digitized. In this instance the data records for the extra frames (always a multiple of 250) are written to the PDS tape immediately after the final data record containing data for the previous 250 frames. An end-of-file separates each group of data records for a particular set of 250 frames. A double end-of-file signals the end of data on the tape.

The conversion program reads the NOVA 800® input tape and stores it in an input buffer. The PDS characters are in 8-bit ASCII format and the numeric data is either in 16-bit or 32-bit integer format. The conversion program determines the format of the information to be converted from the format specifications of the PDS input tape (see Figures 1b-1d). All character data is converted from 8-bit NOVA ASCII to 9-bit UNIVAC ASCII format. Numeric data is either converted to 36-bit UNIVAC floating point format or 36-bit UNIVAC integer format, depending on the format type of the information being converted. For instance, PDS record type is converted from NOVA 16-bit integer to UNIVAC 36-bit integer, while displacement (data) values are converted from NOVA 32-bit integer to UNIVAC 36-bit floating point format. All converted displacement values are then scaled from inches to meters. This is done to provide data compatibility with a group of data analysis programs collectively called EASYFLOW®.

The converted and scaled photo data must be output in a standard file format suitable to subsequent processing; in addition this data must be output in FIELDATA format, which is the NBDL archival standard. This second requirement presents difficulties in an ASCII FORTRAN programming environment. In order to provide for both needs an I/O access package (see Reference 1) has been developed to (1) provide standard I/O access methods for an established photo data file format and (2) maintain data format compatibility with the NBDL archives. The I/O package provides directory maintenance for the user. The conversion program utilizes this capability to output the standardized direct access record. This directory contains unique run identification numbers (for all runs on the output tape) and accompanying sector locations for both header and time records for the run. The directory provides means for rapid direct access of photo data for a given run number.

* EASYFLOW is an NBDL term representing a data reduction and analysis system requiring input data of a specific format. EASYFLOW uses converted photo data to derive position coordinates in 3-dimensional space of targets affixed to anatomical mounts (which are strapped on to the experimental subject). These data are subsequently used to compute and plot displacement, linear (and angular) velocity and linear (and angular) acceleration of the subjects.
There are four record types, in addition to the directory, which the conversion program outputs to a UNIVAC mass storage file. The first of these is the header record. There exists a unique header record for each camera site used in a run. There are no more than three and no less than two cameras used in a run, each assigned a site identification number of 1, 2 or 3. This site ID and the run number in concert make the header record unique. A keyword is stored in the header record along with identification information such as the run number and site ID. The keyword identifies the record as a header. The second type of output record is the time record. It contains the time-of-day information for each of 250 film frames. In addition the time record includes the run number, the site ID and a keyword for identification. The third record type is a photo data record. It contains displacement values (from a fixed point on the film plane) in meters of a particular photo target on the reel of film. Also included are the run number and site ID of the film, along with a keyword identifying the record type. This keyword also contains information identifying the anatomical mount (on which the photo target resides) and the axis type of the data (X or Y). The combination of header record, time record and data records (for all targets on the reel of film) is written to disk, in the order listed, by the conversion program. Data for each reel of film with a camera site ID of 1 are grouped together on the output file and separated by double ends-of-file*. An additional end-of-file is written after the final data record of the final reel of film for site ID 1. This signifies end-of-data. However, the direct access table follows this final end-of-file. It is located at the end of the file so as not to limit its size. In order to provide access to the directory a "locator table" is written as the initial output record. It contains the sector address of the directory.

The above output file structure is duplicated for camera site 2 (and 3 if necessary). A separate UNIVAC disk file is thus created for each site ID. Each of these files is in a standardized format, and serves as input to a UNIVAC runstream designed to write them to a UNIVAC 1100/830 data tape.

The conversion program is designed to correct three common situations in which PDS operator error has occurred. These three special cases are:

1. The run number of a header record for a reel of film is entered incorrectly by the PDS operator and written to the PDS data tape.

2. A reel of film is re-digitized by the PDS operator and output to the end of the same PDS data tape as data from the original reel of film.

3. The camera site ID of a header record for a reel of film is entered incorrectly by the PDS operator and written to the PDS data tape.

* If more than 250 frames of data were digitized on a reel of film, each group of data records for the extra frames (always a multiple of 250) is written in succession and separated by single end-of-file.
In cases (1) and (3) the conversion program uses correction fields in card image input in order to scan the header record for the incorrect entry and then correct it. This is done prior to writing the header to disk. In case (2) the conversion program scans for the first occurrence of the data file for a reel of film. Once found, this data file (header, time and data records) is bypassed and thus not converted and output. However, the second occurrence of a data file for the reel of film is converted and output in the usual manner. The photo conversion program accomplishes the following objectives:

(1) Converts photo data to a format compatible with a large-scale general purpose computer (UNIVAC 1100/83*).

(2) Scales displacement data values from inches to meters in order to provide data compatibility for subsequent processing.

(3) Provides for correction of common PDS operator errors.

(4) Accomplishes output of data with standardized I/O access methods.

(5) Maintains data format compatibility with the NBDL archives.

C. Main Program and Subroutine Descriptions:

(1) MAIN (see Appendix 1b for listing) - This is the driver program which controls all program operations. All correction cards are read by this program and used to rectify the three special cases earlier noted. MAIN then calls subroutine CONVT in order to read and convert the first PDS input record (see Figures 1e-1g for input record formats). The rest of the input records (non-header) for the run are read and converted by virtue of calls to subroutine CNVT1. Here the converted records are written to mass storage. Once a run has finished processing, control returns to MAIN and the process repeats itself until all runs have been processed. At this point a triple end-of-file, denoting end-of-data, is written to the output file.

(2) CNVT1 (see Appendix 1c for listing) - This subroutine controls the conversion and output of all data for each camera. The main program reads the header records and transfers control to this subroutine for reading and conversion of the position and time records. The position data is converted from inches to meters*.

Each header, time and position record is written to mass storage by a call to the photo I0 subroutine PUTRCD (see Appendix 4c). Data for each site ID is written to a separate mass storage unit (see Figure 2a). As mentioned earlier, the necessary data directories are maintained and written by the photo I0 subroutines. After processing an entire run, CNVT1 writes a double end-of-file for each output unit and returns control to the calling program.**

* During the scaling process all data values of 0.0 (points not digitized) are set to 999.0

** If more than 250 frames were digitized for each target in a run, only a single EOF (End-of-file) is written after the final data record for each set of 250 frames.
(3) CONVT (see Appendix 1d for listing) - This subroutine is called by CNVT1 to perform the actual conversion of data from PDS format to UNIVAC 1100/83® format. It is called after each read of a PDS input record. All character data is converted from 8-bit ASCII to 9-bit ASCII format. Numeric data is converted from 32-bit binary integer to UNIVAC 1100/83® 36-bit binary floating point format (see Figures 2b-2f for output record formats).

In this subroutine the sign bit is extended for negative numbers; these negative values are data entered manually by the PDS operator by using the X,Y crosshair input.

(4) SPHAVG (see Appendix 1e for listing) - This subroutine calculates the X and Y position of a stationary reference target relative to the film sprocket hole for each camera used in a run. Sprocket hole values for five consecutive frames are averaged, the result being subtracted from the averaged sled (chair) displacements for the same five frames in order to produce the final value. The X and Y sprocket hole averages are printed for each camera and in those cases where there is no sprocket hole information on the PDS tape, the values can be read from the film by the PDS operator and calculated by hand. Execution of EASYFLOW is expedited by providing sprocket hole averages at the time of conversion. This information is required for an internal consistency check on the data collection and system.

(5) ASC8T9 (see Appendix 1f for listing) - This subroutine converts a string of 8-bit ASCII characters to a UNIVAC 1100/83® compatible 9-bit ASCII character string. All character data read from the PDS input tape is converted by the subroutine. The high order bit of a 9-bit ASCII character is always off, this bit being the only difference between an 8-bit ASCII character and a UNIVAC 1100/82 9-bit character.

2. Plot Program for Photo Data

A. Introduction. This section describes an ASCII FORTRAN UNIVAC 1100/83® program which plots UNIVAC-compatible photo data as (a) X-Y film frame plots and (b) X-Y trajectory plots. These plots provide a method for error detection of faulty photo data, allowing for correction of the data before any further processing takes place.

B. Discussion. The plot program generates both x-y film frame plots and x-y trajectory plots. The film frame plots (see Figure 5a) plot the position of each photo target (on a reel of film) relative to the film frame origin. Each individual frame of film is plotted, and all target positions in the frame are connected by straight lines; this aids in visual recognition of relative changes in target positions from frame to frame. There is sometimes more than one anatomical mount (see Figure 6 for mount example) visible to a camera during a run. It is for the reason that a separate set of film frame plots are generated for each anatomical mount found on a reel of film. For instance, if two mounts are visible to a particular camera and only one mount is visible to a second camera, three sets of film frame plots are generated for the experiment. A total of 250 film frames are plotted for each set,
which consists of ten pages of twenty-five plots each. On top of each page the plot scale (meters) is listed; this measures the greatest target displacement from the first to last frame on the page. In addition all target positions which were manually digitized (not automatically tracked) by the PDS operator are circled for identification. The film frame plots are used to identify errors which may exist in the data. For instance, a target may have been identified incorrectly by the PDS operator; this problem is visually exposed in the film frame plots. Subsequently the data file is corrected.

The x-y trajectory plots (see Figure 5b) identify the path each target (on a reel of film) travels in the film plane during a run. Each individual point on the plot measures target displacement from a fixed film frame origin. A total of 250 data points are plotted, with the first coinciding with time of first motion of the experimental subject. Since more than one anatomical mount may appear on a reel of film, a separate x-y trajectory plot is generated for each mount on the film. The path of every photo target on a mount (visible to the camera) is plotted, each identified by its own target number. The x-y trajectory plot provides at a glance a visual presentation of target movement (for a given mount) for the entire run duration; this allows for quick detection of data discontinuities. For instance, if the pattern recognition algorithms (used in the photo digitizing process) found the wrong target, a detectable discontinuity results in the x-y trajectory plot.

The plot program for photo data possesses versatile plot output capabilities, made available through the use of DISSPLA® plot software (a product of ISSCO®). All plot commands are output to a catalogued UNIVAC disk file (via DISSPLA® software). This "compressed" plot file can then be used as input to a variety of DISSPLA® post-processor programs, which "decompress" the plot commands and direct them to a plotting device. This disk file may be utilized several times on a variety of plotting devices without once repeating execution of the actual plot program. DISSPLA® post-processor programs exist for such plotting devices as the TEKTRONIX 4014®, FR80® and HP 7220® (all available to NBDL).

There are advantages in using any of these post-processors. For instance, the FR80® post-processor is used in a production environment to generate microfiche and high quality hardcopy. The TEKTRONIX 4014® post-processor can be used to view selected plots on the TEKTRONIX 4014® scope immediately after execution of the photo data plot program. This quick turnaround is of benefit in expedient data analysis. The FR80® is not located on-site, so in order to combine the benefits of quick turnaround and high quality hardcopy an HP 7220® plotter is to be installed on-site. This plotter in addition affords a multi-color plotting capability.

* Integrated Software Systems Corporation
The input to the plot program is UNIVAC-compatible photo data. Character data is in 9-bit UNIVAC ASCII format and numeric data is either in 36-bit UNIVAC integer format or 36-bit UNIVAC floating point format (depending on format specifications). As noted in section 1.B data is organized on the input files by film reel, with a separate input file for each camera site. A directory exists in each input file, containing the run numbers (each run number identifies a unique film reel) and accompanying sector locations of the header and time records. The data records for a run (or film reel) immediately follow the time record, and a "locator table" for the directory precedes all other records on the file. Input is accomplished with a general purpose I/O package (see Reference 1) which reads FIELDATA (NBDL archival standard) in an ASCII FORTRAN programming environment. Plots are output for each input file (or camera site). Within each file, plots are output by run number (or film reel) and within each run plots are output by anatomical mount.

The PDS operator is responsible for entering time-of-day values (LED displays) for each frame of digitized film. To insure their legitimacy the plot program performs tolerance checks for the time values. The results (see Figure 5c), called a time record analysis, are output to a line printer*. The plot program for photo data accomplishes the following:

1. Provides error detection methods for photo data by generating:
   (a) x-y film frame plots.
   (b) x-y trajectory plots.

2. Performs confidence checks for time-of-day values for film frames and prints the results.

3. Provides plot flexibility features through the use of post-processor programs.
   (a) Plots can be generated repetitively without repeat execution of the plot program.
   (b) Plots can be viewed immediately upon execution on a demand terminal scope.
   (c) Quality hardcopy and microfiche of plots are generated.

4. Accomplishes input of data with standardized I/O access methods.

* This analysis may be output to a plotting device, but it is slower and more expensive to do so.
C. Main Program and Subroutine Descriptions:

(1) PLOT (see Appendix 2b for listing) - This is the driver program which controls all program operations. A single input card identifies (a) the number of input files, (b) the time record analysis option, (c) the X-Y contour plot option, (d) the frame plot option, (e) the frame plot format option, (f) the lowest run number to be processed and (g) the highest run number to be processed (see Appendix 2a for card format). Once the input card is read, DISSPLA® is called to initialize the compressor. Plotting is accomplished by virtue of a loop which controls the input and subsequent plotting of photo data for each combination of run/camera site/anatomical mount (see Figures 2b-2f for input record formats).

(2) TIMER (see Appendix 2c for listing) - This subroutine generates a time record analysis for a given run number and site ID. The time record is input via the photo data I/O program (see reference 1). A time interval between frames is approximated by averaging total elapsed time over the 250 frames of data. This increment is used to calculate an "expected time" for each of 250 frames. These times are compared to the actual time record data points in order to test for excessive error (a difference between actual and "expected" time greater than .001 second). Any such discrepancies are annotated with an asterisk (*) on the time record analysis (see Figure 5c).

(3) LDATA (see Appendix 2d for listing) - This subroutine loads photo displacement data into common for access by plot subroutines. This routine is called once for each combination of run/camera/mount. The photo data I/O package is used to accomplish input (in order to read FIELDATA within an ASCII FORTRAN program).

(4) MIXY (see Appendix 2e for listing) - This subroutine determines minimum and maximum X and Y values for a page (twenty-five frames) of frame plots. These values are placed in common for scaling use by the plot subroutines. The minimum and maximum values are selected so that the abscissa and ordinate axes are equal lengths in data units (meters). This allows the viewer of the plots to determine at a glance in which direction the targets were moving fastest. For example, if the targets moved from the far left of the initial frame on the page to the far right of the final (25th) frame, the X filmplane displacement was changing more rapidly than the Y displacement.

(5) PLOTD (see Appendix 2f for listing) - This subroutine consists of two entry points. All plotting in both entry points is done with DISSPLA® software.

A. PLOTI - This entry point plots the title information for each page of frame plots. Included in the title information are the (i) run number, (ii) camera number, (iii) plot scale (meters), (iv) anatomical mount type and (v) page number. All plotting is done with DISSPLA® software.
B. PLOT2 - This entry point plots all frame plots. It is called once for every frame of photo data to be plotted. Two-hundred fifty frames are plotted for each mount, twenty-five frames/page, ten pages/mount. The target number for all targets tracked (by the PDS operator) in a particular frame are printed directly at their digitized position. Those targets digitized manually are circled for identification. All targets in a particular frame are connected by vectors in order to present the image of motion as the targets begin to change position from frame to frame. Targets may have been deleted or added as the frames advance. Deleted target numbers are printed in the bottom right corner of the frame in which the target was dropped. Added targets are printed in the bottom left corner of the frame in which the target was added. If chair data are available for the current run/camera, a 'C' is printed in the top left corner of each frame. The frame number is printed in the top right corner of the frame.

(6) XYPLOT (see Appendix 2g for listing) - This subroutine plots X vs. Y filmplane displacement values for a given run/camera/anatomical mount. All targets digitized for the mount are plotted on the same graph. A maximum of 250 data points are plotted (and connected) for each target, with the target number appearing coincident with the first point for a curve. The maximum and minimum values of the abscissa and ordinate are printed on the plot. In addition, titles information is listed which includes (i) run number, (ii) camera number, (iii) anatomical mount type and (iv) current date. The X-Y contour plot affords the viewer a quick method of isolating data incongruities.

3. Reformatting (sign change) Program for Photo Data

A. Introduction. This section describes an ASCII FORTRAN UNIVAC 1100/830 program which converts UNIVAC-compatible photo data, digitized manually, into the format of data tracked under computer control. This prepares the data for subsequent computation of 3-dimensional trajectories of test subjects.

B. Discussion. This program is designed to read all target position data for a given range of run numbers and scan for negative data values. Each negative value represents an entry made manually by the PDS (Photo Digitizing System) operator using the x-y crosshairs. All such values are multiplied by -1 and rewritten to the data file. An archival data tape is later generated (with the reformatted data records) by an independent runstream. This tape serves as input to the EASYFLOW runstream.

Input to the reformatting program is UNIVAC-compatible photo data. Character data are in 9-bit UNIVAC ASCII format and numeric data are either in 36-bit UNIVAC integer format or 36-bit UNIVAC floating point format (depending on format specifications). Data are organized on the input files by film reel (or run number), with a separate input file for each camera site (see section 1.8 for a more detailed description). Direct access tables accompany each input file. Input/output is accomplished with a general purpose I/O package (reference 1) which reads/writes FIELDATA (NBDL archival standard) within the context of an ASCII FORTRAN programming environment. Output file formats for the reformatting program are identical to the input file formats, the only difference being that all values in the output data records are positive.
The reformatting program accomplishes the following:

(1) Prepares UNIVAC-compatible photo data for subsequent computation of three-dimensional trajectories (of test subjects) by converting manually digitized photo data into the format of computer-tracked photo data.

(2) Accomplishes input/output of data with standardized I/O access methods.

C. Main Program Description:

SGFLIP (see Appendix 3b for listing) - This is the driver program which controls all program operations. The only subroutines utilized are those belonging to the photo data ASCII FORTRAN I/O program. A single input card identifies (a) the number of input files, (b) the lowest run number to be processed and (c) the highest run number to be processed (see Appendix 3a for card format). The number of input files and the run number range (from the input card) are used to index a loop which systematically (a) reads data records, (b) scans them for negative data points and (c) changes negative values to positive values. Once a data record is completely processed it is rewritten to the input file unless no negative values were found (see Figures 2b-2f for input/output record formats).

4. Operation of Photo Data Reduction System

A. Introduction. This section describes the actual UNIVAC 1100/830 production runstream which controls data reduction processing of NBDL photo data prior to its input to the EASYFLOW runstream.

B. Discussion. As previously noted, there are three main processing steps in the photo data reduction process: conversion, plotting and reformatting. Each step performs an integral function independent of the other steps. This modular organization of steps provides several benefits:

(1) Any step can be performed independently from the others; i.e., each step has stand-alone capability.

(2) Modifications to steps can be effected with:

(a) Complete non-interference with other steps.

(b) Expediency in that all necessary changes take place in a single module.

(3) Input/output may be examined at any point between steps.

(4) Additional steps may be added to the overall process without altering any existing steps, thus providing a desirable environment for enhancements to the system.
Each step, or module, shall heretofore be referred to as an element. Element is a UNIVAC term describing a collection of computer commands. When it is desired to execute an element, or step, the computer is instructed to carry out all commands in that element. The production runstream described in this section executes a collection of three such elements:

1. **XQTCVT** is the conversion element.
2. **XQTPLT** is the plot element.
3. **XQTSGN** is the reformatting (sign change) element.

The first element to be executed is the conversion element XQTCVT (see Appendix la). The main function of this element is to execute the photo conversion program. Additionally, the output of the conversion program is written to a UNIVAC data tape. In sequence, the conversion element (1) copies a PDS data tape onto disk, (2) executes the photo conversion program and (3) writes the output (from the conversion program) to a UNIVAC data tape. The UNIVAC tape is saved for 10 days as a precautionary measure. The key output from the conversion element is the converted photo data written to disk by the conversion program.

If it is necessary to correct for PDS operator error (see section 1.8.) the conversion element is altered to include correction options. These options are input to the element in card image via the UNIVAC EDIT processor (see reference 5). The EDIT processor commands are part of the production runstream. Thus the runstream actually edits the conversion element before executing it. The correction options allow for rectification (by the conversion program) of instances where the PDS operator:

1. Enters incorrect run numbers on PDS header records.
2. Outputs data to a PDS tape for two film reels with the same run number and camera site ID (on the header record).
3. Enters incorrect camera site ID's on PDS header records.

The first card image of options (input with the EDIT processor) indicates the number of film reels to be corrected for each of the three categories of problem cases. A card image then follows for each occurrence of a problem case (1); it specifies the run number and camera site ID of (the header record for) the problem film reel as well as the occurrence (of this unique combination) to be corrected. It also specifies the correct run number to replace the incorrect field in the header record. After the last correction card image for case (1), card images follow for case (2). Each of these specifies the run number and camera site ID of the problem reel. The first occurrence of this combination of run number and site ID (in a PDS header record) is bypassed and only the second occurrence (of PDS records with this header identification) is converted to UNIVAC format. Lastly, the correction card images for case (3) follow. Each specifies the run number, camera site ID and
occurrence index for the problem reel. Additionally, it contains the correct camera site ID. If no corrections are necessary on the PDS input tape, only one card image is input. It indicates quantity zero for each of the three problem cases.

The EDIT processor not only controls input of all correction options to the conversion element, but it also is used to enter the reel number of the PDS data tape to be used as input to the photo conversion program. As with the EDIT commands for correction options, these commands also constitute a portion of the overall production runstream.

The second element of the runstream to be executed is the plot element XQTPLT. Its primary function is to execute the plot program for photo data. Additionally, the disk file of DISSPLA® plot commands (output from the plot program) is used as input to the FR80® post-processor program to produce production quality microfiche and hardcopy of all plots. In sequence, the plot element (1) executes the plot program for photo data (the plot program uses photo data residing on disk as input), (2) executes the FR80® post-processor program and (3) writes the disk file of plot commands to a catalogued disk file for public access. As previously noted the plot program uses output from the conversion program as its input. This input consists of disk files of photo data, one for each camera site (1, 2 or 3) used in a series of runs. Output from the plot program, a disk file of DISSPLA® plot commands, serves as input to the FR80® post-processor. The post-processor interprets the plot commands in order to produce microfiche and hardcopy of all plots generated by the plot program (these include the x-y film frame plots and x-y trajectory plots). These plots are used in error detection analysis and are maintained on-site.

Certain plot options exist and are input to the plot element via the UNIVAC EDIT processor. The EDIT commands constitute part of the production runstream (see Figure 8); they are used to edit the plot element before executing it. A single card image contains all the plot options. These options include:

(1) Number of input files.

(2) Time record analysis option.
   (a) Output to line printer.
   (b) Output DISSPLA® plot commands to disk file.
   (c) Do not output to any device.

(3) X-Y trajectory plot option.
   (a) Output DISSPLA® plot commands to disk file.
   (b) Do not output to any device.
(4) X-Y film frame plot option.
   (a) Output DISSPLA© commands to disk file.
   (b) Do not output to any device.

(5) Plot direction option for x-y film frame plots*.
   (a) Plot five columns to a page with frame number increasing in each
column (plot direction always downward).
   (b) Plot five columns to a page with frame number increasing in
columns 1, 3 and 5 (plot direction downward) and decreasing in columns 2 and 4
(plot direction upward).

(6) Lowest run number to be plotted

(7) Highest run number to be plotted

In an input file each run number identifies the header record for a single film reel;
photo data is plotted for this range of run numbers (or film reels).

Once the plot program has been executed, the plot element executes the FR80©
post-processor program to produce microfiche and hardcopy of all generated plots. The plot (disk) file is then copied to a catalogued disk file.

The third and final element of the production runstream is the reformatting
element. The primary function of this element is execution of the refor-
mattting (sign change) program. The element also generates a UNIVAC data tape
containing output from the reformatting program. This data tape is saved for
180 days and is used as input to the EASYFLOW data reduction and analysis
system. In sequence, the reformatting element (1) executes the reformatting
program and (2) saves the output on a UNIVAC data tape. The reformatting pro-
gram uses photo data residing on disk (output from the conversion program) as
input. These same files, after being reformatted to reflect automatically
digitized photo data, are output to the UNIVAC data tape.

The UNIVAC EDIT processor is utilized (in the production runstream) to input
(1) reformatting program options and (2) the output tape file name. The input
options include:

(1) Number of input files.

(2) Lowest run number to be processed

(3) Highest run number to be processed

In an input file each run number identifies the header record for a single film reel;
photo data is reformatted for this range of run numbers (or film reels).

* This option is used only in the event x-y film frame plots are generated.
C. Production Runstream. The photo data reduction runstream is a collection of computer commands designed to execute the three steps (elements) in the data reduction process. The runstream consists of (1) UNIVAC Operating System commands (see reference 3) and (2) UNIVAC EDIT processor commands (see reference 5). All three elements to be executed are stored in a single program file. The runstream (1) copies the program file onto a scratch disk file, (2) enters the input options for each element and (3) executes each element. The only input to the runstream is a single PDS data tape containing digitized photo data. Output from the runstream includes:

1. A 10-day UNIVAC data tape of converted photo data.
2. Catalogued disk file containing DISSPLA® plot commands for all generated plots.
3. Microfiche and hardcopy of all generated plots.
4. A 180-day UNIVAC data tape of converted and reformatted photo data.
5. Line printer listing containing:
   (a) Time record analyses.
   (b) Operating system diagnostics.
   (c) User-generated diagnostics.
   (d) Operating system accounting information.

Several considerations entered into the design of the photo data production runstream. The more significant of these are noted as follows:

1. In storing plot commands on a catalogued disk file, plots can be generated as many times as necessary (and on several plotting devices) without once repeating execution of the actual plot program.

2. In some instances (a film reel of) photo data is redigitized in order to improve data quality. It is important (in computing 3-dimensional position of the test subject) that a minimum total of three targets (not all from the same camera site) are digitized successfully for a given test run; these targets may be tracked from any of the available film reels (camera sites) for the run. If the PDS computer cannot successfully locate enough targets (through computer-controlled tracking), the PDS operator generates another data tape in which (s)he manually digitizes (with the x-y crosshairs) the necessary targets.
The production runstream is designed to reprocess the new PDS data tape with a minimum of operator preparation. It is also equipped with the option to process redigitized data residing on the same PDS tape as the initial data (for the same film reel). The production runstream also creates UNIVAC data tapes directly after (a) conversion and (b) reformatting in order to make data available for further analysis, error correction, and reprocessing.

(3) The individual cameras (used in the experiments) are calibrated to provide mathematical camera constants vital in the computation of three-dimensional position (of the test subject). Examples of camera constants are the camera nodal point coordinates \((x, y, z)\) and the lens distortion parameter. These values and other camera constants are generated by a camera-calibration program which uses as input converted data from a camera-calibration film reel. The photo data production runstream is designed to, with slight modification, execute the camera-calibration conversion program. This altered runstream is included as an element in the program file containing all photo data reduction programs.

(4) The modular organization of the steps (elements) in the photo data reduction process allows for efficient and expedient expansion of functions. New elements can be added without changing any existing elements. This feature affords the system a desirable flexibility. Standardized data formats and I/O access methods provide further flexibility. Possible additional functional elements are listed below:

(a) Error editing elements which correct improper data values in the data file.

(b) Packing elements which combine data files (and accompanying directories) from multiple tapes onto a single tape (with one all-inclusive directory).

(c) Plot elements which allow plot selection of individual photo variables residing on a converted photo data file.

(d) Special analysis elements which extract the data easily through the use of standardized data file formats and standardized I/O access methods.

Operational particulars of the runstream are as follows:

(1) **Program File:**

    DATASYSTEMS*PHOTOSTREAM.

(2) **Programming Languages:**

(a) UNIVAC ASCII FORTRAN Level 10R1

(b) UNIVAC Executive System EXEC Level 38R5

(c) UNIVAC Text Editor Level 16R1
CONCLUSION

The photo data reduction and analysis system has been revised to successfully meet all proposed objectives:

1. All constituent programs are now compatible with announced changes in the UNIVAC operating system. These changes incorporate an ASCII FORTRAN programming environment.

2. A new graphics support capability is now utilized to:
   a. Improve error detection procedures by providing additional plot presentations.
   b. Produce output compatible with a variety of graphics devices (such as the Tektronix 4014 and the Hewlett Packard 7220).
   c. Reduce turn-around time of analysis by providing for immediate review of plots.

3. A stand-alone runstream providing for greater operator efficiency is now in effect. In addition, all component programs exist in a single program file.

4. Standardized data formats and I/O access methods, which in unison provide compatibility with stored archival data, are now incorporated into photo data reduction processing.

5. A modular organization of software is now incorporated in order to provide for greater operational flexibility.
PDS OUTPUT TAPE FORMAT

RECORD SEQUENCE

801

Header
Time
Target 1, x values
" y values
Target 2, x values
" y values
First 250 frames
Target N, x values
" y values
Complete Date for One Film
Repeat above format if more than 250 frames are being digitized
End-of-File
Repeat for each film

End-of-File
End-of-File (double EOF:end of data on tape)

March 29, 1979

Figure 1a
### PHOTO HEADER RECORD FORMAT

<table>
<thead>
<tr>
<th>Word 1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>Bit</td>
<td>35</td>
<td>0</td>
<td>35</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Word</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RECORD TYPE (INTEGER 1 = HEADER)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TEST ID (6 CHAR ASCII CONVERTED TO FDATA)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 DIGIT TIME (32 BIT INTEGER)</td>
<td>I36(1)</td>
<td>I36(2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOT SAVED</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Word 7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>Bit</td>
<td>35</td>
<td>0</td>
<td>35</td>
<td>0</td>
<td>35</td>
</tr>
<tr>
<td>Word</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JULIAN DATE (16 BIT INTEGER)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1000 X MACH. + CAM SITE (16 BIT INTEGER)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Δt BETWEEN FRAMES (16 BIT INTEGER)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PDS VERSION NO. (16 BIT INTEGER)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOT USED</td>
<td>I36(4)</td>
<td>I36(3)</td>
<td>I36(5)</td>
<td>I36(6)</td>
<td></td>
</tr>
</tbody>
</table>

FIELD DESCRIPTION

UNIVAC OUTPUT BUFFER (I36) ARRAY LOCATION

FIGURE 1b

28 Mar 79
PHOTO TIME DATA RECORD FORMAT

Word 1  2  3  4  5  6  7
Bit 0  8  0  8  0  8  0  8  0  8  35 0  35 0

NOVA FORMAT

UNIVAC INPUT BUFFER

FIELD DESCRIPTION

UNIVAC OUTPUT BUFFER
(I36) ARRAY LOCATION

Word 1

<table>
<thead>
<tr>
<th>RECORD TYPE (INTEGER 2 = TIME)</th>
<th>TEST ID (6 CHAR ASCII CONVERTED TO FLDATA)</th>
<th>'bb' (2 char ASCII)</th>
<th>'TI' (2 char ASCII)</th>
</tr>
</thead>
<tbody>
<tr>
<td>I36(1)</td>
<td>NOT USED</td>
<td>NOT USED</td>
<td>NOT USED</td>
</tr>
</tbody>
</table>

Figure 1c

Word 7

<table>
<thead>
<tr>
<th>Word 7  8  9  10  11  12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 0  8  0  8  0  8  35 0 35 0</td>
</tr>
</tbody>
</table>

NOVA FORMAT

UNIVAC INPUT BUFFER

FIELD DESCRIPTION

UNIVAC OUTPUT BUFFER
(I36) ARRAY LOCATION

<table>
<thead>
<tr>
<th>'ME' (2 CHAR ASCII)</th>
<th>'bbbbbb' (6 CHAR ASCII)</th>
<th>5 DIGIT TIME (32 BIT INTEGER)</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOT USED</td>
<td>NOT USED</td>
<td>I36(4)</td>
</tr>
</tbody>
</table>

REPEAT FOR TOTAL OF 250 DATA VALUES

Figure 1c

28 Mar 79

20
### PHOTO X (OR Y) DATA RECORD FORMAT

<table>
<thead>
<tr>
<th>Word</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### NOVA FORMAT

- - - - - - - -

#### UNIVAC INPUT BUFFER

- - - - - - - -

#### FIELD DESCRIPTION

- RECORD TYPE INTEGER 3=X
- TEST ID (6 CHAR ASCII CONVERTED TO FLDATA)
- TARGET NUMBER (32 BIT INTEGER)
- NOT SAVED
- I36(1)
- I36(2)

#### UNIVAC OUTPUT BUFFER (I36) ARRAY LOCATION

- - - - - - - -

---

### Word 7

<table>
<thead>
<tr>
<th>Bit</th>
<th>0</th>
<th>8</th>
<th>0</th>
<th>8</th>
<th>0</th>
<th>8</th>
<th>0</th>
<th>8</th>
<th>0</th>
<th>8</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Word</td>
<td></td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### NOVA FORMAT

- - - - - - - -

#### UNIVAC INPUT BUFFER

- - - - - - - -

#### FIELD DESCRIPTION

- 1000 X MACH. + CAM SITE (32 BIT INTEGER)
- X (OR Y) DATA VALUE (32 BIT INTEGER)
- X (OR Y) DATA VALUE (32 BIT INTEGER)
- I36(3)
- I36(4)
- I36(5)

#### UNIVAC OUTPUT BUFFER (I36) ARRAY LOCATION

- REPEAT FOR TOTAL OF 250 DATA VALUES

---

**FIGURE 1d**

21

28 Mar 79
## HEADER FORMAT (PDS DATA)

<table>
<thead>
<tr>
<th>16 Bit Word</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Record type: 1 (16 bit integer)</td>
</tr>
<tr>
<td>2</td>
<td>Test ID (6 character ASCII)</td>
</tr>
<tr>
<td>3</td>
<td>Time of first frame in units of the least significant LED digit (32 bit integer)</td>
</tr>
<tr>
<td>4</td>
<td>Julian date digitized: DDDY (16 bit integer)</td>
</tr>
<tr>
<td>5</td>
<td>1000 X machine # + camera site # (16 bit integer)</td>
</tr>
<tr>
<td>6</td>
<td>Δt: expected time between successive frames, in units of the least significant LED digit (16 bit integer)</td>
</tr>
<tr>
<td>7</td>
<td>Program version number (16 bit integer)</td>
</tr>
<tr>
<td>8</td>
<td>Not used</td>
</tr>
<tr>
<td>9</td>
<td>Not used</td>
</tr>
</tbody>
</table>

March 29, 1979

Figure 1e
### TIME RECORD FORMAT (PDS DATA)

<table>
<thead>
<tr>
<th>16 Bit Word</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Record type: 2 (16 bit integer)</td>
</tr>
<tr>
<td>2</td>
<td>Test ID (6 character ASCII)</td>
</tr>
<tr>
<td>3</td>
<td>'06 TIME 666666' (12 character ASCII)</td>
</tr>
<tr>
<td>4</td>
<td>Time of frame 1 in units of the least significant LED digit (32 bit integer)</td>
</tr>
<tr>
<td>5</td>
<td>Time of frame 250</td>
</tr>
<tr>
<td>6</td>
<td>Not used</td>
</tr>
<tr>
<td>7</td>
<td>Not used (fixed length record)</td>
</tr>
</tbody>
</table>

March 29, 1979
**X (OR Y) COORDINATE RECORD FORMAT (PDS DATA)**

<table>
<thead>
<tr>
<th>16 Bit Word</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Record type: 3=x values (16 bit integer) 4=y values</td>
</tr>
<tr>
<td>2</td>
<td>Test ID (6 character ASCII)</td>
</tr>
<tr>
<td>3</td>
<td>Target # (32 bit integer)</td>
</tr>
<tr>
<td>4</td>
<td>Julian date on film: DDDY (16 bit integer)</td>
</tr>
<tr>
<td>5</td>
<td>x (or y) value for target in frame 1: displacement from a fixed location in the filmplane in units of .00001 inches. Negative value indicates manual entry was made using the crosshairs. Zero indicates location of target was not defined for this frame. (32 bit integer)</td>
</tr>
<tr>
<td>6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>x (or y) value for target in frame 250</td>
</tr>
<tr>
<td>9</td>
<td>Not used</td>
</tr>
<tr>
<td>10</td>
<td>Not used (fixed length record)</td>
</tr>
</tbody>
</table>

March 29, 1979

Figure 1g
FILE FORMAT FOR UNIVAC

COMPATIBLE PHOTO DATA

RECORD SEQUENCE

<table>
<thead>
<tr>
<th>BOT</th>
<th>Location of random access table</th>
<th>5 words</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Header</td>
<td>7 words</td>
</tr>
<tr>
<td></td>
<td>Time</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Target 1, x values</td>
<td></td>
</tr>
<tr>
<td></td>
<td>y values</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Target 2, x values</td>
<td></td>
</tr>
<tr>
<td></td>
<td>y values</td>
<td></td>
</tr>
<tr>
<td></td>
<td>253 words (each record)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Target N, x values</td>
<td></td>
</tr>
<tr>
<td></td>
<td>y values</td>
<td></td>
</tr>
<tr>
<td></td>
<td>End-of-file</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Repeat above format if more than 250 words were digitized</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Repeat for each run</td>
<td></td>
</tr>
<tr>
<td></td>
<td>End-of-file</td>
<td></td>
</tr>
<tr>
<td></td>
<td>End-of-file</td>
<td></td>
</tr>
<tr>
<td></td>
<td>End-of-file (Triple EOF: end of data)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Random access table</td>
<td></td>
</tr>
</tbody>
</table>

SITE* ONE

* Above block of data copied from mass storage to tape using @ copy. There are 3 blocks of site one data copied to tape followed by 3 blocks of site 2 followed by 3 blocks of site 3.

May 5, 1982  

Figure 2a
RECORD FORMAT FOR LOCATION OF RANDOM ACCESS TABLE FOR UNIVAC COMPATIBLE PHOTO DATA

<table>
<thead>
<tr>
<th>WORD</th>
<th>CONTENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sector address of random access table (integer)</td>
</tr>
<tr>
<td>2</td>
<td>Number of runs in this block of data (integer)</td>
</tr>
<tr>
<td>3</td>
<td>Site ID (integer)</td>
</tr>
<tr>
<td>4</td>
<td>Not used</td>
</tr>
<tr>
<td>5</td>
<td>Not used</td>
</tr>
</tbody>
</table>

May 5, 1982
### HEADER RECORD FORMAT FOR UNIVAC

**COMPATIBLE PHOTO DATA**

<table>
<thead>
<tr>
<th>FIELD</th>
<th>CONTENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Run number (alpha - 6 characters) FLDATA</td>
</tr>
<tr>
<td>2</td>
<td>The word 'PHDATA' (alpha) FLDATA</td>
</tr>
<tr>
<td>3</td>
<td>Site ID (integer)</td>
</tr>
<tr>
<td>4</td>
<td>Time of first frame - 5 most significant digits (real)</td>
</tr>
<tr>
<td>5</td>
<td>Julian date: DDDY (alpha-6 char) Date digitized</td>
</tr>
<tr>
<td>6</td>
<td>$\Delta t$: expected time between frames (real)</td>
</tr>
<tr>
<td>7</td>
<td>Program version number (alpha-6 char)</td>
</tr>
</tbody>
</table>
### TIME RECORD FORMAT FOR UNIVAC COMPATIBLE PHOTO DATA

<table>
<thead>
<tr>
<th>FIELD</th>
<th>CONTENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Run number (alpha - 6 characters) FLDATA</td>
</tr>
<tr>
<td>2</td>
<td>The word 'CTSbbb' (alpha) FLDATA*</td>
</tr>
<tr>
<td>3</td>
<td>Site ID (integer)</td>
</tr>
<tr>
<td>4</td>
<td>Time of first frame, 5 least significant digits (real)</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>253</td>
<td>Time of 250th frame</td>
</tr>
</tbody>
</table>

* C is the camera site ID (1, 2, or 3)
DATA RECORD FORMAT FOR UNIVAC

COMPATIBLE PHOTO DATA

<table>
<thead>
<tr>
<th>WORD</th>
<th>CONTENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Run number (alpha - 6 characters)</td>
</tr>
</tbody>
</table>
| 2    | Displacement ID (alpha - 6 characters)  
     | (see Figure 4) |
| 3    | Site ID (integer) |
| 4    | X (or Y) value for target in frame 1: Displacement from a fixed location in meters. Negative value indicates manual entry was made using the crosshairs. A value of 999.0 indicates location of target was undefined for this frame. (REAL) |
|      |          |
|      |          |
|      |          |
| 253  | X (or Y) value for target in frame 250. |
## RANDOM ACCESS DIRECTORY FORMAT

**FOR UNIVAC COMPATIBLE PHOTO DATA**

<table>
<thead>
<tr>
<th>FIELD</th>
<th>CONTENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Words following in this record (integer)</td>
</tr>
<tr>
<td>1</td>
<td>RN(1) Run number for first run (alpha - 6 characters) FLDATA</td>
</tr>
<tr>
<td>2</td>
<td>SH(1) Sector address of header for first run (integer)</td>
</tr>
<tr>
<td>3</td>
<td>NR(1) Number of physical records of 253 words required to make one logical record for each target for first run. This occurs when more than 250 frames are digitized for a run. Output is written in multiples of 253 word physical records (integer)</td>
</tr>
<tr>
<td>4</td>
<td>ST(1) Sector address of time record of first physical record for first run (integer)</td>
</tr>
<tr>
<td></td>
<td>. . . . . .</td>
</tr>
<tr>
<td>NR(1)+3</td>
<td>ST(1,NR(1)) Sector address of time record of NR physical records for first run (integer)</td>
</tr>
</tbody>
</table>

For J number of runs where X is the number of words used by all previous runs

| 1+X   | RN(J) Run number for Jth run FLDATA |
| 2+X   | SH(J) Sector address of header for Jth run |
| 3+X   | NR(J) Number of physical records for Jth run |
| 4+X   | ST(J,1) Sector address of time record of 1st physical record for Jth run |
|       | . . . . . . |
| NR(J)+X+3 | ST(J,NR(J)) Sector address of time record of NR(J)th physical record for Jth run |

See Figure 3 for example
EXAMPLE OF RANDOM ACCESS TABLE WITH
SINGLE PHYSICAL RECORD FOR EACH TARGET

<table>
<thead>
<tr>
<th>RUN NO.</th>
<th>SH</th>
<th>NR</th>
<th>ST(1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LX3036</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>LX3037</td>
<td>134</td>
<td>1</td>
<td>135</td>
</tr>
<tr>
<td>LX3038</td>
<td>267</td>
<td>1</td>
<td>268</td>
</tr>
<tr>
<td>LX3039</td>
<td>400</td>
<td>1</td>
<td>401</td>
</tr>
<tr>
<td>LX3041</td>
<td>533</td>
<td>1</td>
<td>534</td>
</tr>
<tr>
<td>LX3042</td>
<td>666</td>
<td>1</td>
<td>667</td>
</tr>
<tr>
<td>LX3045</td>
<td>799</td>
<td>1</td>
<td>800</td>
</tr>
<tr>
<td>LX3047</td>
<td>932</td>
<td>1</td>
<td>933</td>
</tr>
<tr>
<td>LX3048</td>
<td>1065</td>
<td>1</td>
<td>1066</td>
</tr>
<tr>
<td>LX3050</td>
<td>1198</td>
<td>1</td>
<td>1199</td>
</tr>
<tr>
<td>LX3051</td>
<td>1331</td>
<td>1</td>
<td>1332</td>
</tr>
<tr>
<td>LX3053</td>
<td>1504</td>
<td>1</td>
<td>1505</td>
</tr>
<tr>
<td>LX3049</td>
<td>1637</td>
<td>1</td>
<td>1638</td>
</tr>
</tbody>
</table>
DISPLACEMENT ID (EZFLOW FORMAT)

COMTTb  6 char ID  (Field data)

where

C:  camera site ID (1, 2, or 3)
D:  coordinate component (X or Y) of data
M:  mount system
    1:   neck*
    2:   mouth
    3:   top of head
    4:   pelvis
    5:   other

9:   sprocket hole**

TT:  target ID (01, 02, 03, ...)

*  The chair reference target has the special mount/target ID of 100
** The sprocket hole data has the special mount/target ID of 900
FILM FRAME PLOTS

Figure 5a
ANATOMICAL (MOUTH) MOUNT TARGET LABELS
PHOTO DATA REDUCTION RUNSTREAM ELEMENT

@RUN
@PASSWD
@ASG,A DATASYSTEMS*PHOTOSTREAM.
@COPY DATASYSTEMS*PHOTOSTREAM.
@FREE DATASYSTEMS*PHOTOSTREAM.
@ED,U .XTCVT
C /REELN/XXXXX/A
FIND@XQT
INPUT
( 0 0 0 0)
@EOF
@ADD,EP .XTCVT
@ED,U .XTPTLT
FIND@XQT
INPUT
  3 0 0 0 0 0 LX1111 LX9999
@EOF
@ADD,EP .XTPTLT
@ED,U .XTSGN
C /XXXXXXXX/11119999/A
FIND@XQT
INPUT
  3 LX1111 LX9999
@EOF
@ADD,EP .XTSGN
@FIN

ENTER EDIT PROCESSOR
WHERE XXXX=CURRENT PDS REEL NUMBER
LOCATE PLACE TO INSERT CORRECTIONS
INSERT CORRECTIONS (IF ANY) AS PER ELT .XTCVT (APPENDIX 1a)
(INsert zeroes if no corrections required)
EXIT INPUT MODE AND EDIT PROCESSOR
EXECUTE CONVERSION PROGRAM
ENTER EDIT PROCESSOR
LOCATE PLACE TO INSERT PLOT OPTIONS
INSERT PLOT OPTIONS AS PER ELT .XTPTLT (APPENDIX 2a)
REFER TO .XTPTLT FOR OPTION DEFINITIONS
EXIT INPUT MODE AND EDIT PROCESSOR
EXECUTE FRAME PLOT PROGRAM
ENTER EDIT PROCESSOR
WHERE 11119999=FILE NAME OF 180-DAY UNIVAC OUTPUT DATA TAPE
LOCATE PLACE TO INSERT INPUT OPTIONS
INSERT INPUT OPTIONS AS PER ELT .XTSGN (APPENDIX 3a)
REFER TO .XTSGN FOR OPTION DEFINITIONS
EXIT INPUT MODE AND EDIT PROCESSOR
EXECUTE REFORMATTING PROGRAM

FIGURE 8

38
REFERENCES


1 @ASG.TJ 1...U0H.REELN  . INPUT DATA FROM DIGITIZER
2 @ASG 9...F///300
3 @ASG 10...F///300
4 @ASG 11...F///300
5 @ASG 12...F///300
6 @ASG 13...F///300
7 12345678901234567890123456789012345678901234567890
8 @ASG 14...F///300
9 @ASG 15...F///300
10 @ASG 16...F///300
11 @ASG 17...F///300
12 @ASG 18...F///300
13 @ASG 19...F///300
14 @ASG 20...F///300
15 @ASG 21...F///300
16 @ASG 22...F///300
17 @ASG 23...F///300
18 @ASG 24...F///300
19 @ASG 25...F///300
20 @ASG 26...F///300
21 @ASG 27...F///300
22 @ASG 28...F///300
23 @ASG 29...F///300
24 @ASG 30...F///300
25 @ASG 31...F///300
26 @ASG 32...F///300
27 @ASG 33...F///300
28 @ASG 34...F///300
29 @ASG 35...F///300
30 @ASG 36...F///300
31 @ASG 37...F///300
32 @ASG 38...F///300
33 @ASG 39...F///300
40 @ASG 40...F///300
41 @ASG 41...F///300
42 @ASG 42...F///300
43 @ASG 43...F///300
44 @ASG 44...F///300
45 @ASG 45...F///300
46 @ASG 46...F///300
47 @ASG 47...F///300
48 @ASG 48...F///300
49 @ASG 49...F///300
50 @ASG 50...F///300

40 Appendix 1a
LAMBERT*TPFS(O).MAIN(37)

I
THIS PROGRAM WILL READ DIGITIZED PHOTODATA IN POS OUTPUT
 FORMAT AND CONVERT TO UNIVAC 1182 FIELDATA FORMAT. A
SEPARATE FILE IS GENERATED FOR EACH CAMERA SITE ID.

NRD = NUMBER OF RUNS REQUIRING A CHANGE IN THE RUN NUMBER.
LR = OLD RUN NUMBER
NR = NEW RUN NUMBER
IC = CAMERA NUMBER FOR WHICH A RUN NUMBER IS TO BE CORRECTED
IP = OCCURRENCE OF RUN NUMBER TO BE CHANGED, 1ST, 2ND, ETC.

IT IS ASSUMED THAT WHEN A RUN IS TO BE BYPASSED, IT HAS
BEEN PREVIOUSLY DIGITIZED AND PLACED IN THE FILE, AND
LATER RE-DIGITIZED AND PLACED IN THE FILE AFTER A GROSS
ERROR WAS DISCOVERED. THE FIRST OCCURRENCE OF THE RUN IS
ALWAYS BYPASSED.

NRB = NUMBER OF RUNS TO BYPASS ON THIS TAPE
IRN = RUN NUMBER TO SKIP
ICN = CAMERA NUMBER FOR WHICH TO SKIP RUN

LRN = RUN NUMBER
LCN = OLD CAMERA NUMBER
NCN = NEW CAMERA NUMBER
NOC = OCCURRENCE OF CAMERA NUMBER TO BE CHANGED, 1ST, 2ND, ETC

DIMENSION LCN(6), NCN(6), NOC(6), ITRY(6), ICCD(6)
DIMENSION IC(6), IP(6), ICN(6), IPAS(6)
DIMENSION ITABLE(5),JTABLE(4000),ICAM(3)
COMMON/PHTHDR/RUNID,VARID,ISITID,FITIME,JDTDIG,DELTA,IPGVER
COMMON/IOUT(3).DATA(250).IDL,IUNIT,ITARG
CHARACTER*6 RUNID,VARID,NR(6),IRN(6),FD2ASC,
  ILRN(6),JDTDIG,IPGVER,RUN(100.3)

DATA IOUT/9.10.11/
FORMAT(IX.14)
IERR=0
IUNIT=1

READ CORRECTION INFORMATION
READ (5,230) NRD,NRB,NRC
PRINT 230. NRD,NRB,NRC

IF (NRD.EQ.0) GO TO 20
DO 10 J=1,NRD
  READ (5,240) LR(J),NR(J),IC(J),IP(J)
  PRINT 240, LR(J),NR(J),IC(J),IP(J)
  CONTINUE

IF (NRB.EQ.0) GO TO 40
DO 30 J=1,NRB
  READ (5,250) IRN(J),ICN(J)
  PRINT 250, IRN(J),ICN(J)
IF (NRC.EQ.0) GO TO 60
DO 50 J=1,NRC
   READ (5,250) LRN(J),LCN(J),NCN(J),NOC(J)
   PRINT 250, LRN(J),LCN(J),NCN(J),NOC(J)
50 CONTINUE
CONTINUE
READ RECORD FROM PDS TAPE
CALL CONVT(IOTYPE,IOT)
IF (IST.NE.4) GO TO 210
IF (IST.NE.1) GO TO 70
IF (NRC.EQ.0) GO TO 100
DO 90 J=1,NRC
   IF (RUNID.EQ.LRN(J).AND.ISITID.EQ.IC(J)) ITRY(J)=ITRY(J)+1
90 CONTINUE
CONTINUE
DO 90 J=1,NRC
   IF (RUNID.EQ.LRN(J).AND.ISITID.EQ.IC(J)) ICDD(J)=ICDD(J)+1
   IF (RUNID.EQ.LRN(J).AND.ISITID.EQ.IC(J).AND.ITRY(J).EQ.IP(J)) 1
   IF (RUNID.EQ.LRN(J)) RUID=NR(J)
90 CONTINUE
100 IF (NRB.EQ.0) GO TO 120
DO 110 J=1,NRB
   IF (RUNID.EQ.IRN(J).AND.ISITID.EQ.ICN(J).AND.IPAS(J).EQ.0) 1
110 CONTINUE
CONTINUE
IF (NRB.EQ.0) GO TO 140
DO 130 J=1,NRC
   IF (RUNID.EQ.LRN(J).AND.ISITID.EQ.LCN(J)) 1
   IF (RUNID.EQ.LRN(J).AND.ISITID.EQ.LCN(J)) ISITID=NCN(J)
130 CONTINUE
CONTINUE
ISW=ISITID
IF (ISW.LE.0.OR.ISW.GT.3) GO TO 150
CONTINUE
PRINT 270, RUNID,ISITID,ISW
STOP 'CAM NO. ERROR'
CONTINUE
CONVERT DATA FOR CAMERA 1
CALL CNVT1(ITYPE, IST)
121 IF (NC.LT.2) NC=1
122 GO TO 70
123 C 170 CONTINUE
124 C CONVERT DATA FOR CAMERA 2
125 C CALL CNVT1(ITYPE, IST)
126 IF (NC.LT.3) NC=2
127 GO TO 70
128 C 180 CONTINUE
129 C CONVERT DATA FOR CAMERA 3
130 C CALL CNVT1(ITYPE, IST)
131 NC=3
132 GO TO 70
133 C 190 CONTINUE
134 IPAS(J)=1
135 C BYPASS CURRENT RUN
136 C 200 CALL CNVT1(ITYPE, IST)
137 IF (IST.EQ.4) GO TO 200
138 GO TO 70
139 C 210 CONTINUE
140 C WRITE TRIPLE END-OF-FILE AT END OF DATA
141 C DO 220, I=1, NC
142 DO 225, J=1, 3
143 CALL EOFPUT(IOUT(I).IER)
144 IF (IER.NE.0) THEN
145 PRINT 13, IER
146 STOP 'ABORT IN MAIN'
147 END IF
148 225 CONTINUE
149 C SET CONDITION WORD TO NUMBER OF CAMERAS
150 C CALL FCONO(ICOND)
151 BITS(ICOND.25.12)=BITS(NC.25.12)
152 CALL FSETC(ICOND)
153 C PRINT DIRECT ACCESS TABLES
154 PRINT 251
155 MAX=4000
156 DO 300, I=1, NC
157 CALL GETDIR(IOUT(I).MAX, 0, ITABLE, IER)
158 IF (IER.NE.0) STOP 'DIRECTORY ERR'
159 ICAM(I)=ITABLE(3)
160 NRUNS=ITABLE(2)
161 ISECT=ITABLE(1)
162 PRINT 211, ICAM(I)
163 C 220 CONTINUE
164 C EG7613 10/16/76
CALL GETDIR(IOUT(I),MAX,ISECT,UTABLE,IER)
IF(IER.NE.0)STOP 'DIRECTORY ERR'
IPRINT=2
DO 296,J=1,NRUNS
RUN(NRUNS,ICAM(I))=FD2ASC(UTABLE(IPRINT),6)
PRINT 221,RUN(NRUNS,ICAM(I)),(JTABLE(K).K-IPRINT+1,.IPRINT+1+)
UTABLE(IPRINT+2)+1)
IPRINT=IPRINT+3+UTABLE(IPRINT+2)
296 CONTINUE
PRINT 231,(ITABLE(K),K=1,5)
300 CONTINUE
C STOP
C 13 FORMAT(4X,'STATUS ERROR = ',03)
195 230 FORMAT (315)
196 240 FORMAT (2(4X,A6),215)
197 250 FORMAT (4X,A6,315)
198 260 FORMAT (' READ ERROR, STATUS = ',13)
199 270 FORMAT (' CAMERA NUMBER ERROR, RUN = ',A6,2110)
200 251 FORMAT('1')
201 211 FORMAT(' DIRECT ACCESS TABLE FOR CAMERA ',12)
202 231 FORMAT(1X,5(16,4X)//)
203 221 FORMAT(1X,A6,2X,1318)
204 C END

PRT.S CONVT
SUBROUTINE CNVT1(ITYPE,IST)

C FUNCTION:
C CONTROL (1) READ OF DIGITIZED PHOTO DATA
3 FROM POS TAPE
4 (2) CONVERSION TO UNIVAC COMPATIBLE FORMAT AND (3) SCALING
5 OF DATA
6
7 ARGUMENT DEFINITIONS:
8 I TYPE-RETURNED PDS RECORD TYPE WHERE
9 i=HEADER RECORD
10 2=TIME RECORD
11 3=X DATA RECORD
12 4=Y DATA RECORD
13
14 IST-RETURNED ERROR STATUS WORD FROM ADTIO GENERAL
15 PURPOSE I/O ROUTINE
16
17 COMMON/PHTHDR/RUNID,VARI,ISITID,FITIME,JDDTIG,DELTA,IPGVER
18 COMMON/I36/IOUT(3),DAT(250),IDL,UNIT,ITARG
19 CHARACTER*6 VARIC1(28), VAR2C1(28), VAR3C1(28), VARCC1(28)
20 CHARACTER*6 VARIC2(28), VAR2C2(28), VAR3C2(28), VARCC2(28)
21 CHARACTER*6 VARIC3(28)
22 COMMON/RUNID,VARI,JDDTIG,IPGVER,B(28),TARNUM+3,CAMNO+1
23 1,OLDRUN
24
25 DEFINE MOUNT/TARGET ID'S
26
27 DATA VARIC1 /'1X100', '1Y100', '1X102', '1Y102',
28 '1X103', '1Y103', '1X108', '1Y108', '1X105', '1Y105' /
29 '1X104', '1Y104', '1X101', '1Y101', '1X201', '1Y201',
30 '1X106', '1Y106', '1X202', '1Y202', '1X107', '1Y107' /
31 '1X108', '1Y108', '1X109', '1Y109', '1X103', '1Y103' /
32 '1X110', '1Y110', '1X111', '1Y111', '1X112', '1Y112' /
33 '1X113', '1Y113', '1X114', '1Y114', '1X115', '1Y115' /
34 '1X116', '1Y116', '1X117', '1Y117', '1X118', '1Y118' /
35 '1X119', '1Y119', '1X120', '1Y120', '1X121', '1Y121' /
36 '1X122', '1Y122', '1X123', '1Y123', '1X124', '1Y124' /
37 '1X125', '1Y125', '1X126', '1Y126', '1X127', '1Y127' /
38 '1X128', '1Y128', '1X129', '1Y129', '1X130', '1Y130' /
39 '1X131', '1Y131', '1X132', '1Y132', '1X133', '1Y133' /
40 '1X134', '1Y134', '1X135', '1Y135', '1X136', '1Y136' /
41 '1X137', '1Y137', '1X138', '1Y138', '1X139', '1Y139' /
42 '1X140', '1Y140', '1X141', '1Y141', '1X142', '1Y142' /
43 '1X143', '1Y143', '1X144', '1Y144', '1X145', '1Y145' /
44 '1X146', '1Y146', '1X147', '1Y147', '1X148', '1Y148' /
45 '1X149', '1Y149', '1X150', '1Y150', '1X151', '1Y151' /
46 '1X152', '1Y152', '1X153', '1Y153', '1X154', '1Y154' /
47 '1X155', '1Y155', '1X156', '1Y156', '1X157', '1Y157' /
48 '1X158', '1Y158', '1X159', '1Y159', '1X160', '1Y160' /
49 '1X161', '1Y161', '1X162', '1Y162', '1X163', '1Y163' /
50 '1X164', '1Y164', '1X165', '1Y165', '1X166', '1Y166' /
51 '1X167', '1Y167', '1X168', '1Y168', '1X169', '1Y169' /
52 '1X170', '1Y170', '1X171', '1Y171', '1X172', '1Y172' /
53 '1X173', '1Y173', '1X174', '1Y174', '1X175', '1Y175' /
54 '1X176', '1Y176', '1X177', '1Y177', '1X178', '1Y178' /
55 OLDRUN='
56
57 CONVERTS AND SCALES PHOTO DATA
DO 90 KP=1,1000
   DO 10 J=1,250
   10 DAT(J)=0.0
   IF (ITYPE.NE.1) GO TO 20

C CHECK FOR MORE THAN 250 FRAMES OF DATA.

C IF (RUNID.EQ.OLDRUN) GO TO 30
   OLDRUN=RUNID
   VARID='PHDATA'

C WRITE HEADER
   CALL PUTRCD(IOUT(ISITID).RUNID,VARID.ISITIO.IDL.DAT.IER)
   IF (IER.NE.0) GO TO 112

C READ NEXT PDS RECORD
   CALL CONVT(ITYPE.IST)
   IF (IST.NE.4) GO TO 100
   IF (ITYPE.EQ.1) GO TO 90
   GO TO 40

C HANDLES EOF'S FOR MORE THAN 250 FRAMES

C DETERMINE VARIABLE NAMES FOR A GIVEN CAMERA.

C CHANGE DATA TO METERS & SET ZERO VALUES IN NON-TIME RECORDS EQUAL
   ENCODE ('1.160,CAMNO) ISITID
   SUBSTR(VARID,1,1)=CAMNO
   IF (ITYPE.EQ.2) SUBSTR(VARID,2,2)=’TS’
   IF (ITYPE.EQ.2) GO TO 70
   IF (BITS(ITARG,5,16).NE.0) GO TO 60
   IF (RUNID.LT.'LX0253') NDT=1
   IF (RUNID.LT.'LX1410') NDT=2
   IF (RUNID.GE.'LX1410') NDT=3
   IF (((RUNID.EQ.'LX1234').OR.(RUNID.EQ.'LX1235').OR.(RUNID.EQ.
       'LX1236')).OR.(RUNID.EQ.'LX1237'))) NDT=4

DO 50 J=1,28
   50 IF (((ISITID.EQ.1).AND.(NDT.EQ.1)) B(J)=VAR1C1(J)
   IF (((ISITID.EQ.1).AND.(NDT.EQ.2)) B(J)=VAR2C1(J)
   IF (((ISITID.EQ.1).AND.(NDT.EQ.3)) B(J)=VAR3C1(J)
   IF (((ISITID.EQ.1).AND.(NDT.EQ.4)) B(J)=VARCC1(J)
   IF (((ISITID.EQ.2).AND.(NDT.EQ.1)) B(J)=VAR1C2(J)
   IF (((ISITID.EQ.2).AND.(NDT.EQ.2)) B(J)=VAR2C2(J)
   IF (((ISITID.EQ.2).AND.(NDT.EQ.3)) B(J)=VAR3C2(J)
   IF (((ISITID.EQ.2).AND.(NDT.EQ.4)) B(J)=VARCC2(J)

IF ((ISITID.EQ.3).AND.(NDT.EQ.3)) B(J)=VAR3C3(J)
CONTINUE
ITT=2*ITARG
IF (ITYPE.EQ.3) VARID=B(ITT-1)
IF (ITYPE.EQ.4) VARID=B(ITT)
GO TO 70
CONTINUE
IF (ITYPE.EQ.0) SUBSTR(VARID,2,1)=’X’
IF (ITYPE.EQ.0) SUBSTR(VARID,2,1)=’Y’
NTC=BITS(ITARG,5,16)
ENCODE (3,170,TARNUM) NTC
SUBSTR(VARID,3,3)=TARNUM
CONTINUE
DO 80 J=1,250
IF (ITYPE.EQ.2) DAT(J)=DAT(J)/10000.0
IF (ITYPE.GT.2) DAT(J)=(DAT(J)/100000.0)*.0254
IF (ITYPE.GT.2).AND.(DAT(J).EQ.0.0)) DAT(J)=999.0
CONTINUE
IF ((RUNID.GE.’LX0253’).AND.(VARID.EQ.’ ‘)) GO TO 90
CALL SPROCKET HOLE SUBROUTINE IF 900 OR 100 RECORD
IF (SUBSTR(VARID,2.5).EQ.’X900 ‘) CALL SPH9X (DAT)
IF (SUBSTR(VARID,2.5).EQ.’Y900 ‘) CALL SPH9Y (DAT)
IF (SUBSTR(VARID,2.5).EQ.’X100 ‘) CALL SPH1X (DAT)
IF (SUBSTR(VARID,2.5).EQ.’Y100 ‘) CALL SPH1Y (DAT)
WRITE TIME OR DISPLACEMENT RECORD
CALL PUTRCD(IOUT(ISITID).RUNID.VARID.ISITID.IDL.DAT.IER)
IF(IER.NE.0)GO TO 112
PRINT 200.RUNID.VARID.ISITID.ITYPE.(DAT(J)).J=1,10)
CONTINUE
WRITE DOUBLE END-OF-FILE AT END OF RUN (END OF DATA FOR 1 FILM)
CALL EOFPUT(IOUT(ISITID),IER)
IF(IER.NE.0)GO TO 112
CALL EOFPUT(IOUT(ISITID),IER)
CALL EPSAVE (SPH9X,SPH9Y)
PRINT 180.SP9X,SPH9Y
RETURN
OUTPUT ABORT ROUTINE
PRINT 12.IER
12 FORMAT(1X,’STATUS ERROR = ’,03)
STOP “ABORT”
FORMAT (’/’ SPROCKET HOLE AVERAGES, X = ’,F10.6,’ Y = ’,F10.6)
180  190  FORMAT ('1',IX.2(A6.4X),15.4X,F10.3,4X,A6.4X,F6.0,4X,A6.4X,16)
181  200  FORMAT (IX.2(A6.4X),16,4X,16,4X/I10(F11.8,2X))
182       C       END

@PRT.S  .SPHAVG
SUBROUTINE CONVT(ITYPE,IST)

C FUNCTION:
C (1) READ PDS PHOTO DATA FROM TAPE AND (2) CONVERT
DIGITIZED PHOTO DATA TO UNIVAC COMPATIBLE FORMAT

ARGUMENT DEFINITIONS:
ITYPE-RETURNED PDS RECORD TYPE WHERE
i=HEADER RECORD
2=TIME RECORD
3=X DATA RECORD
4=Y DATA RECORD
IST-RETURNED ERROR STATUS WORD FROM ADTIO GENERAL
PURPOSE I/O ROUTINE

DIMENSION I32(300)
COMMON/PHTHDR/RUNID,VARID,ISITID,FITIME,JDTDIG,DELTA,IPGVER
COMMON/I36/IDUT(3),DAT(250),IDL,UNIT,ITARG
CHARACTER*6 RUNID,VARID,JDTDIG,IPGVER

C DATA FOR EXTENDING THE SIGN BIT.
C DATA NZ /0740000000000/ .SIGN /0020000000000/
NW=255
CALL ATRD(IUNIT,I32,NW,IST)
IF (IST.EQ.1) RETURN
C ZERO OUT RECEIVING ARRAY.
DO 10 I=1,250
10 DAT(I)=0.
C CONVERT RUN NUMBER FROM NOVA ASCII TO UNIVAC ASCII
CALL ASCBT9(I32,1,17,6,RUNID)
C ITYPE=BITS(I32(1),1,16)
IF (ITYPE.GT.1) GO TO 60
C EXTRACT HEADER RECORD AND CONVERT TO UNIVAC WORDS
BITS(ITIME,5,8)=BITS(I32(2),29,8)
BITS(ITIME,13,24)=BITS(I32(3),1,24)
FITIME=FLOAT(ITIME)
BITS(ISITID,21,16)=BITS(I32(4),5,16)
BITS(IDATE,21,12)=BITS(I32(3),25,12)
BITS(IDATE,33,4)=BITS(I32(4),1,4)
ENCODE(6,151,JDTDIG,IDATE)
BITS(IDATE,33,4)=BITS(I32(5),1,4)
DELTA=FLOAT(IDELTA)
BITS(IVER,25,12)=BITS(I32(5),5,12)
ENCODE(6,151,IPGVER)IVER
151 FORMAT(16)
IF(IDATE.EQ.0)JDTDIG=''
IF(IVER.EQ.0)IPGVER=''
C RETURN
60 IF (ITYPE.GT.2) GO TO 70
61 GO TO 80
62 C EXTRACT TARGET NUMBERS FROM INPUT BUFFER
63 C 70
64 BITS(1TARG.5,8)=BITS(132(2),29,8)
65 BITS(1TARG.13,24)=BITS(132(3),1,24)
66 BITS(ISITID.5,12)=BITS(132(3),25,12)
67 BITS(ISITID.17,20)=BITS(132(4),1,20)
68 C EXTRACT DATA VALUES FOR DATA RECORD (OR TIME RECORD)
69 C 70
70 IW=4
71 IB=21
72 GO TO 90
73 IW=5
74 IB=17
75 90 DO 110 I=1,250
76 ITEMP=0
77 IS=5
78 IL=37-IB
79 IF (IL.GT.32) IL=32
80 BITS(ITEMP.IS.IL)=BITS(132(IW),IB,IL)
81 IF (IL.EQ.32) GO TO 100
82 IW=IW+1
83 IB=1
84 IS=4+IL+1
85 IL=37-IS
86 BITS(ITEMP.IS.IL)=BITS(132(IW),IB,IL)
87 100 IB=IB+IL
88 IF (IB.GT.36) IW=IW+1
89 C EXTEND THE SIGN BIT FOR NEGATIVE NUMBERS AND CHANGE
90 C 2'S COMPLEMENT TO 1'S COMPLEMENT
91 C 92
93 IF (AND(SIGN.ITEMP).NE.0) ITEMP=OR(ITEMP,NZ)-1
94 IF (IB.GT.36) IB=1
95 DAY(I)=FLOAT(ITEMP)
96 110 CONTINUE
97 IDL=250
98 C 100
99 C 101
100 RETURN
101 C END
102 C
C THIS SUBROUTINE CALCULATES SPROCKET HOLE AVERAGES FOR EACH CAMERA
FROM 5 CONTINUOUS FRAMES IN THE PHOTO DATA USING THE CHAIR OR SLED
DISPLACEMENTS AND THE SPROCKET HOLE DISPLACEMENT.

ARGUMENT DEFINITIONS:
SPHX-RETURNED SPROCKET HOLE AVG. FOR X VALUES
SPHY-RETURNED SPROCKET HOLE AVG. FOR Y VALUES

ENTRY SPH9X (X)
DO 10 J=1,250
X9(J)=X(J)
ICX=ICX+1
RETURN

ENTRY SPH9Y (X)
DO 20 J=1,250
Y9(J)=X(J)
ICY=ICY+1
RETURN

ENTRY SPHtX (X)
DO 30 J=1,250
Xt(J)=X(J)
ICX=ICX+1
RETURN

ENTRY SPHIY (X)
DO 40 J=1,250
Yt(J)=X(J)
ICY=ICY+1
RETURN

CONTINUE

CALCULATE DIFFERENCES IN SPROCKET HOLE AND SLED OR CHAIR
TARGET DISPLACEMENTS FOR 5 CONSECUTIVE FRAMES,
AND SUM THESE DIFFERENCES (X VALUES)

IBG=0
IF (ICX.LT.2) GO TO 80

SUMX=0
DO 70 J=1,250
IF (X9(J).GT.900..OR.X9(J).GT.900.) GO TO 60
SUMX=SUMX+(ABS(X9(J))-.ABS(X9(J)))
IF (IBG.EQ.0) IBG=J

51 Appendix 1e
IF ((J-IBG).GE.4) GO TO 90
GO TO 70
SUMX=0
IBG=0
70 CONTINUE
C
IF (IBG.GE.250) IBG=0
80 SPHX=0
GO TO 100
90 CONTINUE
C
CALCULATE SPROCKET HOLE AVERAGES AND CONVERT TO PDS UNITS
FROM CENTIMETERS (X VALUE)
SPHX=(SUMX/5.)*(1000./(1.884765*.0254))
C
IF (ICY.LT.2) GO TO 120
C
CALCULATE DIFFERENCES IN SPROCKET HOLE AND SLED OR CHAIR DISPLACEMENTS FOR 5 CONSECUTIVE FRAMES AND SUM THESE VALUES. USE THE SAME 5 FRAMES USED IN CALCULATING THE X VALUES ABOVE
C
IST=4
IF (IBG.NE.0) IST=IBG
IND=250
IF (IBG.NE.0) IND=IST+4
90 C
SUMY=0
DO 110 J=IST,IND
   IF (Y1(J).GT.900..OR.Y9(J).GT.900.) GO TO 120
   SUMY=SUMY+(ABS(Y1(J))-ABS(Y9(J)))
110 CONTINUE
GO TO 130
120 SPHY=0
GO TO 140
130 CONTINUE
C
CALCULATE SPROCKET HOLE AVERAGE AND CONVERT TO PDS UNITS FROM CENTIMETERS (Y VALUE)
SPHY=(SUMY/5.)*(1000./(1.884765*.0254))
C
ICX=0
ICY=0
C
RETURN
END

*PRT.S.ASCB9
SUBROUTINE ASCBT9(INPUTA,IWD,IBIT,NCHAR,CSTRNG)

FUNCTION:
CONVERT STRING OF 8-BIT ASCII CHARACTERS TO UNIVAC COMPATIBLE 9-BIT ASCII CHARACTER STRING

ARGUMENT DEFINITIONS:
- INPUTA-INTEGER INPUT ARRAY OF 8-BIT CHARACTER DATA
- IWD-STARTING WORD IN INPUTA OF 8-BIT ASCII CHARACTER STRING
- IBIT-BEGINNING BIT IN IWD OF INPUT 8-BIT ASCII CHARACTER STRING
- NCHAR-NUMBER ASCII CHARACTERS TO BE CONVERTED TO UNIVAC FORMAT
- CSTRNG-RETURNED OUTPUT STRING OF UNIVAC 9-BIT ASCII CHARACTERS

PROGRAMMER: J LAMBERT 20 MAY 1983

INTEGER INPUTA()

CHARACTER*(*) CSTRNG,ITEMP*
DATA IL1,IL2/2*0/
IB=IBIT
IW=IWD

1ST BIT OF UNIVAC ASCII CHARACTER ALWAYS 0(ZERO)

DO 20,I=1,NCHAR
IF((IB+7).LT.36)THEN
BITS(ITEMP.2,8)=BITS(INPUTA(IW),IB,8)
ELSE
IL1=37-IB
IL2=8-IL1
BITS(ITEMP.2,IL1)=BITS(INPUTA(IW),IB,IL1)
BITS(ITEMP.2+IL1,IL2)=BITS(INPUTA(IW+I),IL2)
END IF
SUBSTR(CSTRNG,I)=SUBSTR(ITEMP.,I)
IB=IB+8
IF(IB.GT.36)THEN
IW=IW+1
IB=IB-36
END IF
20 CONTINUE
RETURN
END
LAMBERT*TPFS O.XOTPLT(11)
1  *XOT MP0051/PLT
2  *
3  CONTROL STATEMENT IMMEDIATELY AFTER *XOT STATEMENT
4  *
5  POS 5 - NO. INPUT UNITS (1, 2 OR 3)
6  *
7  POS 10 - TIME RECORD ANALYSIS PLOT OPTION
8  *
9  POS 15 - XY CONTOUR PLOT OPTION
10  *
11  POS 20 - FRAME PLOT OPTION
12  *
13  POS 25 - FORMAT OPTION FOR FRAME PLOTS
14  *
15  POS 30 - LOWEST RUN NO. TO BE PROCESSED
16  *
17  POS 40 - HIGHEST RUN NO. TO BE PROCESSED
18  *
19  *XOT SCC$*INTLIB.FRSOPOP
20  *
21  Datasystems
22  5-4890
23  Frame Plots
24  Frame Plots
25  1 Hardcopy 1 Microfiche
26  *
27  *
28  Draw 1-Ends
29  @Delete,C Compress.
30  @Free Compress.
31  @Asg,Cp Compress.,F///1000
32  @Copy 17.,Compress.
33  @Free Compress.
34  *
35  .Exited from .Xotplt

@Prt. S .PLOT
FUNCTION:

THIS ASCII FORTRAN DRIVER PROGRAM CONTROLS THE GENERATION
OF MOUNT PLOTS FOR A RANGE OF RUNS SPECIFIED BY THE USER.
TWENTY-FIVE FRAMES (DATA PTS.) ARE PLOTTED ON EACH OF 10
PAGES FOR A GIVEN MOUNT AND SITE ID. THE DISSPLA PLOT
PACKAGE IS USED FOR ALL PLOTTING.

COMMON/LOAD/XMIN,XMAX,IPGE,YMIN,YMAX,TITLE,IST,ILT,IX,XOR,
+YOR,VAXX,VARY,IERR,STEP,DC(250)
COMMON/PHTHDR/RUNID,VARID,ISITID,FITIME,JSTDIG,DELTA,IPGVER
CHARACTER*6 RUNID,VARID,JSTDIG,IPGVER,TITLE(9),VARX(12),VARY(12)
CHARACTER*6 FRUN,LRUN,RUNS(200),FIRST,LAST,ENC=4,SITE=1
DATA MAX/280/
READ USER OPTIONS IN CARD IMAGE
READ(5,6) IFILE,ITIM,IXY,IMTPLT,IWAY,FRUN,LRUN
WHERE:
IFILE: NO. OF INPUT FILES
ITIM: TIME RECORD ANALYSIS PLOT OPTION
IF = OUTPUT TO LINE PRINTER
I = OUTPUT TO COMPRESSED PLOT FILE
NOT 0 AND NOT 1 = NO TIME RECORD ANALYSIS
IXY: XY CONTOUR PLOT OPTION
O = PLOT XY CONTOUR FOR EACH MOUNT/SITE/RUN
NOT 0 = NO XY CONTOURS
IMTPLT: MOUNT PLOT OPTION
O = FRAME PLOTS REQUIRED FOR ALL DATA
NOT 0 = NO FRAME PLOTS
IWAY: PLOT DIRECTION OPTION
O = BOTTOM TO TOP COLUMNS 2 AND 4
NOT 0 = TOP TO BOTTOM ALL COLUMNS
FRUN: LOW ORDER VALUE OF RUN NO. RANGE
LRUN: HIGH ORDER VALUE OF RUN NO. RANGE
SET COMPRESSOR FOR PLOTTING
CALL COMPRS
CALL SETDEV(6,0)
CALL GRACE(0.)
CALL PAGE(14.1/02,1/02)
CALL NOBRDR
CALL ERASE
ONE UNIT = 10.5/1.02 INCHES
CALL UNITS(10.5/1.02)
COMPUTE RUN RANGE
FIRST=SUBSTR(FRUN,3,4)
LAST=SUBSTR(LRUN,3,4)
DECODE(4,FIRST)IFIRST
DECODE(4,LAST)ILAST
FORMAT(I4)
COMPUTE INPUT UNIT ASSIGNMENTS
ILIM=FILE+8
IU=9
CONTINUE
PRINT 3.IU,ILIM,ITIM,IXY,INTPLT,IWAY,FRUN,LRUN
FORMAT(6I10.2(4X,A6))

GENERATE TITLE FOR EACH PAGE
TITLE(2)='CAM'
TITLE(5)='PAG'
TITLE(6)='E'
TITLE(9)=''
TITLE(3)=''
TITLE(4)=''
TITLE(8)=''

PLOT DATA FOR EACH RUN
IRN=ABS(ILAST-IFIRST)+1
INC=IFIRST
DO 500.I=1,IRN

CREATE RUN ARRAY TO BE SEARCHED
SUBSTR(RUNS(I),1,2)='LX'
ENCODE(17,ENC)INC
SUBSTR(RUNS(I),3,4)=ENC
INC=INC+1

ATTEMPT TO SET SECTOR TO HEADER FOR RUN
CALL SETSCT(IU.RUNS(I),'PHDATA',1,ISIT,NREC,IER)
IF(IER.NE.0)GO TO 500
CALL GETRCO(IU.MAX.RUNXVARID.XSITE.IDL,DCIER)
IF(IER.NE.0)THEN
PRINT 13,IER
FORMAT(I3,'STATUS=')
STOP 'HEADER ERROR'
END IF

ENCODE(10,SITE)ISITID
SUBSTR(TITLE(3),3,1)=SITE
TITLE(1)=RUNID
IF(ITIM.EQ.0 .OR. ITIM.EQ.1)THEN
CALL TIMER(IU,ITIM)
IF(IERR.NE.0)GO TO 500
END IF

PLOT DATA FOR EACH MOUNT FOR THIS RUN
DO 400.L=1,4
LOAD DATA
CALL LDATA(IU,L)
IF(IX.NE.0)THEN
  IST=1
  IFM=1
ELSE
  GO TO 400
END IF

GENERATE TITLE INFORMATION FOR THIS MOUNT

IF(L.EQ.1)THEN
  TITLE(5)= 'NECK'
ELSE IF(L.EQ.2)THEN
  TITLE(5)= 'MOUTH'
ELSE IF(L.EQ.3)THEN
  TITLE(5)= 'HEAD'
ELSE IF(L.EQ.4)THEN
  TITLE(5)= 'PELVIS'
END IF

LOOP FOR EACH PAGE OF PLOTS ( FRAMES)

IF(IMTPLT.NE.0)GO TO 300

DO 55,IPGE=1,10

DEFINE PHYSICAL ORIGIN FOR FIRST FRAME ON PAGE

XOR=.16
YOR=.81

DETERMINE MAX AND MIN VALUES FOR THIS PAGE OF PLOTS

ILT=IST+24
CALL MIXY

PRINT TITLE AT TOP OF PAGE

CALL PLOT1

LOOP TO CONTROL COLUMN OF PLOTS

DO 50,K=1,5
IC=MOD(K,2)
PLOT DATA 5 FRAMES FOR EACH OF 5 COLUMNS

DO 25,J=1,5
CALL PLOT2(IFM)

CHECK FOR DIRECTION OF PLOTS ( UP OR DOWN)

IF(IWAY.NE.0)THEN
  IF(J.NE.5)YOR=YOR-.2
END IF
PLOT BOTTOM TO TOP COLUMNS 2 AND 4

IF(J.NE.5)THEN
  IF(IC.NE.0)THEN
    YOR=YOR-.2
  ELSE
    YOR=YOR+.2
  END IF
END IF

CHECK FOR LAST DATA POINT

IF(IST.LT.250)THEN
  IST=IST+i
  IFM=IFM+1
ELSE
  CALL ENDPL(0)
  GO TO 55
END IF

CONTINUE

MODIFY PHYSICAL ORIGIN FOR NEXT COLUMN

IF(IWAY.NE.0)THEN
  XOR=XOR+.2
  YOR=.81
ELSE
END IF

CONTINUE

IF(IU.LT.ILIM)THEN
  IU=IU+1
  GO TO 1
ELSE
END IF

CALL DONEPL
STOP
END
SUBROUTINE TIMER(IU, ITIM)

FUNCTION:
GENERATE AND PLOT TIME RECORD ANALYSIS FOR A GIVEN RUN

ARGUMENT DEFINITIONS:
IU: LOGICAL UNIT NO. OF INPUT FILE
ITIM: OPTION TO PLOT TIME RECORD ANALYSIS WHERE
0 = OUTPUT TO LINE PRINTER
1 = OUTPUT TO COMPRSSED PLOT FILE

COMMON/LOAD/XMIN, XMAX, IPGE, YMIN, YMAX, TITLE, IST, ILT, IX, XOR,
YOR, VARX, VARY, IERR, STEP.DC(250)
COMMON/PHTHDR/RUNID, VARID, ISITID, F1TIME, JUDTDIG, DELTA, IPGVER
CHARACTER*6 RUNID, VARID, JUDTDIG, IPGVER, TITLE(9), VARX(12), VARY(12)
CHARACTER*6 SYM*1(250), LTIME(250), LETIME*9(250)
CHARACTER*8 LINDEX*3(250), LPDST, LHEAD(2)
DIMENSION ETIME(250), DD(250)
DATA LPDST, LHEAD(2) /'D TIME', 'P TIME', 'FRAME '/
CALL HEIGHT(.O18)

SET SECTOR FOR TIME RECORD
VARID=' TS','
SUBSTR(VARID,1,1)+SUBSTR(TITLE(3),3,1)
CALL SETSCT(IU, RUNID, VARID, 1, ISIT, NREC, IER)
IF(IER.NE.0)THEN
  PRINT 13, IER
  GO TO 100
END IF

READ TIME RECORD
MAX=280
CALL GETRCD(IU, MAX, RUNID, VARID, ISIT, IDL, DD, IER)
IF(IER.NE.0)THEN
  IF(IER.EQ.1)THEN
    GO TO 100
  ELSE
    PRINT 13, IER
    STOP 'TIME ERROR'
  END IF
END IF

CHECK FOR TIME RECORD IDENTIFICATION
IF(SUBSTR(VARID,2,2).NE.'TS')THEN
  PRINT 1, RUNID
  FORMAT(1X,'COULD NOT FIND TIME RECORD FOR RUN ', A6)
  STOP 'TIME ERROR'
ELSE
  IERR=0
END IF

COMPUTE TIME INTERVAL BETWEEN FRAMES
DO 15, K=250, 1, -1
IF (DD(K).LE.O) THEN
  CONTINUE
ELSE
  DO 21, J=1, 250
  IF (DD(J).LE.O) THEN
    CONTINUE
  ELSE
    IF (DD(K).LT.DD(J)) THEN
      TINT=(DD(K)-DD(J))/(K-J)
    GO TO 32
  END IF
  CONTINUE
GO TO 100
END IF
35 CONTINUE
PRINT 16, RUNID
FORMAT (1X, 'TIME RECORD FOR RUN ', A6, ' WAS ALL ZEROES')
GO TO 100
32 ETIME(1)=DD(1)
SYM(1)=''
C COMPUTE EXPECTED TIMES
DO 35, JJ=2, 250
ETIME(JJ)=ETIME(JJ-1)+TINT
SYM(JJ)=''
IF (DD(JJ).LT.0.00001) THEN
  CONTINUE
ELSE
  DIF=ETIME(JJ)-DD(JJ)
  IF (ABS(DIF).GT.0.001) SYM(JJ)=''
END IF
35 CONTINUE
WRITE HEADER INFO FOR TIME RECORD ANALYSIS
ENCE(11, LTINT) TINT
IF (ITIM.EQ.1) THEN
  CALL TABLET('CENTER', 'LONG')
ELSE IF (ITIM.EQ.0) THEN
  CALL TABLET(' ', ' ')
  ELSE IF (ITIM.EQ.1) THEN
  CALL CLINE(LHEAD(1))
  ELSE IF (ITIM.EQ.0) THEN
  PRINT 14, LHEAD(1)
  END IF
11 FORMAT ('1')
END IF
WRITE HEADER INFO. FOR INDIVIDUAL COLUMNS
CALL HEIGHT(.01)
DO 23,IREP=1,5
SUBSTR(LHEAD(2),(((IREP-1)*23)+1),7)=LFRNO
SUBSTR(LHEAD(2),(((IREP-1)*23)+8),8)=LPDST
SUBSTR(LHEAD(2),(((IREP-1)*23)+16),8)=LEXPT
23 CONTINUE
IF(ITIM.EQ.1)THEN
SUBSTR(LHEAD(2),116,1)='$
CALL CTLINE(LHEAD(2))
ELSE IF(ITIM.EQ.0)THEN
PRINT 17,LHEAD(2)
17 FORMAT(1X,A115)
END IF
PRINT TIME RECORD ANALYSIS
DO 60,KK=1,50
DO 71,IREC=1,5
INDEX=(((IREC-1)*50)+KK)
ENCODE(8,LINDEX(INDEX))=INDEX
ENCODE(9,LDD(INDEX))=INDEX
ENCODE(10,LETIME(INDEX))=INDEX
9 FORMAT(F9.5)
8 FORMAT(I3)
SUBSTR(LINE,(((IREC-1)*23)+1),1)=
SUBSTR(LINE,(((IREC-1)*23)+2),3)=INDEX
SUBSTR(LINE,(((IREC-1)*23)+5),9)=LDD(INDEX)
SUBSTR(LINE,(((IREC-1)*23)+14),9)= Letime(INDEX)
SUBSTR(LINE,(((IREC-1)*23)+23),1)=SYM(INDEX)
71 CONTINUE
IF(ITIM.EQ.1)THEN
SUBSTR(LINE,116,1)='$
CALL CTLINE(LINE)
ELSE IF(ITIM.EQ.0)THEN
PRINT 18,LINE
18 FORMAT(1X,A115)
END IF
60 CONTINUE
IF(ITIM.EQ.0)THEN
PRINT 78
78 FORMAT('')
END IF
100 CONTINUE
CALL RESET('HEIGHT')
IF(ITIM.EQ.1)THEN
CALL ENDTAB(0)
CALL ENDP(0)
END IF
RETURN
END
SUBROUTINE LOATA(IU,MNT)

* FUNCTION:
LOAD PHOTO DISPLACEMENT DATA INTO COMMON FOR
ACCESS BY PLOT ROUTINES

* ARGUMENT DEFINITIONS:
IU: LOGICAL UNIT NO. OF INPUT DATA FILE
MNT: MOUNT INDEX

COMMON/LOAD/XMIN,XMAX,YMIN,YMAX,TITLE,IST,ILT,IX,XOR,
* YOR,VARX,VARY,IER,STEP,OG(250),DX(250),DY(250),12)
COMMON/PHTHOR/RUNID,VARID,ISITID,FITIME,UDTDIG,DELTAL,IPGVER
DIMENSION 00(250)
CHARACTER*6 RUNID,VARID,,ISIT,FI,TIME,IPGVER,TITLE(9),VARX(12),VARY(12)
CHARACTER*3 TARGET,OLDRUN*6

DATA MAX/260/
IX=0
IY=0

* INITIALIZE CHAIR DATA FLAG
IF(OLDRUN.NE.RUNID)ICHAIR=0

* SET SECTOR TO TIME RECORD
VARID='T5'
SUBSTR(VARID,1,1)=SUBSTR(TITLE(3),3,4)
CALL SETSCT(IU,RUNID,VARID,1,ISIT,NREC,IER)
IF(IER.NE.0)THEN
PRINT 13,IER
END IF

* GET NEXT DATA RECORD
CALL GETRCD(IU,MAX,RUNID,VARID,ISIT,IDL/DD,IER)
IF(IER.NE.0)THEN
IF(IER.EQ.1)THEN
GO TO 56
ELSE
PRINT 13,IER
GO TO 72
END IF
ELSE
PRINT 13,IER
GO TO 72
END IF

* CHECK FOR CHAIR DATA AND STORE
IF(ICHAIR.EQ.1)THEN
CONTINUE
ELSE
TARGET=SUBSTR(VARID,3,3)
IF(TARGET.EQ.'100')THEN
DO 55 J=1,250
DC(J)=DD(J)
55 CONTINUE
PRINT 606,VARID,RUNID
FORMAT('CHAIR DATA, VARIABLE',A6,' LOADED FOR RUN ',A6)
ICHAR=1
GO TO 8
END IF
END IF
C C PROCESS DATA RECORD
C TGCCHK=SUBSTR(VARID,4,2)
IF(TGCCHK.LT.ONE)THEN
END IF
ELSE
MTCHK=SUBSTR(VARID,3,1)
IF(MTCHK.EQ.MNT)THEN
FORMAT(I1)
CONTINUE
ELSE
END IF
C C CHECK FOR CURRENT MOUNT
IF(IMTCHK.NE.MNT)THEN
CONTINUE
ELSE
IF(SUBSTR(VARID,2,1).EO.'X')THEN
IX=IX+1
DO 22,J=1,250
DX(J,IX)=DD(J)
VARX(IX)=VARID
ELSE IF(SUBSTR(VARID,2,1).EO.'Y')THEN
IY=IY+1
DO 32,J=1,250
DY(J,IY)=DD(J)
VARY(IY)=VARID
END IF
END IF
C C READ ANOTHER DATA RECORD
GO TO 8
CONTINUE
IF(IY.EQ.0)THEN
IX=O
ELSE IF(IX.EQ.0)THEN
CONTINUE
ELSE
PRINT 58,IX,IY,MNT,RUNID
FORMAT(1X,I2,2X,'X DISP AND ',I2,2X,'Y DISP',
      * LOADED FOR MOUNT ',I2,5X,'RUN ',A6)
PRINT 66,(VARX(I),I=1,IX),(VARY(I),I=1,IY)
FORMAT(12(2X,A6))
END IF
C C IF NO CHAIR DATA STORE 999'S IN ARRAY
IF(ICHAIR.EQ.0)THEN
DO 71,J=1,250
DC(J)=999.
END IF
CONTINUE
OLDRUN=RUNID
RETURN
END
SUBROUTINE MIXY

FUNCTION:
Determine minimum and maximum X and Y values for
a page of mount plots

COMMON/LOAD/XMIN.XMAX.IPGE.YMIN.YMAX,TITLE.IST.ILT.IX.XR.
*YR.VARX.VARY.IERR.STEP.DC(250),DX(250,12),DY(250,12)
COMMON/PHTHDR/RUNID.VARID.ISITI.FITIME.TIMES.DELTA.IPGVER
CHARACTER*6 RUNIT.VARID.JDTIM.DELTA.IPGVER,TITLE(9),VARX(12),VARY(12)
XMIN,YMIN=999.
XMAX,YMAX=0.

C INITIALIZE COUNTERS
ISTNEW=IST
ILTNEW=ILT
C CHECK IF ALL PTS TO BE PLOTTED
IF(IERR.EQ.999)THEN
ISTNEW=1
ILTNEW=250
END IF
C EXAMINE DATA PTS. FOR ALL TARGETS
DO 20.JK=ISTNEW,ILTNEW
DO 20.JL=1,IX
OXX=ABS(DX(JK,JL))
DYY=ABS(DY(JK,JL))
IF(OXX.LE.990. AND. DYY.LE.990.)THEN
XMIN=AMIN(DXX,XMIN)
YMIN=AMIN(DYY,YMIN)
XMAX=AMAX(DXX,XMAX)
YMAX=AMAX(DYY,YMAX)
END IF
20 CONTINUE
C EQUATE X AND Y AXES
XDIF=XMAX-XMIN
YDIF=YMAX-YMIN
IF(XDIF.LE.YDIF)THEN
XMAX=XMAX+(YDIF-XDIF)/2.
XMIN=XMIN-(YDIF-XDIF)/2.
ELSE
YMAX=YMAX+(XDIF-YDIF)/2.
YMIN=YMIN-(XDIF-YDIF)/2.
END IF
C COMPUTE STEP INTERVAL FOR PLOT
IF(IERR.NE.999)THEN
STEP=(XMAX-XMIN)/.18
ELSE
STEP=(XMAX-XMIN)/.8
END IF
RETURN

60 END

*PRTS .LDATA
SUBROUTINE PLOTO
C
FUNCTION:
CONTROLS (1) GENERATION OF TITLE FOR EACH PAGE OF MOUNT
PLOTS AND (2) GENERATION OF MOUNT PLOTS WITH DISPLA
PLOT PACKAGE
C
COMMON/LOAD/XMIN,XMAX,IPGE,YMIN,YMAX,TITLE,IST,ILT,IX,XOR,
*YOR,VARX,VARV,IERR,STEP,DC(250),DX(250,12),DY(250,12)
COMMON/PHIHOR/RUNID,VARI,ISITID,TIME,JOPT,DELTA,IPGVER
CHARACTER*6 RUNID,VARI,JOPT,DELTA,IPGVER,TITLE(9),VARI(12),VARY(12)
CHARACTER*2 TGT(12),TADD,TDROP,TNEW(12),TOLD(12),SSCL*8,SC*7
CHARACTER*4 FRAM,' ','TEMP*2,' '
DIMENSION DAT(12,2),NPT(12)
C
ENTRY PLOTO
C
FUNCTION:
GENERATE TITLE INFO. FOR PAGE OF MOUNT PLOTS
C
CALL RESET('PHYSOR')
CALL AREA2D(1..8)
CALL HEIGHT(.0I8)
C
PRINT X-SCALE VALUE FOR THIS PAGE OF PLTS (METERS)
SCALE=XMAX-XMIN
ENCODE(11,SC)SCAL
FORMAT(F7.6)
SUBSTR(SSCL,1,7)=SC
SUBSTR(SSCL,7,1)='$'
CALL MESSAG(SSCL,100..3g..925)
C
ENTRY PLOT2(IFM)
C
FUNCTION:
GENERATE MOUNT PLOT FOR A FRAME OF DATA
C
ARGUMENT DEFINITION:
IFM: CURRENT FRAME COUNT
C
ENTRY PLOT2(IFM)
C
FUNCTION:
PLOT DATA POINTS
C
CALL HEIGHT(.01)
CALL PHYSOR(XOR,YOR)
CALL AREA2D(.18.,.18)
WRITE BLOCK NO. CORRESPONDING TO DATA PT.

XFM=.153
YFM=.165
ENCODE(7.,FRAM)IFM

FORMAT(13)
SUBSTR(FRAM,4,1)='$
CALL MESSAGE(FRAM,100,XFM,YFM)

PLOT SYMBOL FOR CHAIR DATA (IF CHAIR DATA AVAILABLE)

IF(DC(IST).GT.990)THEN
  CONTINUE
ELSE
  XCH=.005
  YCH=.165
  CALL MESSAGE('C',1,XCH,YCH)
END IF

SCALE DATA TO PRESENT PLOTTING AREA

CALL GRAF(XMIN,STEP,XMAX,YMIN,STEP,YMAX)

DRAW BLOCK AROUND SUBPLOT

CALL RLVEC(XMIN,YMIN,XMIN,YMAX,0000)
CALL RLVEC(XMAX,YMAX,XMIN,YMAX,0000)
CALL RLVEC(XMAX,YMAX,XMAX,YMIN,0000)
CALL RLVEC(XMIN,YMIN,XMAX,YMIN,0000)

DO 50 J=1,IX
  X.XPOSN(DAT(J,1),DAT(J,2))
  Y.YPOSN(DAT(J,1),DAT(J,2))
50 CONTINUE

LOAD DATA FOR ALL TARGETS IN THIS FRAME

DAT(NTHIS,1)=ABS(DX(IST,J))
DAT(NTHIS,2)=ABS(DY(IST,J))

IDENTIFY MANUAL ENTRIES

IF(DX(IST,J).LT.0.)NPT(NTHIS)=I
TGT(NTHIS)=SUBSTR(VARX(J),4,2)
IF(SUBSTR(VARX(J),4,2).NE.SUBSTR(VARY(J),4,2))THEN
  PRINT 52
  'STOP 'PLOTD'
END IF

DO 60 J=1,NTHIS
  X=XPSON(DAT(J,1),DAT(J,2))
  Y=YPOSN(DAT(J,1),DAT(J,2))
60 CONTINUE

IF(NTHIS.NE.0)THEN
  DO 60 J=1,NTHIS
  END IF
IF(TGT(J).LE.'09') THEN
   XT=X
   YT=Y
   XM=X-.004
   YM=Y-.009
   SUBSTR(TEMP,1,1)=SUBSTR(TGT(J),2,1)
   IT=1
ELSE
   XT=X-.006
   YT=Y
   XM=X-.007
   YM=Y-.009
   TEMP=TGT(J)
   IT=2
END IF
C PLOT TARGET
CALL MESSAG(TEMP,IT,XT,YT)
C PLOT CIRCLE AROUND TARGET IF MANUAL ENTRY
IF(NPT(J).NE.0) THEN
   CALL HEIGHT(.029)
   CALL INTNO(0,XM,YM)
   CALL HEIGHT(.01)
END IF
60 CONTINUE
INS=NTHIS-1
C CONNECT TARGETS WITH LINES
DO 68,J=1,INS
   JF=J+1
   DO 68,J2=J1,NTHIS
      CALL RLVEC(DAT(J,1),DAT(J,2),DAT(J2,1),DAT(J2,2),0.0000)
   CONTINUE
68 CONTINUE
C DETERMINE WHICH TARGETS HAVE BEEN ADDED OR DELETED FROM LAST FRAME
IF(IST.NE.I) THEN
   IF(NTHIS.NE.0.OR.NLAST.NE.0) THEN
      IF(NTHIS.NE.0) THEN
         DO 112,J=1,NTHIS
            TNEW(J)=TGT(I)
         END DO
      ELSE IF(NLAST.NE.0) THEN
         DO 113,J=1,NLAST
            TOLD(J)=TNEW(K)
            IF(IBOTH(N).EQ.1) THEN
               TOLD(J)=''
               TDROP=''
            ELSE IF(TOLD(J).NE.TNEW(K)) THEN
               IBOTH=0
               TDROP=''
            ELSE
               TDROP=''
            END IF
         END DO
      END IF
   END IF
END IF
DO 114,J=1,NLAST
   TDROP=TOLD(J)
114 CONTINUE

END IF
115 DO 115,K=1,NTHIS
116 IF(TNEW(K).NE.' ')TADD=TNEW(K)
117 ELSE IF(NTHIS.EQ.0)THEN
118 IF(NLAST.EQ.0)THEN
119 DO 214,J=1,NLAST
120 IF(TOLD(J).NE.' ')TDROP=TOLD(J)
121 END IF
122 ELSE IF(NTHIS.NE.0)THEN
123 IF(NLAST.EQ.0)THEN
124 J=1,N-LAST
125 IF(TOLD(J).NE.' ')TDROP=TOLD(J)
126 END IF
127 END IF
128 ELSE IF(NTHIS.EQ.0 .AND. NLAST.EQ.0)THEN
129 GO TO 111
130 END IF
131 C STORE TARGET NOS. FOR THIS FRAME
132 C DO 116,J=1,NTHIS
133 TOLD(J)=TGT(J)
134 NLAST=NTHIS
135 C PLOT SYMBOL IF NEW TARGET WAS VISIBLE IN THIS FRAME
136 XNEW=.005
137 YNEW=.007
138 IF(TADD.LE.'09')THEN
139 SUBSTR(TEMP,1,1)=SUBSTR(TADD,1,2)
140 IA=1
141 ELSE
142 TEMP=TADD
143 IA=2
144 END IF
145 CALL MESSAG(TEMP,IA,XNEW,YNEW)
146 C PLOT SYMBOL IF A TARGET WAS DROPPED IN THIS FRAME
147 XDROP=.159
148 YDROP=.007
149 IF(SUBSTR(TDROP,1,1).EQ.'0')SUBSTR(TDROP,1,2)=''
150 CALL MESSAG(TDROP,2,XDROP,YDROP)
151 CONTINUE
152 C END SUBPLOT
153 C CALL RESET('HEIGHT')
154 CALL ENDDR(O)
155 RETURN
156 END

@PRT,S,MIXY
SUBROUTINE XYPLOT

C FUNCTION: PLOT X AND Y DISPLACEMENT VALUES FOR GIVEN RUN AND ANATOMICAL MOUNT; EACH PHOTO TARGET PLOTTED ON SAME GRAPH

DIMENSION DXARY(250),DYARY(250)
COMMON/LOAD/XMIN,XMAX,IPGE,YMIN,YMAX,TITLE,IST,ILT,IX,XOR,
*YOR,VARX,VARY,IERR,STEP,DC(250),DX(250),DY(250)
COMMON/PHTHDR/RUNID,VARID,ISITID,TITLE,JDTDIG,DELTA,IPGVER
CHARACTER*6 RUNID,VARID,JDTDIG,IPGVER,TITLE(9),VARX(12),VARY(12)
CHARACTER*6 XYTIT(9).TAR*2,TIME.B.DATE*B
3 DATA NPTS.ITAR/2.O/
C LOAD TITLE INFO.
DD 12.I=1,9
XYTIT(1)=TITLE(1)
CONTINUE
XYTIT(5)="'
XYTIT(6)="'
XYTIT(7)="'
XYTIT(8)="'
XYTIT(9)="'
CALL ADATE(DATE,TIME)
XYTIT(4)=TITLE(5)
SUBSTR(XYTIT(5),4,2)=SUBSTR(DATE,1,2)
SUBSTR(XYTIT(6),1,2)=SUBSTR(DATE,3,2)
SUBSTR(XYTIT(6),4,2)=SUBSTR(DATE,5,2)
C SUPPRESS LISTING FOR OUT OF RANGE POINTS
CALL NOCHEK
CALL RESET('PHYSOR')
C LABEL AXES
CALL AREA2D(.8,.8)
CALL XNAME('FILM PLANE X DISPLACEMENT (METERS)',.35)
CALL YNAME('FILM PLANE Y DISPLACEMENT (METERS)',.35)
C PRINT HEADING
CALL HEIGHT(.027)
CALL MESSAG(XYTIT,.5,.92)
CALL HEIGHT(.016)
C COMPUTE MIN AND MAX VALUES FOR X AND Y
IERR=999
CALL MXY
IERR=0
C SCALE DATA FOR PLOTS
CALL GRAF(XMIN,STEP,XMAX,YMIN,STEP,YMAX)
C PLOT X-Y CONTOUR FOR EACH TARGET
CALL HEIGHT(.022)

DO 10, 1 = 1, IX
LINE = MOD(1.5)

IF(LINE.EQ.0) THEN
  CALL DOT
ELSE IF(LINE.EQ.1) THEN
  CALL RESET('DOT')
ELSE IF(LINE.EQ.2) THEN
  CALL CHNDOT
ELSE IF(LINE.EQ.3) THEN
  CALL DASH
ELSE IF(LINE.EQ.4) THEN
  CALL CHNDASH
END IF

C LOAD DATA INTO PLOT ARRAY AND CHANGE NEGATIVE SIGNS
C
DO 30, J = 1, 250
DXARY(J) = DX(J, 1)
DYARY(J) = DY(J, 1)

IF(DXARY(J).LT.0.) THEN
  DXARY(J) = -1.*DXARY(J)
END IF

IF(DYARY(J).LT.0.) THEN
  DYARY(J) = -1.*DYARY(J)
END IF

CONTINUE

DO 35, J = POS, 250
ZF(J) = (DXARY(J).GT.990. .OR. DYARY(J).GT.990.)

CONTINUE
ELSE
  NPTS = 0
  IPLOT = J
  GO TO 15
END IF

CONTINUE
GO TO 10

IF(DXARY(K).GT.990. .OR. DYARY(K).GT.990.) THEN
  IF(ITAR.EQ.0) THEN
    SX = XPOSN(DXARY(IPLOT), DYARY(IPLOT))
    SY = YPOSN(DXARY(IPLOT), DYARY(IPLOT))
    TAR = SUBSTR(VARX(I), 4, 2)
    IF(SUBSTR(TAR, 1, 1).EQ. 'O') SUBSTR(TAR, 1, 1) = ' ' 
    CALL MESSAG(TAR, SX, SY)
    ITAR = 1
  END IF
  CALL CURVE(DXARY(IPLOT), DYARY(IPLOT), NPTS, 0)
  NPOS = K + 1
  IF(K.LT.250) GO TO 5
ELSE
  NPTS = NPTS + 1
  IF(K.EQ.250) THEN
    IF(ITAR.EQ.0) THEN
      SX = XPOSN(DXARY(IPLOT), DYARY(IPLOT))
    END IF
  END IF
SY+YPOSN(DXARY(IPLT),DYARY(IPLT))
TAR=SUBSTR(VARX(I),4,2)
IF(SUBSTR(TAR,1,1).EQ.'O')SUBSTR(TAR,1,1)=''
CALL MESSAG(TAR,2, SX, SY)
ITAR=1
END IF
CALL CURVE(DXARY(IPLT),DYARY(IPLT),NPTS,0)
END IF
CONTINUE
CALL CURVE(DXARY(IPLT),DYARY(IPLT),NPTS,0)
END IF
40 CONTINUE
10 CONTINUE
CALL RESET('HEIGHT')
CALL RESET('XNAME')
CALL RESET('YNAME')
CALL RESET('DOT')
CALL ENDPL(0)
RETURN
END

*PRT.S .TIMER
LAMBERT*TPFS(O).XQTSGN(3)
  &Q70 MOO051/51N
  &. CONTROL STATEMENT IMMEDIATELY AFTER &Q70 STATEMENT
  &. POS 5 - NO. INPUT UNITS (1, 2 OR 3)
  &. POS 10 - LOWEST RUN NO. TO BE PROCESSED
  &. POS 20 - HIGHEST RUN NO. TO BE PROCESSED
  &TEST TNE/0/S6
  &JUMP OUT
  &ASG.TF PCSXXXXXXX..U9S.SAVEOS . PCSXXXXXXX
  &COPY.GM 9..PCSXXXXXXX.
  &COPY.GM 9..PCSXXXXXXX.
  &COPY.GM 9..PCSXXXXXXX.
  &TEST2:
  &TEST TG/1/S6
  &JUMP TEST3
  &COPY.GM 10..PCSXXXXXXX.
  &COPY.GM 10..PCSXXXXXXX.
  &COPY.GM 10..PCSXXXXXXX.
  &TEST3:
  &TEST TG/2/S6
  &JUMP OUT
  &COPY.GM 11..PCSXXXXXXX.
  &COPY.GM 11..PCSXXXXXXX.
  &COPY.GM 11..PCSXXXXXXX.
  &OUT:
  &FREE.I PCSXXXXXXX.
  &. EXITED FROM .XQTSGN

&PR&T.5 .SGFLIP
FUNCTION: THIS ASCII FORTRAN DRIVER PROGRAM PERFORMS (1) SIGN CHANGE OF NEGATIVE DATA PTS. (MANUAL ENTRIES) TO POSITIVE AND (2) WRITES REFORMATTED DATA RECORD TO MASS STORAGE

DIMENSION DC(250)
COMMON/PHTHDR/RUNID,VARID,ISITIO,FITIME,JDTDIG,DELTA,IPGVER
CHARACTER*6 RUNID,VARID,JDTDIG,IPGVER,FRUN,LRUN,RUNS(200)
*.FIRST,LAST,ENC+4,SITE*1
READ USER PARAMETERS IN CARD IMAGE
READ(5,6)IFILE,FRUN,LRUN
DATA MAX/320/

READ(5,6)IFILE,FRUN,LRUN
DATA MAX/320/

COMPUTE RUN RANGE
FIRST=SUBSTR(FRUN,3,4)
LAST=SUBSTR(LRUN,3,4)
DECOD(4,FIRST)IFIRST
DECOD(4,LAST)ILAST

COMPUTE INPUT UNIT ASSIGNMENTS
ILIM=IFILE+8
IU=9
CONTINUE
PRINT 3,IU,ILIM,FRUN,LRUN
FORMAT(210.2(4X,A6))
NIU=I+IU
BEGIN RUN LOOP
IRN=ABS(ILAST-IFIRST)+1
INC=IFIRST
DO 100,I=1,IRN
SUBSTR(RUNS(I),1,2)="LX"
ENCOD(17,ENC)INC
SUBSTR(RUNS(I),3,4)=ENC
INC=INC+1
ATTEMPT TO SET SECTOR TO HEADER FOR RUN
CALL SETSCT(IU,RUNS(I),"PHDATA",1,ISITE,NREC,IER)
IF(IER.NE.0)GO TO 100
CALL GETRCD(IU,MAX,RUNID,VARID,ISITE,IDL,DC,IER)
IF(IER.NE.0)THEN
PRINT 13,IER
FORMAT(1X,ST"ATUS",.03)
STOP 'HEADER ERROR'
END IF
SET SECTOR FOR TIME RECORD
VARID="TS"
ENCODE(I1,SITE)ISITID
FORMAT(I1)
SUBSTR(VARID,1,1)=SITE
CALL SETSCT(IU,RUNID,VARID,1,ISIT,NREC,IER)
IF(IER.NE.0)GO TO 110
C
READ NEXT RECORD
C
DO 10,K=1,100
CALL SECTOR(IU,ISEC)
CALL GETRCD(IU,MAX,RUNID,VARID,ISIT,IDL,DC,IER)
IF(IER.NE.0)THEN
  IF(IER.EQ.1)THEN
    GO TO 100
  ELSE
    GO TO 110
  END IF
END IF
INEG=0
CHANGE SIGNS OF NEGATIVE NOS.
DO 8,M=1,250
  IF(DC(M).LT.0.0)THEN
    INEG=1
    DC(M)=-DC(M)
  END IF
CONTINUE
WRITE RECORD BACK TO MASS STORAGE
IF(INEG.NE.0)THEN
  CALL SETADR(IU,ISEC)
  CALL PUTRCO(NIU,RUNID,VARID,ISIT,IDL,DC,IER)
  IF(IER.NE.0)GO TO 110
END IF
CONTINUE
ERROR ROUTINE
PRINT 111,RUNID,IER
FORMAT(' ERROR IN SIGN FLIP ROUTINE FOR RUN ',A6,' STATUS=',I3)
CONTINUE
IF(IU.LT.ILIM)THEN
  IU=IU+1
  GO TO 1
END IF
END
SUBROUTINE SETSCT

a. Function: Subroutine SETSCT is an ASCII FORTRAN program designed to control the positioning of sector address for either a photo header record or a photo time record. The calling program must provide the logical unit number of the photo data file and the ASCII variable RUNID; these arguments are used to locate the desired run in the random access directory. In addition, the user provides the variable name (see Calling Sequence) and the desired occurrence of the time record as arguments. There may be more than one PDS (Photo Digitizing System) physical record/target in a run, hence it is necessary to designate which one is desired (even if there is only one PDS physical record/target).*

Entry DATSCT of subroutine DIRBFR is called in order to retrieve the sector address of the desired record. This address is used in a subsequent call to Entry SETADR of subroutine FORTIO in order to correctly position the data file.

This subroutine returns the site ID of the data file as argument ISITE. Additionally, the number of PDS physical records required per target for the desired run is returned in argument NREC.

b. Subroutines (see reference 1):

DIRBFR/Entry DATSCT, FORTIO/Entry SETADR

c. Calling Sequence:

CALL SETSCT (IUNIT, RUNID, VARID, NPHYS, ISITE, NREC, IER)

* In the event more than 250 frames of photo data were digitized for a target, an extra PDS physical record is required for each additional 250 frames.
SUBROUTINE SETSCT (Continued)

where:

<table>
<thead>
<tr>
<th>ARGUMENT</th>
<th>DESCRIPTION</th>
<th>TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>IUNIT</td>
<td>Logical unit number of photo directory and data file</td>
<td>Integer</td>
</tr>
<tr>
<td>RUNID</td>
<td>Run number for requested sector positioning</td>
<td>6 char</td>
</tr>
<tr>
<td>VARID*</td>
<td>Variable name for requested sector positioning</td>
<td>6 char</td>
</tr>
<tr>
<td>NPHYS</td>
<td>Desired occurrence of time record (Nphys=1 unless more than 250 frames were digitized)</td>
<td>Integer</td>
</tr>
<tr>
<td>ISITE</td>
<td>Returned site ID of data file</td>
<td>Integer</td>
</tr>
<tr>
<td>NREC</td>
<td>Returned number PDS physical records required for each photo target</td>
<td>Integer</td>
</tr>
<tr>
<td>IER</td>
<td>Returned error status code (See Appendix 4e for code definitions)</td>
<td>Octal</td>
</tr>
</tbody>
</table>

* This argument identifies whether the sector is to be set for a header record (VARID='PHDATA') or a time record (VARID='ITS' for site 1).
SUBROUTINE GETR2I

a. Function: This subroutine is an ASCII FORTRAN program designed to control the execution of sequential binary reads of FORTRAN V photo (1) header records, (2) time records and (3) displacement records. GETRCD utilizes Entry READIO of subroutine FORTIO to retrieve the data record from mass storage (the data record is stored in named common IOBUFF). It then extracts the run number (RUNID) and variable name (VARID) from the data record and converts each from FIELDATA to ASCII with system function subprogram FD2ASC; these converted character variables are returned to the calling program. The site ID (ISITE) of the data record is also returned as an argument.

If the retrieved record is either a time record or a displacement record, the word length of the data array (IDL) and the data array itself (DATARY) are returned to the calling program. If the retrieved record is a header these arguments have no meaning. However, when a header record is read GETRCD calls subroutine DECHED in order to (1) convert all alphanumeric data in the header to ASCII and (2) store all header information in named common PHTHDR.

The user specifies MAXLEN as the maximum word length of the data record being read from mass storage. GETRCD utilizes function subprogram EXPAND to insure that the user has specified a maximum length large enough to accommodate those sentinels and checksums which will be read from mass storage as part of the input record. If MAXLEN is too small, an appropriate error status code is returned.

b. Subroutines (see reference 1):

FORTIO/Entry READIO, FD2ASC, ASC2FD, DECHED, EXPAND

c. Calling Sequence:

Call GETR2I (IUNIT, MAXLEN, RUNID, VARID, ISITE, IDL, DATARY, IER)
SUBROUTINE GETRCD (Continued)

where:

<table>
<thead>
<tr>
<th>ARGUMENT</th>
<th>DESCRIPTION</th>
<th>TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>IUNIT</td>
<td>Logical unit number of photo data and directory file</td>
<td>Integer</td>
</tr>
<tr>
<td>MAXLEN</td>
<td>Maximum word length of input record</td>
<td>Integer</td>
</tr>
<tr>
<td>RUNID</td>
<td>Returned run number of input record</td>
<td>6 char</td>
</tr>
<tr>
<td>VARID*</td>
<td>Returned variable name of input record</td>
<td>6 char</td>
</tr>
<tr>
<td>ISITE</td>
<td>Returned site ID</td>
<td>Integer</td>
</tr>
<tr>
<td>IDL</td>
<td>Returned word length of data array</td>
<td>Integer</td>
</tr>
<tr>
<td>DATARY</td>
<td>Returned data array</td>
<td>Real</td>
</tr>
<tr>
<td>IER</td>
<td>Returned error status code (see Appendix 4e for code definitions)</td>
<td>Octal</td>
</tr>
</tbody>
</table>

d. Common Block Usage:

COMMON/IOBUFF/IOBUFF(4100)

where:

<table>
<thead>
<tr>
<th>VARIABLE</th>
<th>DESCRIPTION</th>
<th>TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>IOBUFF(4100)</td>
<td>Input array containing FORTRAN (FIELDATA) photo data record read from mass storage (see Figures 2c-2e for data record formats)</td>
<td>Integer</td>
</tr>
</tbody>
</table>

The contents of this array are passed as an argument to the header decoding subroutine (DECHED). Alternatively, if the photo input record is either a time record or a displacement record, the data array is extracted from IOBUFF for return to the calling program.

* VARID='PHDATA' for a photo header record
  VARID='ITS ', '2TS ' or '3TS ' for a photo time record (first character represents site ID)
  See Figure 4 for possible values of VARID for a photo displacement record
SUBROUTINE PUTRCD

a. Function: Subroutine PUTRCD is an ASCII FORTRAN program designed to control the execution of sequential binary writes of FORTRAN (FIELDATA) photo (1) header records, (2) time records, and (3) displacement records. This routine provides the user with complete directory maintenance. The user may utilize PUTRCD to (1) create a photo data output file with accompanying directory or (2) perform sequential output of photo records without directory maintenance. If the user desires to perform sequential writes without building a corresponding directory, (s)he must transmit the negative of the logical unit number (of the photo data file) as the value of the argument IUNIT in the call statement.

In the event a photo header record is to be written, it is the user's responsibility to provide the ASCII version of the header in named common PHTHDR (see Figure 2c for format). However, if the user utilizes input subroutine GETRCD to read an existing header record, its ASCII version is automatically stored in common. This ASCII version of the header must be converted to FIELDATA; this is accomplished by a call to subroutine ENCHED.

The user must provide PUTRCD with the (1) logical unit number of the photo data output file (IUNIT), (2) run number of the output record (RUNID), (3) variable name of the output record (VARID*), (4) site ID of the output record (ISITE), (5) word length of the output data array (IDL) and (6) output data array (DATARY). Values for items (5) and (6) must be assigned for all output records except header records. Items (1) through (4) are passed by PUTRCD to subroutine DIRCTY where the necessary directory entries are created.

Once the directory has been updated, PUTRCD uses the word length of the output array (IDL) and the output array itself (DATARY) to construct the output record. Entry WRITIO of subroutine FORTIO is then called to perform the sequential write. The output record is passed down in an argument array (IOBUFF), and is also stored in named common IOBUFF.

b. Subroutines (see reference 1):
ASC2FD, ENCHED, FORTIO/Entry WRITIO

c. Calling Sequence:
Call PUTRCD (IUNIT, RUNID, VARID, ISITE, IDL, DATARY, IER)

* VARID='PHDATA' for a photo header record
VARID='ITS ', '2TS ' or '3TS ' for a photo time record (first character represents site ID)
See Figure 4 for possible values of VARID for a photo displacement record

80 Appendix 4c
SUBROUTINE PUTRCD (Continued)

where:

<table>
<thead>
<tr>
<th>ARGUMENT</th>
<th>DESCRIPTION</th>
<th>TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>IUNIT</td>
<td>Logical unit number of photo data and directory file</td>
<td>Integer</td>
</tr>
<tr>
<td>RUNID</td>
<td>Run number of output data record</td>
<td>6 char</td>
</tr>
<tr>
<td>VARID</td>
<td>Variable name of output data record</td>
<td>6 char</td>
</tr>
<tr>
<td>ISITE</td>
<td>Site ID of output record</td>
<td>Integer</td>
</tr>
<tr>
<td>IDL*</td>
<td>Word length of output data array</td>
<td>Integer</td>
</tr>
<tr>
<td>DATARY*</td>
<td>Output data array</td>
<td>Real</td>
</tr>
<tr>
<td>IER</td>
<td>Returned error status code (see Appendix 4e for code definitions)</td>
<td>Octal</td>
</tr>
</tbody>
</table>

d. Common Block Usage:

(1) COMMON/PHTHDR/RUNID, VARID, ISITID, FITIME, JOTDIG, DELTA, IPGVER

(See Figure 2c for format of named common PHTHDR)

(2) COMMON/IOBUFF/IOBUFF(4100)

where:

<table>
<thead>
<tr>
<th>VARIABLE</th>
<th>DESCRIPTION</th>
<th>TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>IOBUFF(4100)</td>
<td>Output array containing FORTRAN (FIELDATA) photo (1) header record, (2) time record or (3) displacement record to be written to core (see Figures 2c-2e for data record formats)</td>
<td>Integer</td>
</tr>
</tbody>
</table>

* This argument has no meaning for header record.
SUBROUTINE EOFPUT

a. Function: This ASCII FORTRAN program is designed to control the writing of an "end-of-file" to the end of a data file. Initially EOFPUT calls subroutine DIRCTY in order to have the output file positioned for the write. Once this is done, the "end-of-file" is written by virtue of a call to entry WRITIO of the subroutine FORTIO. Control is then returned to the calling program. In the event the user wishes to write an "end-of-file" without directory maintenance, (s)he must pass down the negative of the logical unit number (of the photo data file) as the value of the argument IUNIT.

b. Subroutines (see reference 1):

DIRCTY, FORTIO/Entry WRITIO

c. Calling Sequence:

Call EOFPUT (IUNIT, IER)

where:

<table>
<thead>
<tr>
<th>ARGUMENT</th>
<th>DESCRIPTION</th>
<th>TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>IUNIT</td>
<td>Logical unit number of photo data and directory file</td>
<td>Integer</td>
</tr>
<tr>
<td>IER</td>
<td>Returned error status code (see Appendix 4e for code definitions)</td>
<td>Octal</td>
</tr>
</tbody>
</table>
ERROR CODE DEFINITIONS

IER (Octal) = error status code of the I/O function

<table>
<thead>
<tr>
<th>IER (Octal)</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Normal I/O completion</td>
</tr>
<tr>
<td>1</td>
<td>&quot;End-of-file&quot; has been identified</td>
</tr>
<tr>
<td>5</td>
<td>Attempted to read from unassigned area of mass storage</td>
</tr>
<tr>
<td>20</td>
<td>Write attempted on read only file</td>
</tr>
<tr>
<td>21</td>
<td>Reference made to unassigned file</td>
</tr>
<tr>
<td>22</td>
<td>Write attempted beyond assigned area of mass storage</td>
</tr>
<tr>
<td>25</td>
<td>Maximum length of input/output array exceeded</td>
</tr>
<tr>
<td>26</td>
<td>FORTRAN sentinels are unequal - read was not executed</td>
</tr>
<tr>
<td>30</td>
<td>Run number not found in run directory</td>
</tr>
<tr>
<td>31</td>
<td>Variable name not found in variable directory</td>
</tr>
<tr>
<td>32</td>
<td>Checksum error</td>
</tr>
<tr>
<td>33</td>
<td>Run number not last entry in run directory</td>
</tr>
<tr>
<td>34</td>
<td>Update in place aborted due to different data record lengths</td>
</tr>
<tr>
<td>35</td>
<td>Invalid FORTRAN sentinel - word length indicated for physical record greater than 249</td>
</tr>
<tr>
<td>36</td>
<td>Invalid FORTRAN sentinel - record number indicated for first physical record not equal to 1</td>
</tr>
</tbody>
</table>