N O T I C E

NASA-CR-167544

E82-10344

FOREST RESOURCE INFORMATION SYSTEM

PHASE III SYSTEM TRANSFER REPORT

for the period

1 April 1979 to 31 December 1980

Prepared for

NATIONAL AERONAUTICS and SPACE ADMINISTRATION

Johnson Space Center
Earth Observations Division
Houston, Texas 77058

Contract: NAS 9-15325
Technical Monitor: R. E. Joosten/SH2

Submitted by:

The Laboratory for Applications of Remote Sensing
Purdue University
West Lafayette, Indiana 47906

Principal Investigator: R. P. Mroczynski

(E82-10344) FOREST RESOURCE INFORMATION SYSTEM. PHASE 3: SYSTEM TRANSFER REPORT Final Report, 1 Apr. 1979 - 31 Dec. 1980 (Purdue Univ.) 142 p HC A07/MF A01 CSCL 02F

N82-25606

Unclas
G3/43 00344

Star Information Form

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle Forest Resource Information System, Phase III System Transfer Report		5. Report Date 30 July 1981
		6. Performing Organization Code
7. Author(s) FRIS Project Staff, R. P. Mroczynski, editor		8. Performing Organization Report No. 073081
9. Performing Organization Name and Address Laboratory for Applications of Remote Sensing Purdue University West Lafayette, IN 47906		10. Work Unit No.
		11. Contract or Grant No. NAS 9-15325
12. Sponsoring Agency Name and Address R. E. Joosten/SH2 NASA/Johnson Space Center Houston, TX 77058		13. Type of Report and Period Covered Final: 1 Apr 79-31 Dec 80
		14. Sponsoring Agency Code

15. Supplementary Notes

This report represents the final contract report for the Forest Resource Information System Applications Pilot Test.

16. Abstract

This report documents the FRIS System Transfer Phase Activities, including sections on image processing and preprocessing technologies, applications tests, and Benefit/Cost analysis.

17. Key Words (Suggested by Author(s)) Forest Inventory Technology Transfer Data Bases Landsat MSS Cost Studies	18. Distribution Statement

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price

FOREST RESOURCE INFORMATION SYSTEM

PHASE III SYSTEM TRANSFER REPORT

for the period

1 April 1979 to 31 December 1980

Prepared for

NATIONAL AERONAUTICS and SPACE ADMINISTRATION

Johnson Space Center
Earth Observations Division
Houston, Texas 77058

Contract: NAS 9-15325
Technical Monitor: R. E. Joosten/SH2

Submitted by:

The Laboratory for Applications of Remote Sensing
Purdue University
West Lafayette, Indiana 47906

Principal Investigator: R. P. Mroczynski

Index

APPENDICES

LIST OF FIGURES

LIST OF TABLES

Appendices

Exhibits

FRIS PROJECT EXECUTIVE SUMMARY

The Forest Resource Information System Project (FRIS) was a cooperative effort between the National Aeronautics and Space Administration (NASA) and St. Regis Paper Co. Purdue University's Laboratory for Applications of Remote Sensing (LARS), under contract to NASA, supplied technical support to the project.

FIRS was an Applications Pilot Test (APT) Project funded by NASA. The project was interdisciplinary in nature involving expertise from both the public and private sectors. FRIS also represented the first APT to involve a large broad base forest industry in a cooperative with the government and the academic communities.

Purpose

The goal of FRIS was to demonstrate the feasibility of using computer-aided analysis techniques applied of Landsat Multispectral Scanner Data to broaden and improve the existing St. Regis forest data base, thereby creating the foundation of a dynamic information system. The successful demonstration of this technology during the first half of the project led to the establishment by St. Regis of an independently controlled operational forest resource information system in which Landsat data makes a significant contribution. FRIS can be viewed by the user community as a model of NASA's involvement in practical application and effective use of space technology. Additionally, FRIS served to demonstrate the capability of Landsat MSS data and machine-assisted analysis technology to private industry by:

- Determining economic potentials,
- Providing visibility and documentation, and
- The ability to provide timely information and thus serve management needs.

The ultimate long term successfulness of FRIS will be measured through future development of remote sensing technology within the forest products industry.

Scope

FRIS was funded as a modular of phased project. The original project concepts were developed in 1973, and a formal project plan was submitted to NASA in 1976. The project officially began in October 1977 after the signing of a cooperative agreement between NASA and St. Regis; and after the completion of contractual arrangements with Purdue University. The project officially ended in May 1981.

Summary

In retrospect, the FRIS project addressed more than the demonstration and implementation of remote sensing technology in an operational industrial forestry environment. Conceptually, the FRIS project dealt with the entire range of activities which are required for intensive forest management. The success of FRIS depends on its ability to intergrate and manipulate digital inventory data, maps, and land cover to provide information to serve management needs. Key to meeting these requirements is the geographic information system acquired by St. Regis.

Contractually, LARS provided remote sensing support for FRIS. The remote sensing elements of the project were basically; a) a proof-of-concept, and b) the transfer and implementation of system capabilities to St. Regis. The demonstration phase of the FRIS project proved the concept that computer-aided analysis of Landsat MSS data could effectively be used to delineate, map, and monitor the southeastern forest resources. Based on encouraging demonstration results, the decision was made by St. Regis management to pursue the System Transfer portion of the project. This report addresses those activities.

The most significant aspect of the transfer and implementation of the image processing technology to St. Regis was the level of commitment of the user. Without the dedicated efforts of the St. Regis staff and support of their management, FRIS would have been just another technology evaluation project. In the future, FRIS may be looked back on as a pioneering effort which fostered the application of remote sensing technology in forestry. For the present, FRIS is an example of how man's imagination and ingenuity help him do his job.

Key accomplishments of the FRIS project were:

- Satellite acquired data provides important information for forest management.
- Effective use of satellite acquired data requires that it be combined with other data sources. This combination is most efficiently done with an automated mapping system.
- FRIS represents a multi-functional information system wherein the independent functions of imagery, mapping, and inventory are brought together to form an integrated digital data base.
- The FRIS data base provides management with an accessible and retrievable information sources in a timely and efficient manner. Furthermore, this data base is totally responsive to changes and modifications to the data as they occur.

ACKNOWLEDGEMENTS

The scope of this report in terms of technical detail and background is well beyond the productive capacity of a single individual. The principal investigator, and editor, are grateful to a number of individuals who have made this, and previous FRIS reports possible. Special thanks are due also to the other members of the FRIS Steering Committee, Bob Barker of St. Regis and Rig Joosten of NASA, for their patience and understanding. The principal investigator is especially indebted to Brenda Prather for her tolerance and patience in typing this manuscript.

1.0 INTRODUCTION

The third phase of the Forest Resource Information System Application Pilot Test designated as the System Transfer Phase, was designed to transfer remote sensing technology to St. Regis. The third phase spanned an 21-month period and involved the definition, transfer and preliminary implementation of an image processing capability at St. Regis.

The system definition activity began during Phase II with the identification of user's needs in terms of remote sensing inputs. These needs were transformed into system performance requirements and a list of functional specifications. Once these specifications were defined, available software and hardware systems were evaluated in terms of their suitability for FRIS.

A more complex task undertaken during Phase III involved technology transfer. These activities were directed at providing St. Regis personnel with the capabilities to understand, analyze and interpret remote sensing data. Since an important aspect of the project goal was to provide St. Regis with an independent capability to utilize the technology, this aspect of Phase III was critical to the ultimate success of FRIS.

The last major emphasis of Phase III related to the implementation and documentation of image processing software transferred to St. Regis. Since St. Regis decided to implement the LARSYS version 3.1 software, a major effort was mounted to upgrade this software. The modified software package is called LARSFRIS and includes the basic elements of LARSYS version 3.1 plus some of the new software that was included in developmental LARSYS or LARSYSDV. Another major activity associated with upgrading the software was an upgrade of the software documentation for inclusion in COSMIC.

Details regarding these activities and other tasks conducted during Phase III are reported in the sections which follow.

2.0 SYSTEM TRANSFER ACTIVITIES

The Forest Resource Information System consists of a forest inventory or tabular component, a graphics component and an image processing component. The primary emphasis of LARS staff involved with the project was directed toward the image processing portion of FRIS, figure 2.1. System transfer activities, therefore, consisted of transferring software and capabilities to St. Regis.

Software transferred to St. Regis included data preparation, or preprocessing software, and data classification software. The basis for the data classification software was LARSYS version 3.1, and developmental LARSYS or LARSYSDV. These elements combined form LARSFRIS.

Capability or the capacity to optimally utilize this software was more difficult to transfer. Therefore, various technology transfer activities were conducted for St. Regis to build their capability. Ultimately, the most effective transfer of technology occurred when St. Regis hired two LARS staff as permanent FRIS employees.

The system transfer activity was dedicated to the physical transfer, implementation, and testing of software that was transferred to St. Regis. The software which was transferred related specifically to those routines required to prepare the Landsat digital data for analysis and those routines used to classify and display the data. These comprise respectfully the Preprocessing software and the LARSYS software. These elements form the FRIS Image Processing Subsystem.

A subgroup of the Preliminary System Design Committee, which was created during Phase II, was responsible for the software transfer and installation task. The subgroup was comprised of personnel from both St. Regis and LARS. This group met in mid-July, 1979 to outline the plan for transfer and installation of software. Highlights of their plan are presented below:

- o LARSYS version 3.1 and LARSYSDV will be the foundation for the FRIS image processing subsystem.
- o LARS data preparation software will be the nucleus of the FRIS preprocessing software.
- o Software will be installed at the St. Regis National Computer Center (NCC) in Dallas, Texas.
- o The software will operate in batch mode on an IBM 3033, or similar computer.
- o User interface will be provided via ROSCOE.

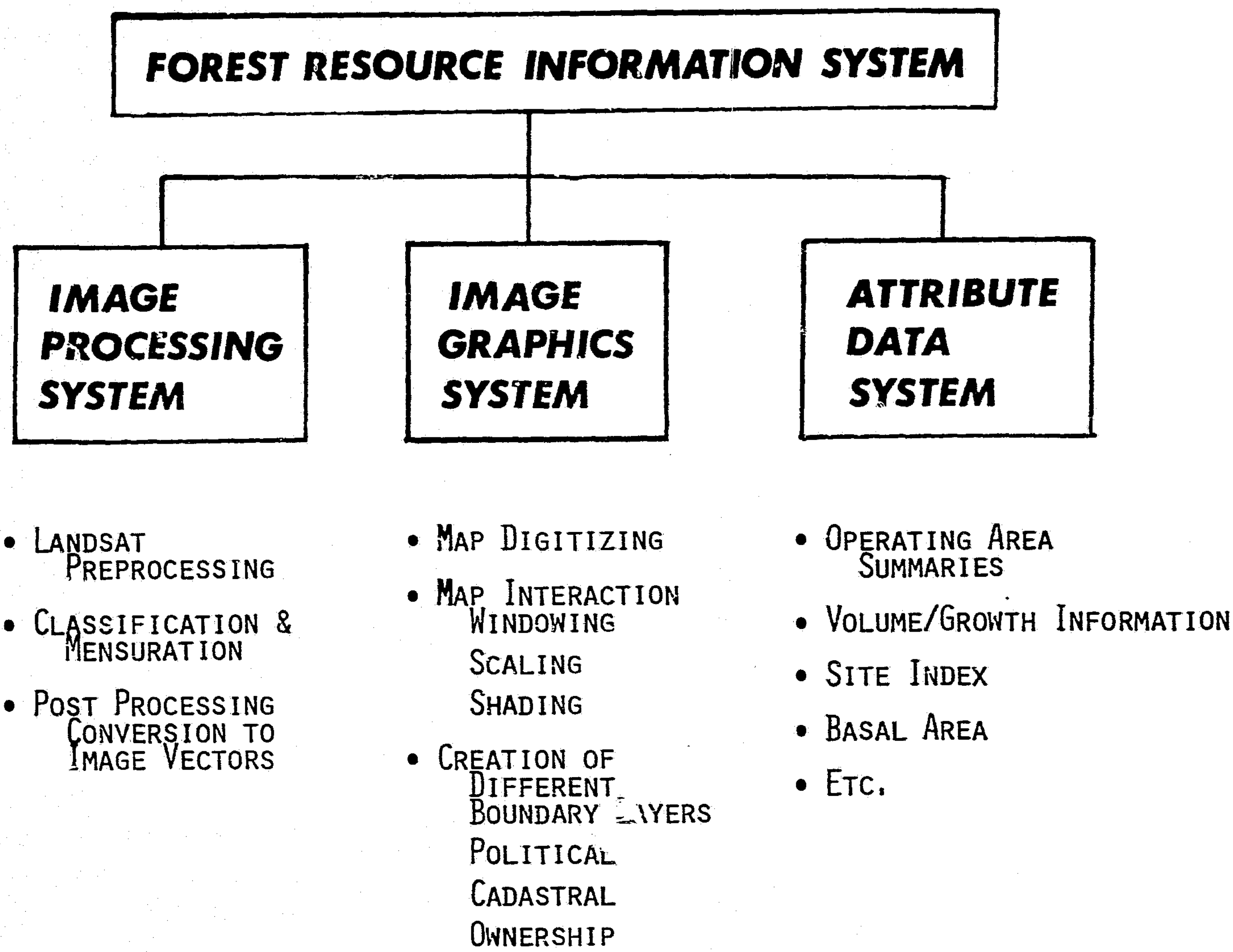


Figure 2.1 The proposed structure for FRIS consisting of three independent subsystems.

o The Landsat CCT data and permanent intermediate files will be maintained on tape.

o Temporary files will be kept on disk.

o Documentation consisting of User's Manuals, System's Manuals, and Program Abstracts will be provided to St. Regis. This documentation will be included in the public domain via COSMIC.

A listing of the LARSYS, LARSYSDV and preprocessing software that was transferred appears in Appendix A. Responsibility for implementation of this software on the St. Regis computer rested with St. Regis personnel. LARS staff provided program tapes, listings and documentation. They also acted as consultants during the software installation, providing assistance when needed.

As an aid to LARS staff during the system implementation activity a remote terminal link to NCC was provided by St. Regis. The terminal, an IBM 3275, operated under ROSCOE protocol, thereby allowing LARS to emulate a St. Regis remote site. Computer output was acquired through a Data 100 printer. The printer operated as a remote job entry terminal and was connected to NCC via a dial-up modem. Support for the RJE station was provided through the FRIS contract.

The System Design Team had the remote hardware to NCC functional at LARS in November, 1979. Once the hardware was operational a ROSCOE training session was held at LARS to train FRIS personnel. At the completion of these activities, which closely coincide with the initial installation of software at NCC, the system transfer activities proceeded.

The LARS/NCC terminal connection was designed to:

o Assist St. Regis staff in debugging the NCC installation of LARSYS and LARSYSDV software.

o Suggest program updates or modifications to St. Regis staff based on ROSCOE remote batch operations on a NCC computer.

o Develop user documentation for batch Preprocessing and LARSYS operations initiated from a remote terminal via ROSCOE.

o Develop user training sessions for St. Regis analysts in the use of the St. Regis/LARSFRIS software.

o Develop analyst aids.

Remote terminal operations between LARS and Jacksonville were maintained during this period to provide for continuing analyst training of St. Regis staff. When the preprocessing and LARSYS software were tested and considered "operational" at NCC the LARS/JAX terminal link was disconnected.

2.1 System Design Task

System design activities began during Phase II with the identification of system requirements and constraints. Preliminary design requirements were focused on:

- o Communications - movement of data and information between computing sites.
- o Resources - identification of system component requirements in terms of; a) hardware, b) software, and c) man power.
- o Costs - financial requirements necessary for system start-up and operations.
- o Documentation - level of system and user explanation necessary to operate the system.
- o Transferability - the ease, or difficulty, of implementing any software module that is an intergral part of the system.
- o Languages - refers to software programming language.
- o Interface = describes how the user would access and manipulate the system.

The constraints that were considered in developing the preliminary system design were:

- o The system that was to be implemented would be specifically tailored to the St. Regis application.
- o The system would be operational, that is, St. Regis would have an independent remote sensing data analysis capability at the end of the Application Pilot Test.
- o The remote sensing components of FRIS (both hardware and software) would have to be attractive in terms of cost to St. Regis management, i.e.:
    - a. reasonable start-up and operating costs,
    - b. relatively quick (aim for 5-year) pay-back period,
    - c. potential cost-efficiencies or cost reductions or cost avoidance associated with the system, and
    - d. require a minimum of new human resources.
- o The system designed should utilize existing and computational resources where feasible.

o The system should be easy to implement.

o The quality of information from the system should be compatible to or better than currently available.

A plan for implementing these concepts was developed during the latter stages of Phase II. This plan became the focus of the system transfer activities of Phase III.

Between the sixth and ninth months of Phase II the System Design Committee met several times to formalize FRIS specifications. The list of functional specifications that were developed by the committee appear in Table 2.2.1. Three vendors of data base systems were asked to demonstrate their systems capabilities and bid on the system installation in Jacksonville, Florida.

Demonstration materials were prepared by FRIS staff. Each vendor received the following data:

1. Map of AU's (Administrative Units) 264, 267, 268, and 271.

2. Documentation of map contents.

3. Tape containing digitized map information.

4. Documentation of digitized tape format and contents.

5. Tape containing Landsat classification data.

6. Documentation of classification tape format and contents.

The requirements for manipulation of these data sets are defined in Table 2.2.2. Each vendor was asked to demonstrate their capability in these nine areas, or to indicate how they would meet these requirements if the capability did not exist. In addition to demonstrating their systems capabilities, vendors were asked to provide a firm bid for installation of the System in Jacksonville.

During the final System Design Committee meeting in Dallas, Texas in early December 1979, vendors capabilities were evaluated. The committee was primarily concerned with the vendors capability of meeting the FRIS system requirements. Bid information was used by St. Regis staff to prepare financial evaluation for St. Regis management.

2.2 Image Processing Transfer

The core of the FRIS image processing systems consists of modifications to the LARSYS software package. LARSYS is a well documented system designed to process digital multispectral scanner data. The system currently operates on an IBM 370/148 in a virtual machine environment. The software transferred to St. Regis did not include the entire LARSYS package. Furthermore, it operates on an IBM 3033, or the equivalent, and job initiation is through remote job entry stations.

Table 2.2.1 Functional Specifications for Evaluation of FRIS System Design Alternatives

- I. Graphic Data Capability
  - A. Input
  - B. Analysis
  - C. Update
  - D. Output
- II. Tabular Data Capability
  - A. Input
  - B. Analysis
  - C. Update
  - D. Output
- III. Image Data Capability
  - A. Conversion from vector to grid
  - B. Conversion from grid to vector
- IV. Other
  - A. Hardware
    - 1. Configuration
    - 2. Deliverability
    - 3. Support
    - 4. Data Communications
  - B. Software
    - 1. Availability/cost of source
    - 2. Support
    - 3. Transportability
  - C. Implementation
    - 1. Cost
    - 2. Time
  - D. Vendor Profile
    - 1. Customer base
    - 2. Customer Service
    - 3. Expertise in forest based applications
    - 4. Vendor stability
- V. Overall Cost

Table 2.2.2 FRIS data base manipulation requirements

1. Produce a plot of the digitized data, containing the AU (Administrative Unit) and OA (Operating Area) boundaries for all four of the AU's.

2. The fourth file of the tape contains some extraneous points, produce a clean plot demonstrating the editting capabilities.

3. Convert the Landsat classification data from grid to vector format.

4. Produce a plot of each layer of information
   a. AU boundaries
   b. OA boundaries
   c. Landsat classification

5. Associate attribute data with each layer of information
   a. for the AU boundaries layer, the attributes would consist of the AU numbers (264, 267, 268, and 271).
   b. for the OA boundaries layer, the attributes of interest would be the OA numbers, the forest type, and the age of the stand (this information may be found on the sheets describing each individual AU).
   c. for the classification data, this would be the names of the classes taken from the classification results tape.

6. Produce an overlay of the three layers of information.

7. Graphically represent where the Landsat classification and the map are in disagreement for a cover type. What we have in mind is a map depicting areas that would satisfy such Boolean combinations as: NONSTOCK (from the Landsat classification) .AND. .NOT. (forest types 9 .OR. 92 (from the map)).

8. It would also be desirous to have maps of areas based upon the attributes of the OA's (e.g., Forest types 2, 11, and 21 which are greater than 15 years old).

9. Demonstrate the capability to apply transformations to the vector data sets (e.g., for rotation and scale).

The major requirements of this task were the modification of the existing software to run on a batch machine. St. Regis staff were responsible for this implementation, and LARS personnel provided guidance and consultation when needed. A list of functions and subroutines that were transferred are included as Appendix A of this report.

LARSYS version 3.1 as it currently exists at Purdue University's Laboratory for Applications of Remote Sensing (LARS), consists of 41 processing functions contained in 377 FORTRAN routines and 49 IBM ASSEMBLER routines. LARSYS is not just an integrated set of computer programs designed for the analysis of remote sensing data. It is an entire approach to the conversion of remote sensing data into information useful for monitoring and inventorying earth resources. Results of the Demonstration Phase of FRIS document the utility of the approach to industrial forest management in the southeast.

The System Transfer Phase of FRIS therefore not only dealt with the implementation of the LARSYS software, but also the transfer of the concept. The FRIS image processing subsystem is comprised of a subset of the LARSYS version 3.1 software package which are currently available through COSMIC. In addition select developmental and experimental routines, some of which were developed specifically for FRIS, have also been transferred.

As part of this transfer activity, source tapes, program listings, users' manuals, system manual, control card references, and program abstracts were provided to St. Regis for 23 image processing processors. The following is a brief description of these processors:

PICTUREPRINT - histograms and displays data in picture form on a line printer for each channel selected.

CLUSTER - using reflectance values from selected channels, groups data into classes and displays the results on a line printer.

STATISTICS - calculates transformed divergence between all class pairs and performs these calculations for every set of channels requested.

CLASSIFYPOINTS - assigns each pixel in the data to a class, using either the maximum likelihood algorithm or minimum distance rule. The results are written to tape or disk.

PRINTRESULTS - using the classification results located on tape or disk, prints a map and tabulates the number of pixels classified into each class.

IDPRINT - prints most of the information contained in the MSS data header record.

DUPLICATERUN - duplicates a data run from tape to tape, and optionally allows arithmetic expressions to be applied to the data.

COPYRESULTS - copies classification results from disk or tape to another tape.

LISTRESULTS - prints information located in the header records of the classification results.

PUNCHSTATISTICS - punches a copy of the statistics deck located on a classification results tape.

LINEGRAPH - graphs a line of MSS data on a line printer.

COLUMNGRAPH - graphs a column of MSS data on a line printer.

HISTOGRAM - histograms data and produces a deck of the histogram information.

GRAPHHISTOGRAM - on a line printer, displays the histogram produced by PICTUREPRINT or HISTOGRAM processors.

SECHO - extracts and classifies homogeneous objects as if they were single pixels.

MERGESTATISTICS - combines more than one statistics deck into a single deck.

RATIOMEANS - using the mean vectors of classes in a statistics deck, calculates and prints the ratio of the values for the specified channels and the sum for each class.

BIPLOT - produces a bispectral plot of classes contained in a statistics deck.

COMPARERESULTS - compares two classifications results over the same area for purposes of identifying changes.

The above processors represent approximately 42,000 lines of FORTRAN, 5,000 lines of Assembler, and 1,500 lines of CMS (Cambridge Management System) EXEC language. The programs were transferred in card image format on 9-track computer compatible tapers. Copies of tape listings and user documentation were also provided.

The task faced by St. Regis personnel was to convert the software which ran on an IBM 3031 operation under VM to an IBM 3033 operating under MVS. That is the LARS computer operates as a virtual machine while the St. Regis computer operates as a batch machine. This meant that the LARSYS version 3.1 programs were not directly compatible between the LARS and St. Regis IBM machines. The following changes were required in order for the software to be compatible on the NCC machine:

A. Add the function COPYTAP, this allows the data to be read from tape to disk and stored on disk. St. Regis has a disk based system while LARS is tape based.

B. Replace command language with ROSCOE. Due to the operation system differences between the two machines, the command language had to be modified. ROSCOE is a software package that permits St. Regis users to initiate jobs from remote sites. This will replace the CMS software which is currently used in LARSYS to perform similar functions.

C. LARSYS contains some bookkeeping routines that were deleted because these functions were already handled by St. Regis.

D. All non-standard file handling routines in LARSYS were replaced to meet St. Regis computer software conventions.

E. All tape handling routines were modified to deal with disks.

F. Machine dependent assembler language routines were eliminated where feasible.

In order to implement the software at NCC, St. Regis staff had to accomplish the following tasks:

1. Compile programs from tape.

2. Create disk files.

3. Modify software for compatibility to St. Regis machine, including elimination of bookkeeping, assembler routines and modification to tape callable routines.

4. Create ROSCOE modules.

5. Develop links to GIS.

6. Develop St. Regis/LARS user documentation.

LARS staff were available for consultation and debugging when needed. Our experience during the software implementation was that very little assistance was requested by St. Regis personnel. Implementation of these software progressed with very few problems. This was most likely due to; a) the level of documentation provided with the LARSYS software, and b) the knowledge of the staff involved with the implementation.

2.2.1 Programming Additions

The LARSYS software packages were originally designed to process digital multispectral scanner data in a research environment. Periodically, modifications and embellishments have been added to LARSYS version 3.1 support packages to improve interaction with the human component of the analysis activity. Since FRIS is a user oriented, operational system there were certain additions to the LARSYS version 3.1 software since the midpoint of Phase II. The two newest additions reported this quarter are significant because they directly affect FRIS requirements. The two new program additions are SMOOTHRESULTS and COMPARERESULTS.

SMOOTHRESULTS is a post classification processor designed to emulate the human action of creating a mapping cell. Mapping cells are the basic component of timber type or operating area maps. The theory behind the mapping cell is simply that areas less than a minimum size, say five acres, are ignored for map generation purposes and included as part of a larger population. Therefore, a two or three acre inclusion in a type would be ignored when the map is created.

The human quickly handles these small inclusions when making a type map. A Landsat classification, however, will display most inclusions that fall within the scanner resolution. These will result in a salt and pepper effect on classification output. A situation that may accurately protray the cover composition but which is often not appealing to land managers who are used to working with "clean" (no salt and pepper) maps.

SMOOTHRESULTS allows the analyst to define a mapping unit and produce a classification results map which does not exhibit a salt and pepper pattern. The processor scans a LARSYS Classification Results File and replaces groups of classified points (cells) with the dominant class from that group. The analyst has the option to specify the size of the cell (CELLSIZE card), class numbers which are to be replaced (PRIORITY card) and weighting factors for each class (WEIGHTS card). The output from this function is to tape to disk in LARSYS Classification Results File format. Figure 2.2.1 is an example of a classification result which shows output both before and after use of the SMOOTHRESULTS processor.

An additional option to SMOOTHRESULTS allows the analyst to define new classes which are mixtures of old classes was developed and transferred to St. Regis. The control card reference file and program abstracts are included in Appendix B.

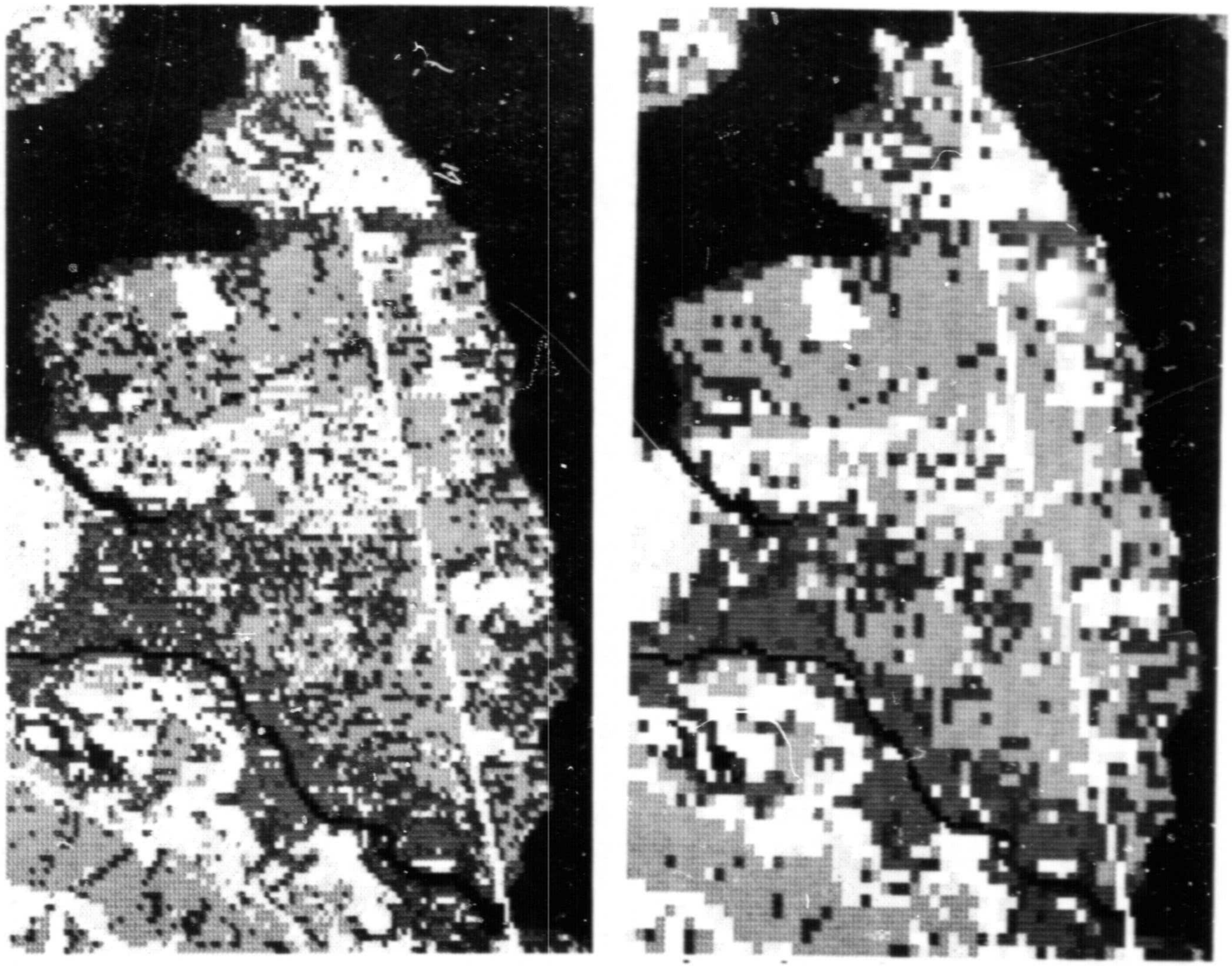


Figure 2.2.1 Example of classification output both before and after SMOOTHRESULTS processor implementation.

The other processor that was upgraded for addition to the FRIS package of software and transferred to St. Regis was COMPARERESULTS. COMPARERESULTS is a post classification processor, and is designed to make comparisons between classification results. This processor is intended to be used to compare two anniversary Landsat classifications which have similar class structures. The resulting product of this comparison is a LARSYS results tape containing "change" classes.

Change classes are designated by the analyst and are in the form where:

Pine (time 1) goes to Non-Pine (time 2), and

Non-stocked (time 1) goes to Stocked (time 2).

Optimally, Landsat data is of an anniversary nature, that is the data of collection for both dates is nearly coincident but chronologically a year or more apart in time. Present requirements of the COMPARERESULTS program are that the Landsat scenes be precision registered. Independent classifications are generated for time 1 and time 2. The analyst is careful to insure that class structure, that is the various spectral groups that comprise the information classes is similar. Once the classifications have been generated, COMPARERESULTS is run and an output similar to figure 3.3.3 is produced. Tabular information which indicates the amount of change in acres percent of area by class can also be produced. Program abstracts for COMPARERESULTS appear in Appendix B.

2.3 Preprocessing Transfer

This task involved transfer of the "front-end" software that is necessary to prepare the Landsat data for classification. A significant expenditure of effort was required for this task because of the complexity of the software and its level of documentation. Initially, a software definition or planning activity was required to define the specific components to be transferred to NCC.

Software to handle the Landsat data formats, including the new P tape format, had been defined, programmed and transferred to St. Regis. An evaluation of the Landsat 3 data was made to define the extent which other processors, designed to accomplish more precise scene registration, should be transferred. The other part of the preprocessing software that was transferred included geometric correction software.

Other major activities included under this task involved assisting in the development of a FRIS map coordinate system and defining the form and operations of a remote reformatting capability.

LARSYS preprocessing software development task resulted from a number of FRIS system design meetings which began in July of 1978. As of July 1979, the FRIS system design had progressed to a point that the LARSYS preprocessing and analysis software to be transferred to the St. Regis has been determined. LARSYS preprocessing software consists of three major processors. The three processors convert digital Landsat data to LARSYS format, perform systematic geometric corrections of Landsat data and register two images of Landsat data. The requirement to systematically register a Landsat scene to a map or another Landsat scene, the image registration capability, was subsequently modified. The original requirement included implementing the image registration on the host main frame in Dallas, Texas and controlling job execution from the mini computer at Jacksonville, Florida. The selection by St. Regis, of a sophisticated software package to reside on the mini at Jacksonville, Florida, eliminated the need for the LARS image registration software. The registration capability on the mini is more efficient, and therefore, more effective than the research software that was tentatively planned for implementation. A discussion of the planned implementation activity for this software module is included for information. The final task beyond the transfer of software was the documentation of the software.

2.3.1 Reformatting

The first preprocessing system of programs converts digital Landsat data to a format compatible to LARSFRIS. The functional specifications for this processor required the conversion of input EDIPS Landsat MSS data, including ne "p" format data, received in a band interleaved by line (BIL) format to LARSYS format. "P" format refers to EDIPS format computer compatible tape data requested as CCT-PM or fully processed MSS data with geometric corrections applied and resampled to a map projection. Details of this format may be found in the "Manual on Characteristics of Landsat Computer-Compatible Tapes" published by the EROS Data Center in December 1978.

Preliminary work on the design specifications to incorporate the "P" format Landsat data processor began in early March 1979. In particular, the LARS reformatting group determined that a comprehensive design phase would substantially shorten implementation during the programming, debugging, and documentation phases. Work on the design lasted into early July. Every algorithm was defined, program modules specified, and nearly all substantive variable and buffer areas were identified before programming began. The main routine, along with all subroutines and calling sequences, were thus determined and documented. The program design included accomodation for function specifications which would support both nearly automatic operations in an operational environment as well as multiple options required in a scientific research environment.

Several techniques were utilized to solve this problem. First, the basic approach was top-down structure programming. All routines have a top to bottom flow of control, and top of calling sequences modules were programmed first. As much testing as possible was done after completion of each module and assembly of it with previously completed modules higher in the calling sequence. The second technique was to place a unique or substantive process in a separate module. Modules were allocated based on the structured "English" version of the processing algorithm (Appendix C). The question asked by the analyst as he scans this "structured English" program would be what processes must occur for this "sentence" or "group of sentences" to successfully execute. The answer defined the modules to be programmed. Third, the primary programming language was FORTRAN IV utilizing the IBM Level G or H compilers. Some IBM Assembly Language program includes the structured "English" versions of the algorithms used.

In addition, the implementation of the entire EDIPS to LARSYS programming task was PERT charted. This aided in the management of time and resources for the project. Parallel programming efforts could then be spotted as well as known or potential bottlenecks. Finally, to facilitate the control of the program in the most humanly efficient default mode, only three cards are required to execute the EDIPS processor.

2.3.2 Geometric Correction

The geometric correction processor was the second major system of programs. The geometric correction processors original functional specifications called for maximum correction of geometric distortions of Landsat I data with minimum use of resources. The most important distortions thus were corrected. In particular, the data is assumed to consist of square 80 meter pixels which are rotated to true north, deskewed for the earth's rotation and rescaled for output on a line printer with 8 x 10 aspect. In the context of FRIS pre-EDIPS format Landsat data may be corrected for geometric distortions.

In the current FRIS image preprocessing system, this program may be utilized to rotate Landsat 3 data to true north and rescale it if necessary. This is especially important considering the number of data sources already in true north orientation. Examples are the St. Regis Administrative Unit maps. Data in the same orientation is far easier to use for the human than data skewed or rotated relative to a given true north reference data set as the forest AU mentioned previously. For example, checkpoints are more readily defined and located as part of the image registration process. Relatively minor updates to the control card reader to incorporate the rotation-only parameter were required to bring this program to a transferable status. Inspection and update of program listings, program abstracts, and user documentation were also required.

2.3.3 Image Registration

The last major processor contemplated was the image registration system. The primary purpose of this system was to register two coincident digital images such as two Landsat digital image data sets. The secondary purpose was to provide for the registration of any known two-dimensional grid to another known or defined two-dimensional grid. An example is the registration of Landsat data to a U.S. Geological Survey standard quadrangle map. The former has a grid X-Y of pixel locations while the latter has a grid of inches and meters both horizontally and vertically. Input images are assumed to be in LARSFRIS format. Functional specifications for the image registration system are given in Appendix D. The information in the appendix and the discussion which follows is provided only as information since this software was not implemented. Operationally, image registration is accomplished by the data base system acquired by St. Regis.

The image registration system is a composite of research software developed at LARS and consists of three functional sections: 1) the main image registration section, 2) the coincident image cross-correlation section, and 3) the multifit least squares analysis. While such software exists, it was never intended to be operational. A revised processor was desirable to achieve a more modern supportable software system. The writers of the old system are no longer available and documentation for the program was sparse. The procedure followed in this image registration system programming task followed that of the EDIP3 processor mentioned previously. Once the need for the new processor was established, functional specifications were determined. The overall goal was to produce a maintainable system which is modularized, as well as documented for program contents, programming techniques and user documentation. Furthermore, the latest obtainable registration techniques have been used for implementation. No attempt was made to duplicate the Goddard MDP (Master Data Processor). The function of the system, however, is similar. Two standard registration procedures were utilized to allow more accurate, cost efficient registrations. These activities occured prior to acquisition of the St. Regis data base software.

The two implementation procedures were cubic polynomial for the overall registration blocking with linear interpolation. The first refers to the cubic polynomial whose coefficients are derived from the MULTIFIT processor. This processor uses least squares analysis to derive the best affine, bi-quadratic or bi-cubic fits for the checkpoints taken from the respective digital images or known grids as appropriate. With the best equation fit determined, normally a bi-cubic one, the blocking concept is utilized to reduce computation time.

The concept of blocking during digital image registration is a moderately complex one. First, the bi-cubic polynomial for image location is investigated for rates of change and saddle points by solving the first and second derivatives. Utilizing these values one may determine the minimum block size within which a bilinear function accurately approximates the bi-cubic one. Block size may be though of as Y lines by Z columns. At least "Z" number of multiple times are eliminated from the calculation of each pixel location within the block. Only the corner pixel locations of each block need to be calculated in full bi-cubic polynomial mode. The linear interpolation within the block is relatively fast and predictable with far fewer calculations. Should the bi-quadratic polynomial be the best fit for the data, blocking may still be used. However, the reduction in the location calculation time will not be as great. In the unlikely event that a linear fit will suffice, blocking is not used.

Other features of the registration system include an automated cross correlation processor and two forms of pixel gray level interpolation. First the automated cross-correlation processor is an aid for acquiring checkpoint locations which are selected from two coincident Landsat digital image data sets. This cross-correlation will be accomplished by the implementation of a numberical integration image correlator. Control of where checkpoints are sought may be by line and column intervals and starting and stopping locations. Alternate control may be by a set of arbitrary checkpoints for location after cross-correlation. An appropriate initial transformation will accompany either control method. Should this concept not be practical because of data dependency problems, manual checkpointing methods will be used.

The second feature consists of a gray level interpolation method. A gray level must be determined for each pixel location in the output grid. The nearest neighbor is the default. The advantage of nearest neighbor interpolation is that no new data values are created. Classification algorithms may use the same statistics before and after registration. Cubic interpolation of pixel gray levels is the alternative. This cubic interpolation algorithm assumes surrounding pixels input to the respective "center" pixel's gray level. The "center" pixel refers to the calculated subpixel location outputed from the registration polynomial. The pixel location is theoretically subpixel and the level of each surrounding pixel to the "center" pixel is determined by which of the sixteen subpixel locations is calculated for the "center" pixel. To facilitate the implementation of this third order Lagrange interpolation, "center" pixels locations are calculated to one quarter of a pixel. Coefficients are pre-supplied in a table for each of the sixteen possible "center" pixel locations. The level of calculation is thus restricted to simple addition and multiplication. Cubic interpolation of gray levels smooths the visual look of images. This approach has the potential for portraying slightly more accurate subpixel locations for given features of the scene. The cubic interpolation algorithm is described in Appendix E. Compared to the nearest neighbor interpolation technique, the cubic convolution approach requires more computer-resources.

2.3.4 Preprocessing Documentation

Documentation is a key to the technology transfer of the LARS image processing/analysis system totally known as LARSYS. Good documentation although expensive, was necessary to inform the programmer and user. The programs will be more maintainable by less readily knowledgeable programming professionals. Over the long term this potentially means less total time and expense. To the creator of the documentation, the effort means a more thorough knowledge of just what he or she has transferred to a fellow programmer in another organization.

Documentation was the last major effort of the LARSFRIS preprocessing software implementation. Documentation consists of three main efforts for each of the three processors previously described. The three types of documentation were: 1) program listing documentation, 2) program module abstracts, and 3) user documentation. The first form of documentation was guided by a standards document (Appendix F) produced by the reformatting group at LARS. These standards expand and clear up details of program listing documentation to be followed in the preprocessing software. Inputs, outputs, and major variables and arrays are detailed at the top of each program listing under this standard. In addition, processing procedures are clearly explained through comments in the listing. In essence, a new programmer should be able to read the comments within the listing and know what algorithm the code is implementing.

Programming abstracts are the second form of documentation. These follow the LARSYS standard manual. This form of documentation normally will be used with the program listing for maximum communication to the programmer.

Finally, user documentation was generated. The user document describes what a processor is used for as well as how to use it. Sample control card sets are included along with explanations of what each set does. User documentation was designed to address how the program is run. Detailed information on aglorithm implementation and function are not included in this documentation.

2.4 Landsat 3 Evaluations

Prior to the launch of Landsat 3 in March 1978, NASA announced their intention to upgrade the ground handling capability of the CCT data. Two elements of the announced change that were thought to have a significant and positive impact on data users were:

1) Improved data order turnaround, and
2) Geometrically corrected and ground registered CCT data.

Although order turnaround of data by the EROS Data Center (EDC) is not a critical aspect to most forestry applications, it was important to the successful operation of FRIS. Order turnaround, that is the elapsed time between the date of data collection and the date the user receives the CCT, was important to St. Regis if current Landsat data was to make a real contribution to the company's ongoing forest updating system.

In order for Landsat data to be useful in FRIS, the data must be collected between the months of October and February. Not only must the data be collected during this time, but it must also be available to the system if the annual updating cycle is to be maintained. Availability to the system means that St. Regis will have; ordered, received, processed, and classified the data, so that these results can be reviewed when land managers review their annual updates.

The key to meeting this time requirement rests with receipt of the CCT data from EDC. Historically, order turnaround from EDC was never better than 21-day and often order receipts could take upwards to 60-days. The announcement of a 10-day order turnaround time from EDC would help to insure the success of an operational FRIS. The improved order efficiency would at least make data available to the system faster and therefore help eliminate a bottleneck that was non-FRIS dependent.

The second element of the new CCT format, the geometrically corrected ground registered scene, could also be a benefit to FRIS. The new Landsat 3 data was provided by EDC to the user in a geometrically corrected by non-rotated format. The availability of geometrically corrected data has the potential to save the user both time and computer resources since these steps may be eliminated from the preprocessing sequence. However, this data is not rotated and therefore not corrected for north orientation. Since one of the uses of the Landsat data in FRIS will be to provide updated maps, the rotation of the data is an important consideration.

Conceptually, the image rotation problem could be handled as a post-classification process in the FRIS mini-computer. This approach would involve classifying the data as received from EDC and then converting the classified data from an image grid (an image raster) to a set of classified vectors. The classified vecotrs would then be rotated using the appropriate transformation and overlaid and registered to the St. Regis ownership boundaries.

Using the approach all the preprocessing activities with the exception of Landsat data tape reformatting would be eliminated. Only the reformatting, image processing and possibly the raster to vector conversion would be performed on the mainframe. The remaining activities could be accomplished on a mini-computer with suitable geo-referenceing software. Savings would occur primarily in the reduction of time necessary to prepare the data. Two important assumptions are necessary to enable this approach to work:

1) EDC will operationally be capabile of providing geometrically corrected data, and
2) The Landsat rotation and overlay can be suitable performed on a mini-computer.

During Phase III one Landsat 3 fully corrected data set was ordered. This data, designated as P-format, was available over the Picayune test site in Mississippi. Data turnaround by EDC was within the specified announced time of 14-days. This acquisition proved the EDC was able to meet announced delivery dates. However, the test was not repeated so we have no way of knowing if this capability is operational.

After receipt of the P-tape from EDC, a quick evaluation was conducted to determine if this data would eliminate the front-end preprocessing currently required prior to image classification. Another important preprocessing transfer activity involves the future potential use of fully corrected, P-format, data available from EDC. The availability of P-tape data to FRIS will eliminate much of the front-end preprocessing currently required prior to image classification. The discussion that follows gives preliminary results on the use of fully corrected Landsat 3 data from the Picayune test site in southern Mississippi, figure 2.3.1.1.

The fully geometrically corrected Landsat MSS frames acquired for the forest resource data base are placed in a specific projection and orientation. This makes possible a one-to-one correspondence between earth coordinates and row column pixel locations in the data. Having such a relationship for each frame will enable resource polygons on maps to be automatically related to row column locations in the data. Visual searching in the imagery would then be unnecessary once corner latitude, longitude, or UTM coordinates were known. A program was developed to enable user conversion of coordinates and some of the details are included here.

The fully corrected MSS data are placed in a Hotine Oblique Mercator (HOM) projection and in the future they will be placed in the Space Oblique Mercator (SOM) projection. These projections are discussed in Appendix D of the new Landsat User's Handbook. The scale distortions of these projections is very small (1:10,000); thus a linear transformation can accurately

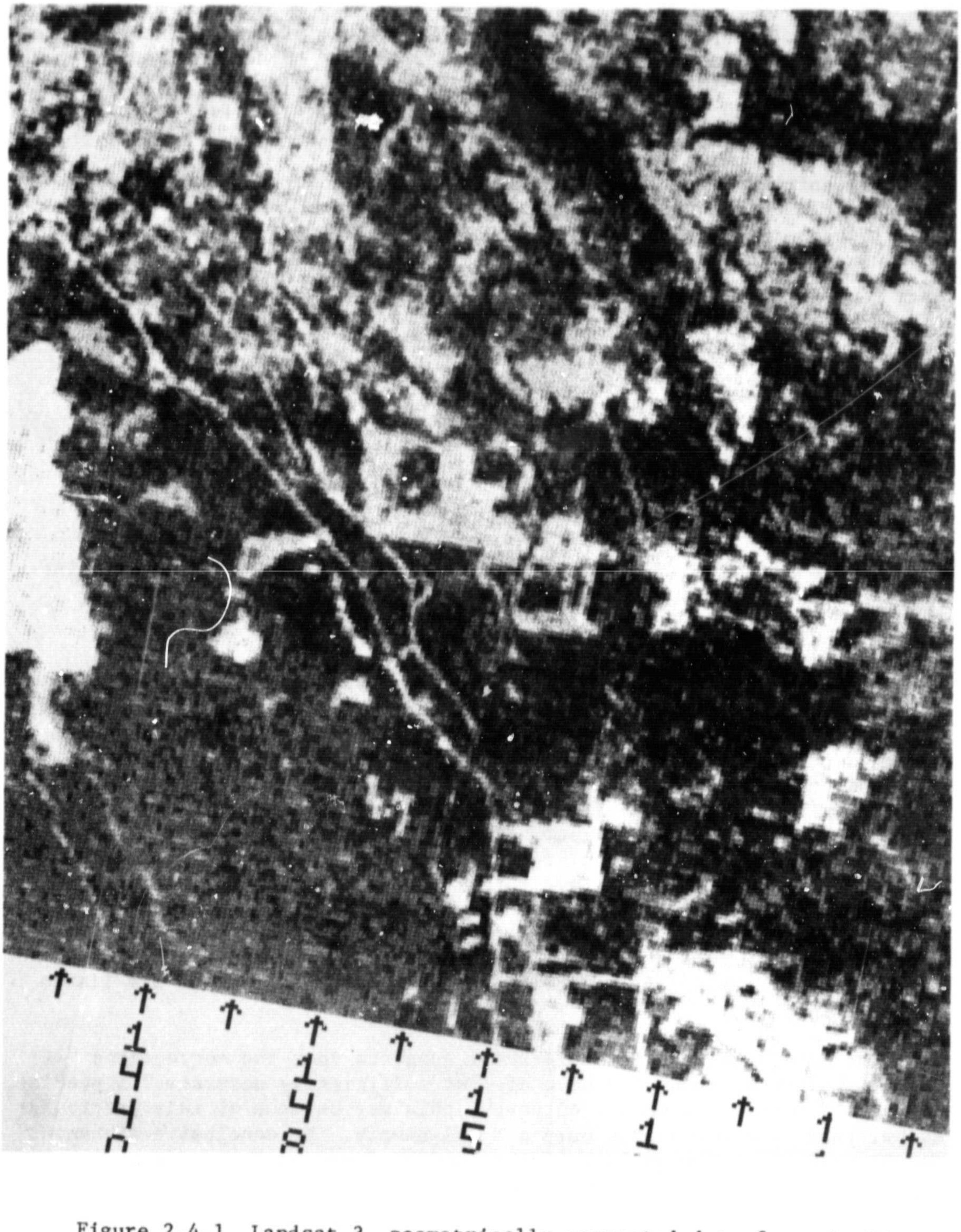


Figure 2.4.1 Landsat 3, geometrically corrected data from the Picayune, Mississippi test site has been overlaid with photographically reduced ownership maps to indicate the visual correlation of this new data type.

be used to relate points in the frame. The earth is divided into zones of latitude and within each zone the corrected frames have a constant azimuth. The zone covering the areas of interest here is zone 2 with latitude range 23° N to 48° N and the zone azimuth is 14.3394993°. The pixel scale of the fully corrected data is 57 meters in both directions.

The software employed utilized a latitude-longitude to Universal Transverse Mercator conversion program to transform user input latitude-longitude coordinates first to UTM. Then a linear conversion was made to line column using the expressions:

LINE = CLINE + DLINE

COL = CCOL + DCOL

DLINE = (-DELEAS . SIN(ALPHA) - DELNOR . COS(ALPHA))/57

DCOL = (DELEAS . COS(ALPHA) - DELNOR . SIN(ALPHA))/57

DELEAS = EAST - CEAST

DELNOR = NOR - CNOR

where: CLINE, CCOL are the center line and column of the frame.

CEAST, CNOR is the UTM easting and northing of the center point.

EAST, NOR are the UTM easting and northing of the point to be transformed to LINE, COLUMN.

The conversion program (LOCPNT) was developed for interactive terminal use and required typing in the frame center latitude and longitude; then the user enters any number of latitude-longitude points in the frame he wants to convert. Problems were encountered in testing the program on the Picayune frame. Four test points were taken from the Nicholson and Dead Tiger Creek quadrangles in the Picayune frame and the latitude-longitude coordinates were input and the output line and column were observed. The input parameters are a part of the problem. A latitude and longitude are given as the frame format center; however, it was uncertain what exact line-and-column number corresponded to this. The bias observed at one of the test points was removed and the resulting center line column was taken as the format center. Thus, there is no error at this point. At the other three points, errors were observed. The results are listed in Table 2.4.1.

The data presented in Table 2.4.1 suggests that the corrections supplied with the Landsat 3 data are not sufficiently accurate for precise registration to the ground. Although, this may be true of this particular scene, this data represents only a small sample. No conclusive statement can be made about the quality of the Landsat 3 correction system.

Table 2.4.1 Coordinate Conversation Tests for Picayune Frame. Format Center: 30.18°N., 89.52°W. Center Line, Col: 1518,1796.

Test Point				Estimated Point		Error	
Lat.	Long.	Line	Col.	Line	Col.	Line	Col.
30.375	89.625	1189	1518	1189	1518	0	0
30.5	89.75	987	1265	999	1285	12	-7
30.375	89.5	1150	1725	1141	1724	-9	-1
30.5	89.625	948	1473	952	1463	4	-10

However, the very existance of the geometrically corrected Landsat data sets are a benefit to the FRIS system. If this data can provide no more than a rudimentary correction that software within the data base can provide the final registration to the ground. The resultant registration should meet the precision requirements of St. Regis and incur a time savings over a system that would have to start with uncorrected raw data.

2.5 Technology Transfer Task

Probably the most important component of the entire System Transfer Phase involved transferring the computer-aided analysis capability to St. Regis staff. The basic elements of the system that were transferred included; hardware, software, and analyst capabilities. Hardware was assumed to be available or easily acquired. Software was modified or developed, and analysts were trained. Since the system is composed of the sum of its parts, all parts must be complete or a viable system could not be achieved. Hardware and software acquisition and modification were relatively straight forward and easily attained. Preparation of a core of knowledgable analysts within the company was a somewhat more complicated task. The major emphasis of this task was to insure that a useable capability was transferred to St. Regis staff.

The foundation for the Phase III Technology Transfer activities were the results from the Phase II Demonstration. The decision by St. Regis staff to implement the LARSYS software defined the type of training that St. Regis staff would be given.

This task focused on educating St. Regis personnel regarding specific classification procedures associated with the use of Landsat MSS data. The success of the technology transfer activity was paramount to the development of an independent remote sensing capability by St. Regis. This was an important goal defined at the outset of the Project. Various activities included:

- o Support of a remote terminal hook-up between St. Regis at Jacksonville, Florida and LARS

- Support of a remote ROSCOE terminal between the St. Regis National Computer Center (NCC) at Dallas, Texas and LARS

- Development of User Documentation

- System Transfer consultation

- On-site machine processing workshops

2.5.1 Remote Terminal - LARS/JAX

The remote terminal was the central theme around which all St. Regis in-house training was developed. Basic background about remote sensing was transferred to St. Regis personnel through various mechanisms. However, these devices were not designed to provide hands-on analysis experience. These activities were reserved for Phase III.

The remote terminal between LARS and Jacksonville, Florida went online at the beginning of Phase III. The terminal provided a focal point for training workshops to be held in Jacksonville, rather than at LARS. The in-house workshops did more than educate. They also converged the stability of the technology and committment to the concept by management. Thereby they formed the first phase of the orderly transition of the technology from academia to industry. Working together during the beginning of Phase III, LARS and St. Regis personnel developed a training calendar. A preliminary training session was conducted during a week in June, 1979. A more detailed training session, which included "hands-on" training with the remote terminal was offered for one week in July, 1979.

As part of the July training a user handbook was developed, see Appendix G. The handbook was designed as a step-by-step guide to LARSYS. It was set up to support the intermittent terminal user to easily access and conduct a classification session.

2.5.2 Remote Terminal - LARS/NCC

Another terminal interface was employed during Phase III to help facilitate the orderly transfer of software from LARS to NCC. This involved a ROSCOE terminal link to the St. Regis National Computer Center. ROSCOE is a software package designed for interactive editing of batch jobs. This terminal link was intended to allow LARS staff to interact with St. Regis personnel during installation of LARSYS at the St. Regis National Computer Center.

This capability was never extensively exercised because St. Regis staff were easily able to implement LARSYS. Conceptually the NCC terminal link would have saved both time and money if needed.

2.5.3 Other Technology Transfer Activities

A broad variety of technology transfer activities, in addition to training St. Regis personnel and implementing software, were pursued during Phase III. These other related activities are discussed in some detail below.

A key element of the FRIS project plan addressed the dissemination of project related information to the public. The project staff was able to take advantage of various forums to disseminate information about FRIS. These opportunities are discussed in chronological order of occurrence.

- November 1979 - Third Conference of the Economics of Remote Sensing Information Management Systems,, Incline Village, Nevada.

  The conference coordinators made an entire afternoon session available for discussion of the Concept. This meeting was significant because it was the first public hearing of the economic analysis of the project.

- October 1980 - "Remote Sensing for Resource Managers Conference" Kansas City, Missouri.

  A comprehensive set of posters which defined the entire FRIS program was presented to a wide variety of professional resource managers.

- May 1981 - "Conference on Space Technology and Industrial Forest Management" Jacksonville, Florida.

  This conference was hosted by St. Regis, Purdue and NASA specifically to inform forest industry about FRIS.

In addition to presentations made at professional conferences, papers were prepared for inclusion in the proceedings of the Incline Village and Kansas City meeting. No formal proceeding will accompany the Jacksonville meeting, since these results are reported in this and the St. Regis final project reports.

Other published results of the project appeared in the "Congressional Record" (see Exhibit I) and as a project brochure. The FRIS project brochure was prepared to summarize the project goal, needs and describe its implementation and significance. The brochure was destributed at the "Conference on Space Technology and Industrial Forest Management" and is available for general distribution to interested parties.

The last and possibly most significant form of published project related materials consists of software documentation which was prepared for COSMIC. The software documentation, discussed in the previous section, is identified as LARSFRIS. This documentation consists of seven volumes of user manuals, system manuals, and program abstracts, and includes the information necessary for the preparation and processing of multispectral data sets. The documentation along with program listings in card image format on computer tapes will be generally available through COSMIC.

3.0 APPLICATIONS TEST

Two related activities occurred during the System Trnasfer Phase that highlight the benefits of Landsat data to forest management. The first involved the application of COMPARERESULTS software to assist St. Regis personnel assess a new land acquisition. The second involved use of the Landsat reflectance data to estimate crown closure.

3.1 Knabb Tract Analysis

In July 1979 the project staff were given an opportunity to undertake an operational test of the technology. St. Regis had acquired a tract of land in Baker County, Florida (figure 3.1.1). The new acquisition, here after referred to as the Knabb Tract, encompasses approximately 40,000 acres of land and is ecologically similar to the Fargo test site. St. Regis staff were of the opinion that the timber removals had been extensive in recent years. Furthermore, they felt that removals were especially extensive since 1977.

The application test was designed to address the feasibility of using Landsat classified data to:

1) Evaluate the areal extent of the standing timber resource, from 1979 data, and
2) determine the change in standing timber that was detected by Landsat that occurred between 1977 and 1979.

Our goal was to meet these objectives and to provide classification results by 1 November 1979. Timing was an important criteria to this application test because if the data could not be:

Acquired

Preprocessed

Classified, and

Final products available

by the deadline, than the timeliness of the technology would be seriously questioned. In November the window for aerial photographic data collection opens, and this tract was planned to be flown. If we were not able to provide Landsat information by the time the photography is collected and interpreted then the utility of Landsat will be seriously questioned.

Since the Knabb Tract is geographically close to the Fargo Test Site, it is therefore included on the same Landsat scene. The latest Landsat data available to the project was December, 1977. However, inadvertently during reformatting part of the raw data, the part which included the Knabb Tract, was destroyed.

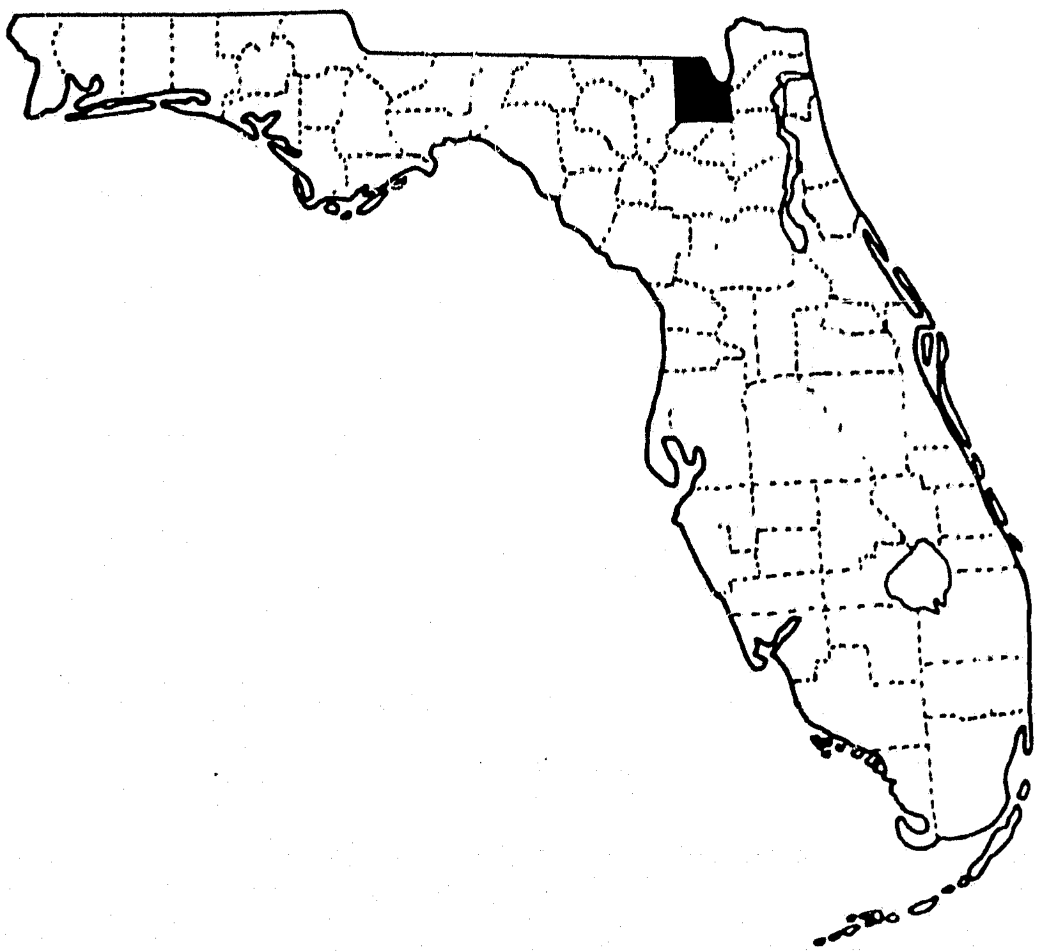


Figure 3.1.1 Location of Baker County, Florida and the Knabb Tract Application Test Site.

A search was requested from the EROS Data Center to identify a suitable set in the early 1979 time frame. Due to a ground system modification, EDC was only able to provide data collected after February 1, 1979. A February 12, 1979 data set was selected for the second date. Both the December 1977 and February 1979 data sets were ordered in early August.

The MSS data ordered in early August was received by mid-August. This rapid turn around provided by EDC was a tight requirement if the deadlines were to be met. The rapid turn around was a pleasant surprise, since data acquisition from EDC had previously been upwards of six months.

The data was in two formats. The December 1977 tape was the old Landsat format. The February 1979 data was in the new Landsat 3 format. This did not pose any problems in preparing the image overlay. The February data was expanded to fit the December data and the combined sets registered to the ownership channel. The only problem encountered with the February data is that it appears to be excessively noisey. Figure 3.1.2 is an example of the December data for Baker County, Florida. Figure 3.1.3 shows the same data set with the ownership boundary channel overlay.

During the latter half of August 1979, personnel from St. Regis Southern Timberlands were at LARS to prepare the ownership boundary channel for the Knabb Tract. Ownership boundaries were digitized, edited, connected, and check points located in the data within a one week time frame. In short, everything necessary to create the final data set up to but not including the data set registrations was completed by the end of August.

The Knabb classifications involved testing the feasibility to extend Fargo training statistics. Supplemental training were added where appropriate. Preliminary classification were field checked before final products were prepared. The classification activity began as soon as data had been reformatted and coarsely corrected and before the final data set was ready for classification.

A detailed discussion of the steps involved in the Knabb Test follows.

3.2 Knabb Tract Data Preprocessing

The primary preprocessing task involved the registration of two Landsat frames to a 1:24,000 scale base map with property boundary information merged with the Landsat imagery. Although one of the Landsat frames (21050-14515) was available in-house, a portion of the frame required for the preprocessing was destroyed during an earlier process, necessitating the reordering of the data. The second data set, Landsat frame 21482-15101 was not expected to arrive until 28 September 1979. Both Landsat frames would have to be reformatted to the LARSYS Version 3.1 format and geometrically corrected (systematic removal of first order distortions) to a scale of 1:24,000 with a line printer aspect (10X:8Y).

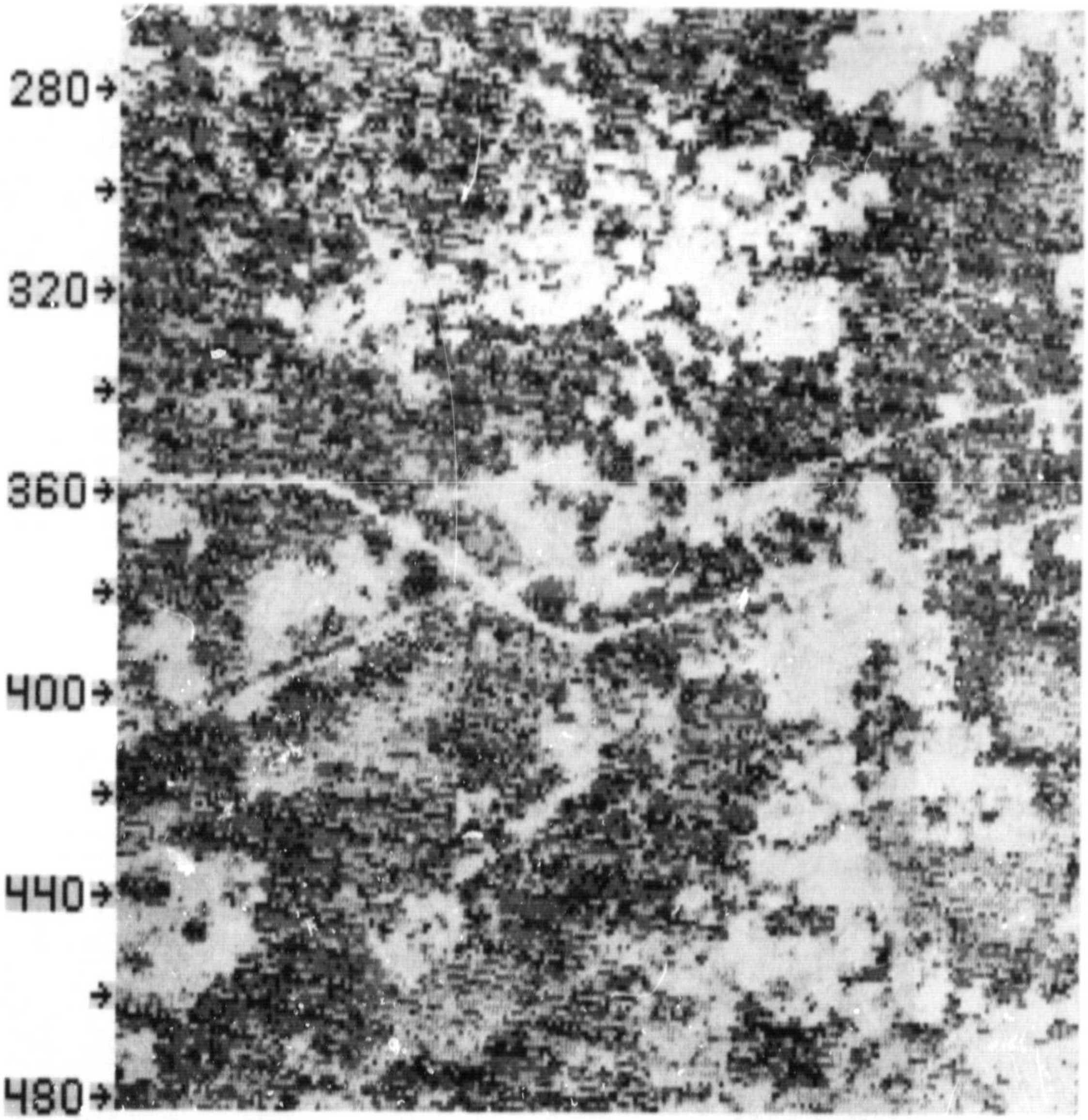


Figure 3.1.2 Electrostatic printer greyscale output from Band 6 of the 1977 Landsat data of a portion of Baker County which includes the Knabb Tract.

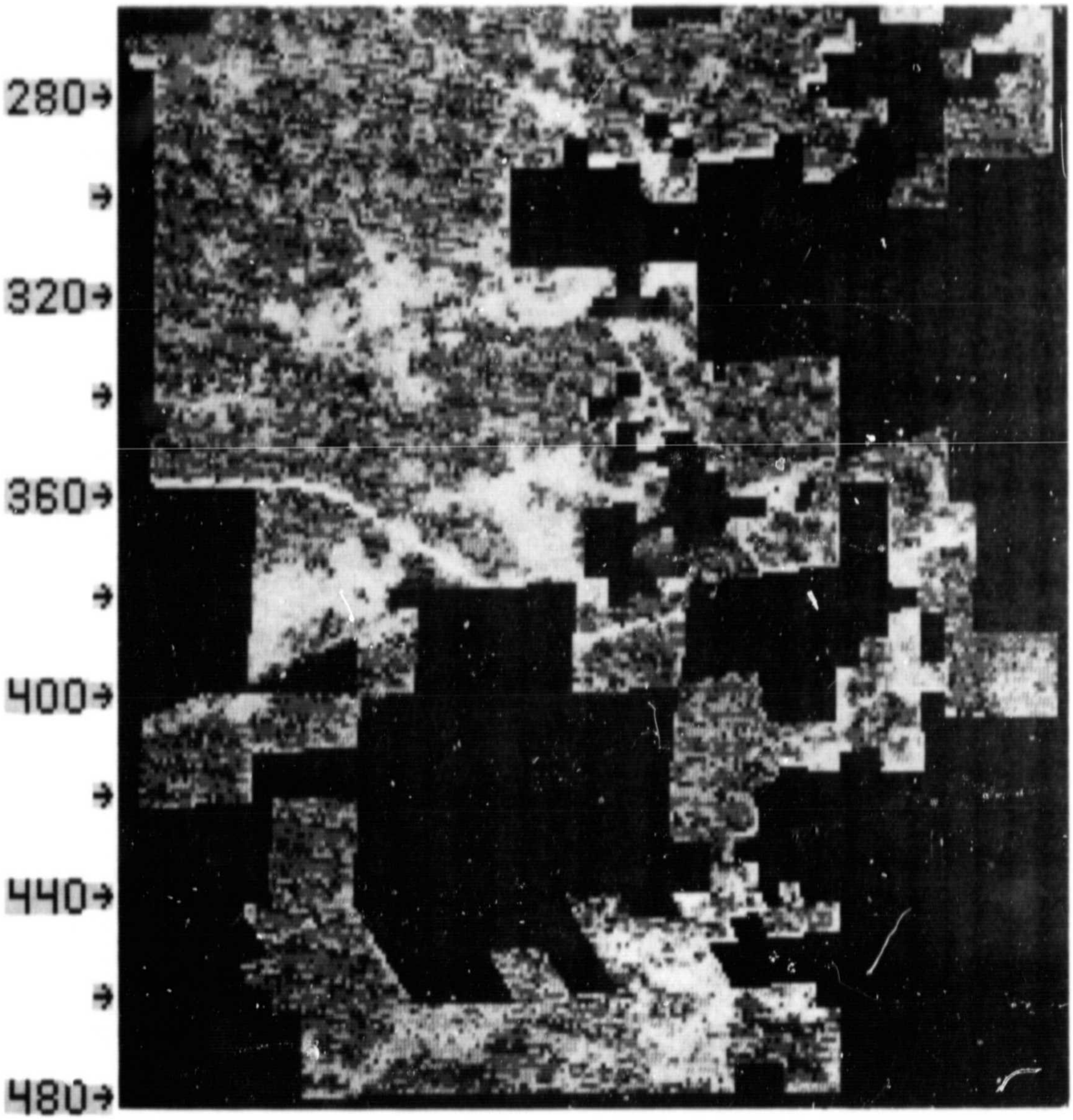


Figure 3.1.3 Data in figure 3.1.2 except that the Knabb Tract ownership boundaries have been included.

Since one of the intents of the test was to determine the timeliness of Landsat in providing land cover information, it was important to complete the preprocessing activity as quickly as possible. Using PERT planning, a probable completion date for reformatting was estimated as 12 October 1979. This date was based upon a starting date of 1 August 1979 and receipt of the February 1979 Landsat data on 28 September 1979. Actual completion of the preprocessing task was 11 October 1979.

Map digitization was performed by St. Regis Southern Timberlands Division personnel. Originally a 1 inch to 1 mile map was to be digitized in the hope of eliminating the editing problem of digitally reconnecting the maps. However, the boundaries to be digitized were drawn on 1:24,000 scale USGS quadrangle maps. A determination was made that accuracy would be lost by transferring the boundaries to the 1 inch to 1 mile map and then rescaling the data back to 1:24,000. A decision was made to digitize the boundaries directly from the USGS 1:24,000 scale maps and join the maps together digitally. This final method worked very well with no unanticipated problems. A portion of the digitized data is shown in Figure 3.2.1.

At the same time the maps were being digitized and the digital boundary information edited the 7 December 1977 data was reformatted to LARSYS format and geometrically corrected. After completing the digitizing, 14 checkpoints were located between the 7 December 1977 data and the USGS quadrangle maps using the LARSYS IMAGEDISPLAY program.

The 14 control points were run through an affine (6 parameter nonconformal) least squares fit. The resulting transformation function exhibited a line error of 0.708 root mean square error (rms) and a column error of 1.032 (rms). The following first order distortions were corrected by the transformation of the systematically corrected 7 December 1977 data to the 1:24,000 USGS map coordinates:

Scale X	1.0152
Scale Y	1.0000
Rotation	0.326 degrees
Skew	0.0299 degrees

At this point, both the digitized map boundaries and the Landsat data were in the same reference coordinate system.

The next preprocessing step was to actually create the ownership information in grid form. This was accomplished by "rasterizing" the vectored digital boundary data. Some editing of an intermediate file is normally required when the boundaries to be rasterized are of regular rectangular polygons. This was the case of the Knabb Tract although minimal editing of the intermediate file was required. The final result was a tape in LARSYS format containing the precision (map) registered data with an auxiliary data channel containing ownership information. All data outside of the ownership was set to a null value (hex 00).

ORIGINAL PAGE IS
OF POOR QUALITY

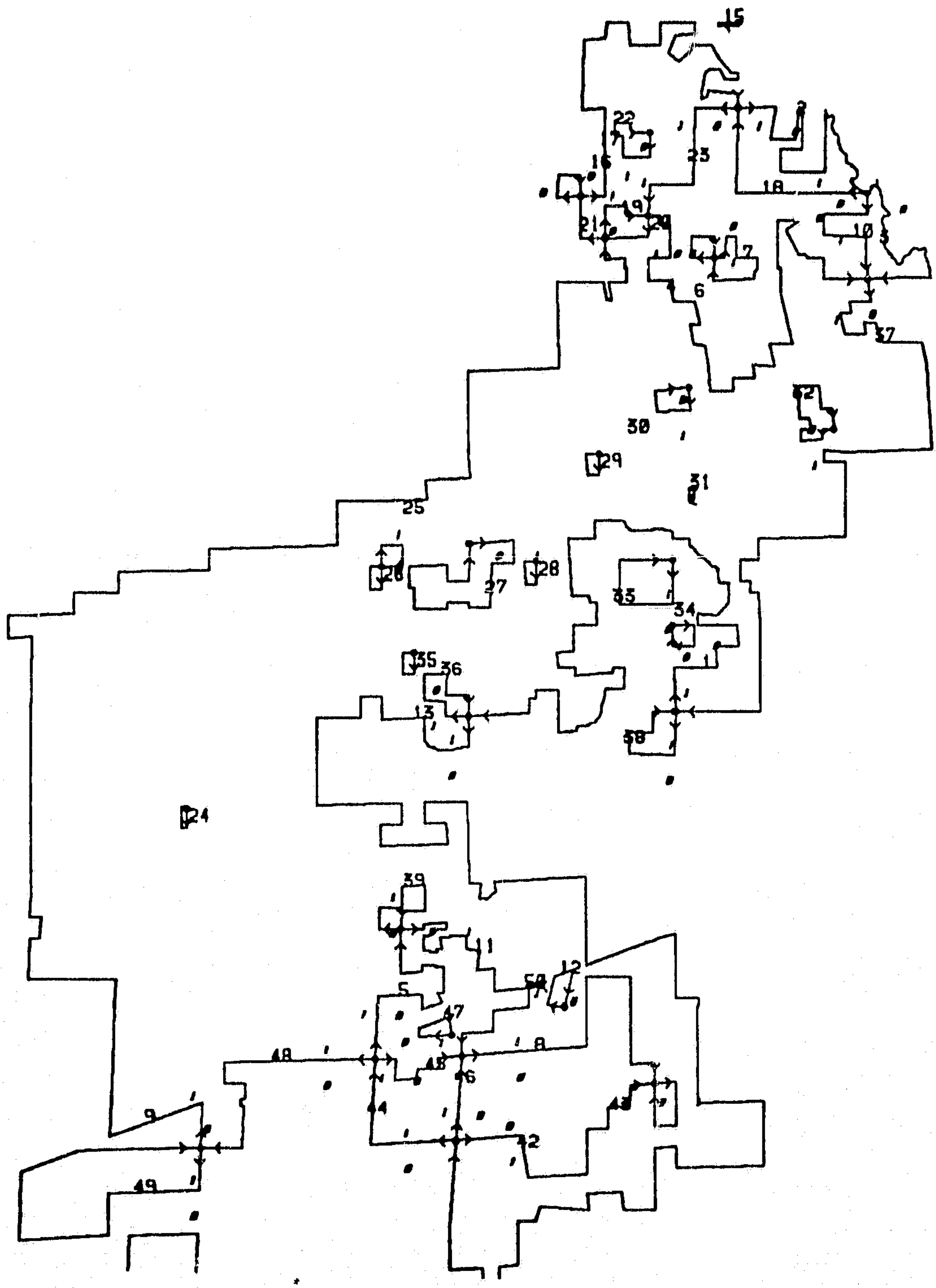


Figure 3.2.1 An example of a portion of the digitized Knabb Tract map data.

The second Landsat data set (21482-15101, 12 February 1979) arrived on September 28. The tape received was created in the newer EDIPS format rather than in the expected X (or older) format. The tape was reformatted using the new EDIPS to LARSYS reformatting software.

Since the second data set was already corrected by NASA to eliminate radiometric and geometric distortions, it was necessary to correct only for rotation, scale, and aspect. The EDIPS corrected tapes are registered to a Mercator projection (either Hotine or Space Oblique) using ground control. The resulting scale of this data set is approximately 1:17,952 with each pixel representing a ground resolution of 57 meters square. Since the current geometric correction processor is designed to correct for pixel size and skew, corrections already applied to the data by the NASA process, a transformation to correct for rotation, image scale, and pixel aspect using the image registration system was developed. The method for performing this type of correction through the image registration system is described in Appendix H.

The final step was to register the corrected second Landsat scene to the December 1977 data. A total of 185 checkpoints were located between the two images using the numerical autocorrelator of the image registration system. An average correlation coefficient of 0.69 was obtained through 270 correlation attempts between the second channel of each scene. The average error between the predicted and observed checkpoint location was 0.67 pixels. The checkpoint pairs were then run through a biquadratic least squares fit. All control points were accepted with rms errors of 0.099 in the line direction and 0.283 in the column direction. The following first order distortions were corrected by registering the corrected EDIPS Landsat scene to the map reference grid:

Scale X	0.9999183
Scale Y	1.0006683
Rotation	-0.1515851 degrees
Skew	0.075 degrees

3.3 Knabb Classification

The acquisition by St. Regis of the Knabb Tract provided an opportunity to extend the classification procedures into an unknown area - one for which no photography or forest cover type information was available to the analyst to aid in defining a classification training set. In order to save time the December 7, 1977 data was classified with the December 30, 1976 training set. Normally this difference in data sets would have caused serious, if not insurmountable, data calibration problems. In this case, however, the dates of data collection were both in December and were very near to the data of minimum sun angle. This, plus the fact that the weather condition were ideal over the training area in 1976 and the Knabb Tract in 1977 allowed the use of the 1976 training set with 1977 data without calibration. The only significant classification problem was found to be the lack of a training class to represent the clear cut/site prepared areas which were present at the Knabb site but did not occur at Fargo. This deficiency was quickly corrected by adding two training classes generated from the Knabb 1977 data to represent these cover type conditions. A summary is shown in Table 3.3.1.

Table 3.3.1 Area statistics for the Knabb Tract calculated from a classification of December 7, 1977 Landsat data.

Cover Type	Acres	Hectares	Percent
Pine	22,723	9,200	52.2
Pine/Hardwood	10,916	4,420	25.0
Slash/Cypress	8,275	3,350	19.0
Nonstocked	1,521	616	3.5
Wet lands	122	49	0.3
	43,557	17,635	100.0

After this initial classification was completed a new data set was received. This data set, collected on February 12, 1979, was overlain onto the 1977 data. Property boundary lines were digitized and added to this data set. A separate analysis was carried out using the previous classification augmented with information gathered during the field checking as training aids. The data quality was not nearly as good as that of the two previous sets. The 0.6-0.7 micrometer band was unusable due to severe banding. The classification was done with the three remaining bands. A summary of the classifications is shown in Table 3.3.2.

In the two-year interval between the two data collections, several areas were cut and planted or were being prepared for planting. The two classifications were compared with the COMPARERESULTS processor to find and identify these areas and those results are shown in Table 3.3.3. The two original classifications are shown in figures 3.3.1 and 3.3.2 and the change map is shown in figure 3.3.3.

Table 3.3.2 Area statistics for the Knabb Tract calculated from a classification of February 12, 1979 Landsat data.

Cover Type	Acres	Hectares	Percent
Pine	25,487	10,319	58.5
Pine/Hardwood	5,972	2,418	13.7
Slash/Cypress	10,959	4,437	25.2
Nonstocked	719	291	1.7
Wet lands	420	170	0.9
	43,557	17,635	100.0

Table 3.3.3 Area statistics for the Knabb Tract showing changes in ground cover which occurred between the two classifications.

Change	Acres	Hectares	Percent
New Plantation	7,686	3,112	17.7
Harvested	484	196	1.1
No change	30,634	12,403	70.3
Unidentified change	4,753	1,924	10.9
	43,557	17,635	100.0

Table 3.3.3 demonstrates the utility of multi-temporal Landsat classifications and a processor like COMPARERESULTS for updating basic forest inventory data. Because detailed ground data was not available when Table 3.3.3 was developed the change classes identified were relatively broad. However, sufficient information was available to identify areas that were in new plantations, or forest lands that were recently harvested. The unidentified change class is most likely composed of areas being site prepared, burn areas, excessively wet areas, or any area whose spectral composition was markedly different between the two dates.

The ability to classify and compare anniversary Landsat data introduces new capabilities in monitoring the forest environment. The capability now exists to take a "quick look" at the resource, compare these results with annual updates and assess the need for reinventory or detailed investigations of un-reconciled changes. Remote sensing and image processing will be able to provide basic resource information that is as dynamic as the forest itself, thereby providing managers with a powerful, timely source of information.

ORIGINAL PAGE
BLACK AND WHITE PHOTOGRAPH

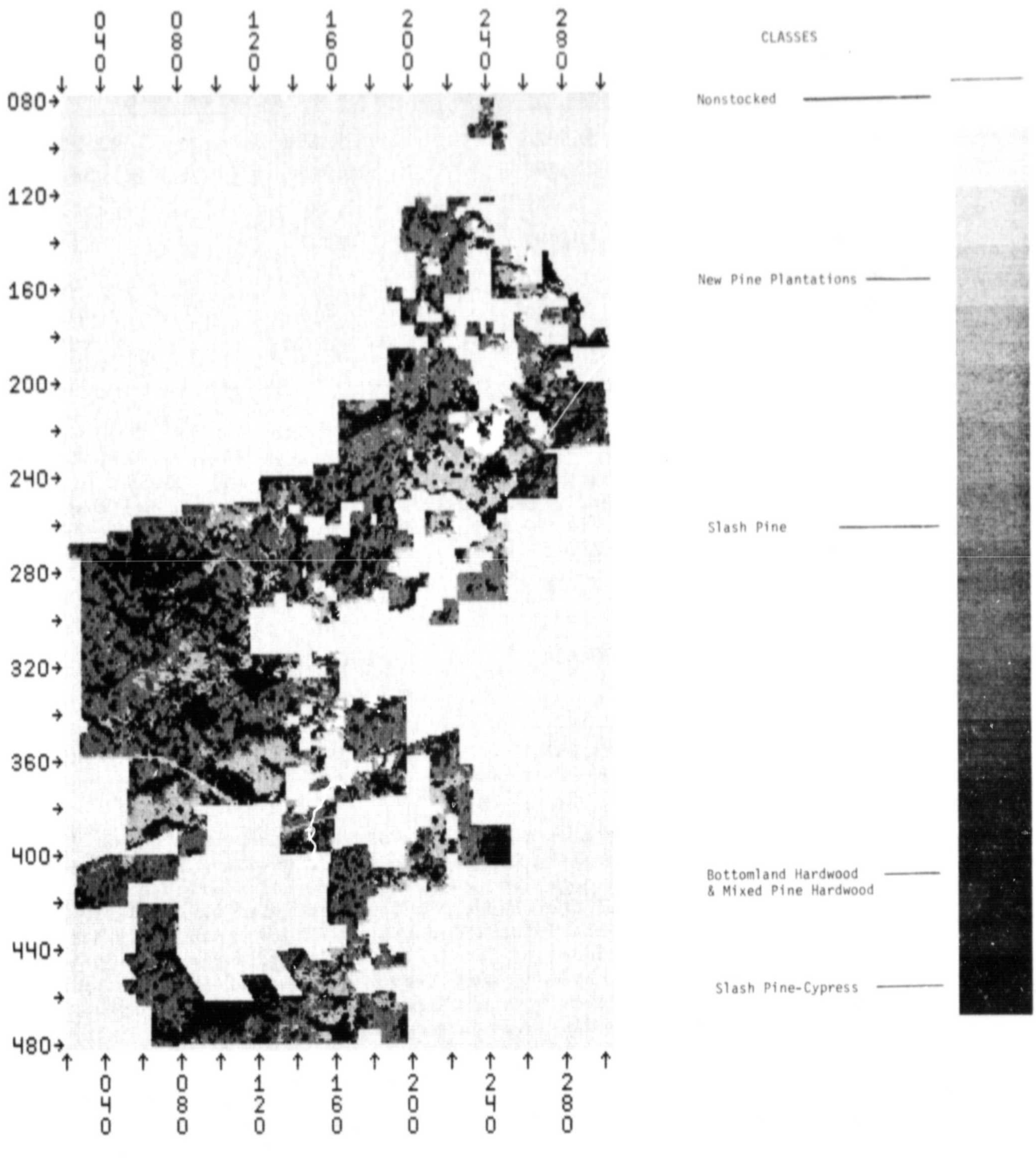


Figure 3.3.1 Classification of December 7, 1977 Landsat data for the Knabb Tract. Area statistics for this classification appear in Table 3.3.1.

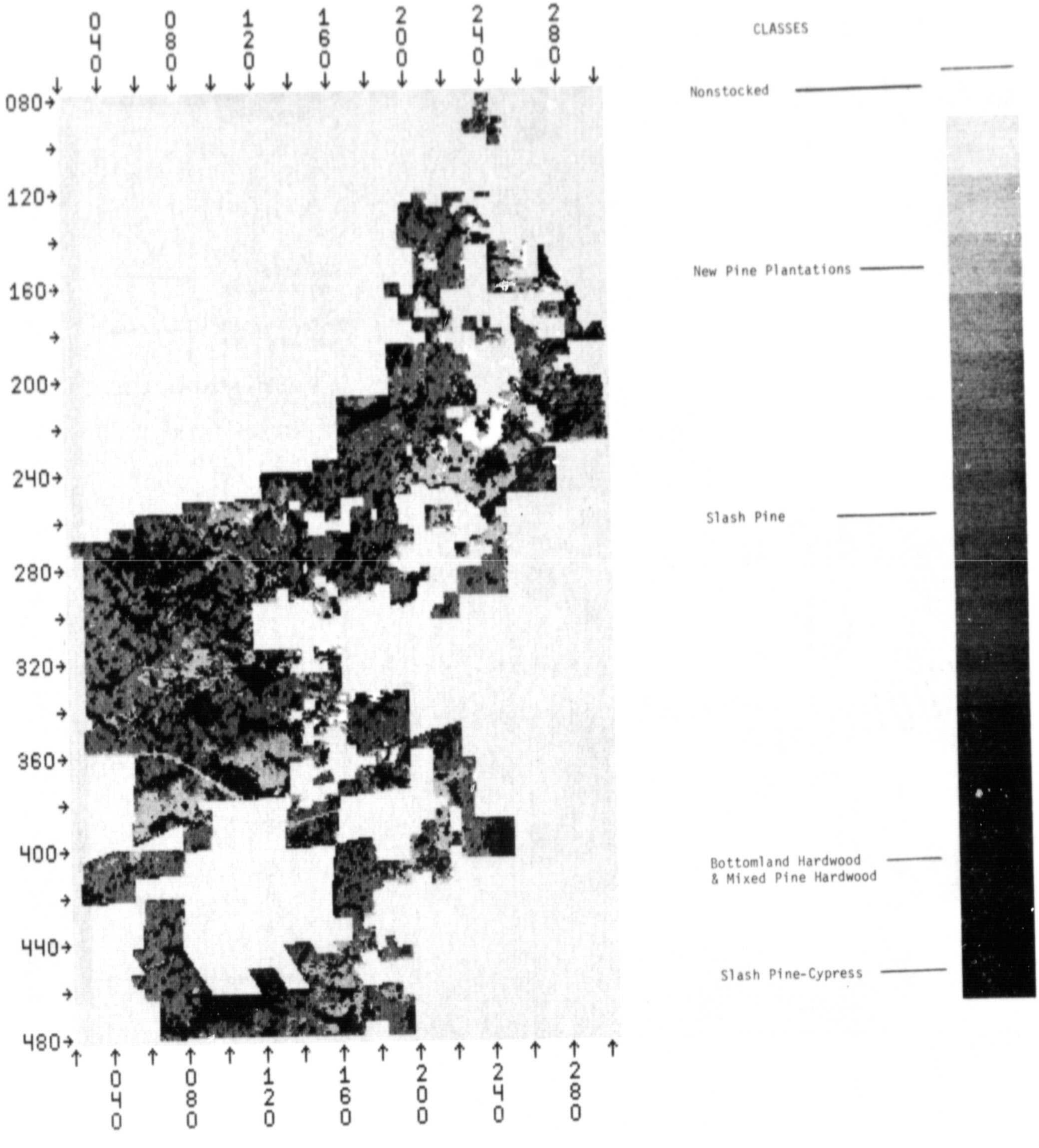


Figure 3.3.2 Classification of February 12, 1979 Landsat data for the Knabb Tract. Area Statistics for this classification appear in Table 3.3.2.

ORIGINAL PAGE
BLACK AND WHITE PHOTOGRAPH

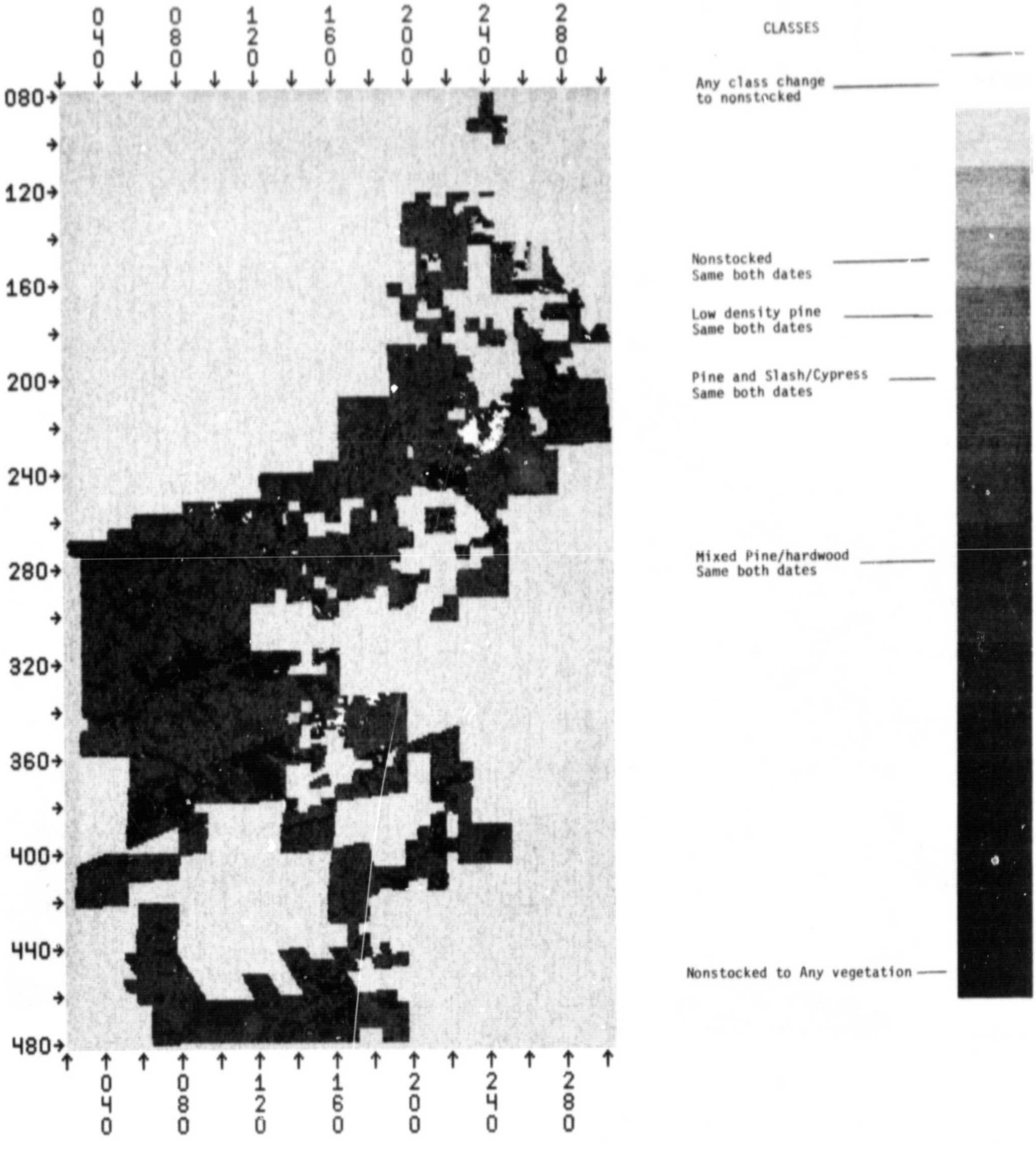


Figure 3.3.3 This is an example of a change map which shows the areas which changes between the 1977 and 1979 classifications. Area statistics for the changes shown on this map appear in Table 3.3.3.

3.4 Ratio Evaluation

During the course of the FRIS Project LARS Project personnel became aware of forest managements need to quantitatively relate Landsat and forest inventory data. One approach that was especially noteworthy involved the application of regression analysis to Landsat MSS reflectance values. The predicted variable was the age of pine plantations, which is an indirect measure of crown closure. Crown closure is a measure of stand stocking which is an inventory measure.

More precisely the ratio of the infrared to visible band responses are assumed to be affected by stand occupancy, which is reflected in crown closure. As stands mature, individual tree crowns occupy a greater proportion of the site (figure 3.4.1). The increasing crown closure affects the ratio, which in preliminary tests corresponds well to a measure of age.

3.4.1 Knabb and Picayune Ratio Results

The ratio of IR channels to visible channels from December 1977 Landsat data for the Knabb and Picayune tracts were used to predict the age of selected pine fields. The exact ratio used, the method of picking pine fields, the analysis used to predict the fields' ages and the results of these predictions are outlined below.

The exact data ratio generated was as follows:

$$\text{ratio} = 40.0(C3 + C4)/(C1 + C2 + 0.1)$$

where

C1 = channel 1

C2 = channel 2

C3 = channel 3

C4 = channel 4

The multiplier 40.0 and the constant 0.1 were needed to enhance the range of and information in the data, and to prevent a divisor of zero.

The Knabb Tract was first categorized into pine and non-pine classes. From this classification fourteen fields of seemingly homogeneous pine were selected and the average ratio for each field was determined. Due to the proximity of this tract to the Fargo test site and their similar physiography, a regression equation developed for Fargo was used to predict the ages of the selected Knabb fields. Four of the Knabb fields were dropped from further analysis. Two of these discarded fields were accidently picked outside the Knabb boundaries and the other two dropped fields were inaccessible for checking ground truth. Of the ten pine fields left, a ground inspection of the area established that (1) all ten fields were pine, and (2) nine of the ten fields had ages within the ninety percent confidence interval for each predicted age. Ages were derived by taking increment cores and counting growth rings of randomly selected dominant trees.

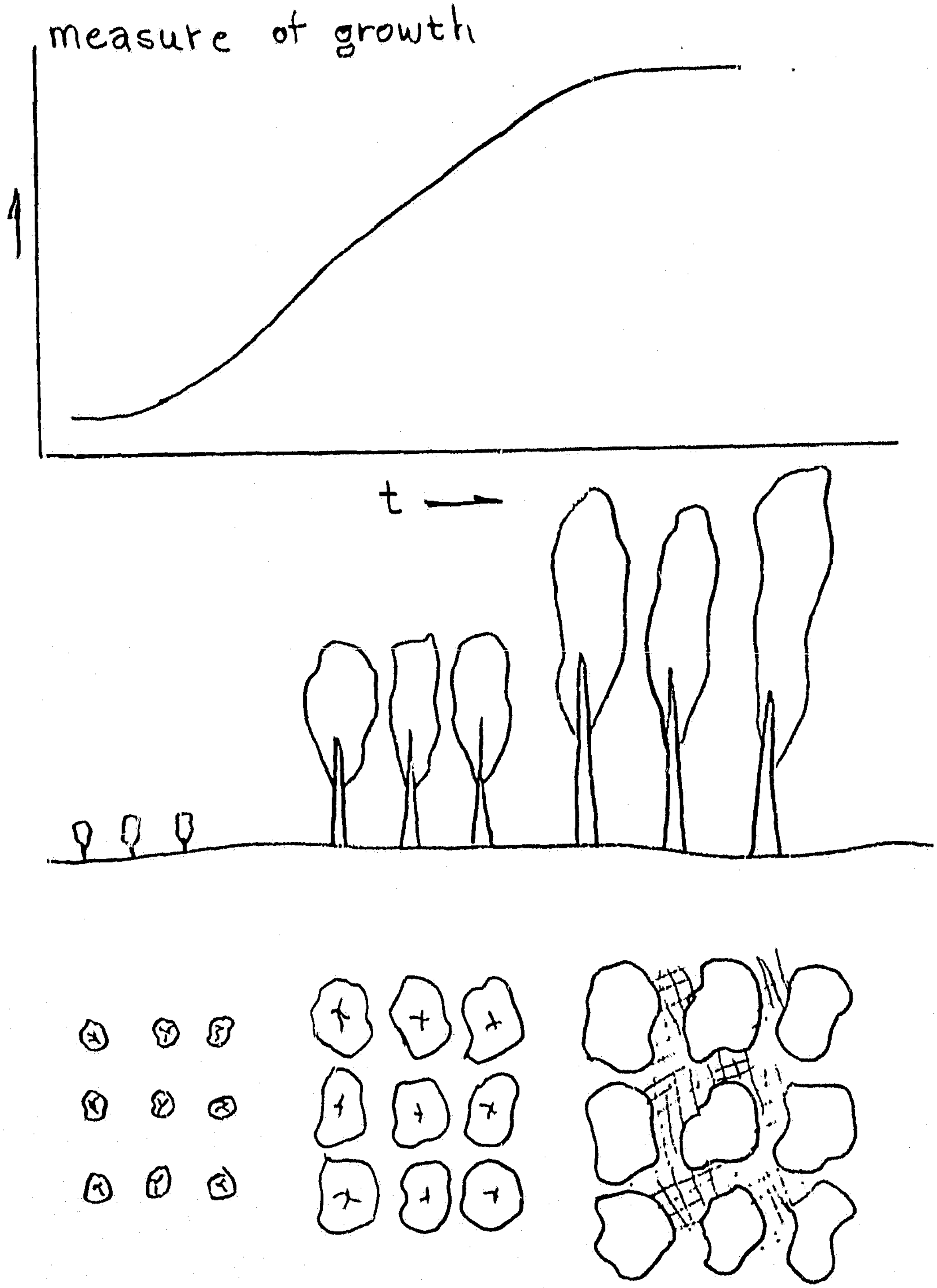


Figure 3.4.1 This is a conceptual representation of the biological growth response over time. The basic hypothesis of a ratio evaluation assumes that as a plantation matures the reflectance of the tree crowns will be the dominant factor affecting the calculated ratio. Therefore, this measure of crown closure can be related to stand age.

Table 3.4.1 shows the preliminary results obtained by applying the Fargo prediction equation to the ratio calculated from Landsat data over the Knabb Tract.

Similarly, ten pine fields were chosen from the Picayune Test Site and the predicted age of each field was determined from an equation developed specifically for the Picayune data. The age for each field was verified by ground investigation. Table 3.4.2 shows the preliminary results.

Table 3.4.1 Preliminary results for ten pine plantations in the Knabb Tract.

Fields #	Age Measured on site	Age Predicted from Landsat Ratio	90% CI on predicted age
1	30	19	(9.4, 37)
2	40	26	(13, 51)
3	16	24	(12, 46.7)
4	24	16	(8, 31)
6	32	12	(6, 25)
7	16	20	(10, 40)
8	14	27	(14, 54)
10	5	4	(2, 9)
13	29	19	(9.5, 38)
14	18	17	(8, 33)

Prediction Equation:

$$\log_{10}(AGE + 1) = -9.333088 + 5.567559 \log_{10}(\text{ratio})$$

Table 3.4.2 Preliminary trials of an age prediction equation using Landsat ratio values for Picayune, Mississippi.

Field	Ground Verified Age	Age Predicted from Landsat Ratio
1	15	17
2	18	9
3	14	10
*4	26	6
5	2	3
6	14	10
7	13	18
8	0	1
*9	26	9
10	14	16

Prediction Equation:

$$\log_{10}(\text{AGE} + 1) = -4.913634 + 3.218647 \log_{10}(\text{ratio})$$

*Both these fields had actual ages beyond the range of the regression equation. Of the ten Picayune fields checked, two (fields 4 and 9) fell outside the 90% confidence interval for the predicted age.

Another application of the generated ratio channel was a classification of the Knabb area done solely with the ratio channel (LEVELCLASSIFY). Analysis done on the Fargo test site revealed the fact that the average ratio of hardwood fields in winter data fell below the average ratio of pine fields. Hence using the ratio intervals developed on the Fargo test site, the Knabb Tract was classified into hardwood, young pine (less than 15 years old), and old pine (15 years old or over).

Since the levels for the level classifier were determined using averages over fields, these levels did not apply directly to classifying pixels. Also the levels were determined on another site causing even more inherent error in this classification. The ten pine fields used in Table 3.4.1 and four hardwood fields were used to test the accuracy of this classification. The results are presented in Table 3.4.3.

Table 3.4.3 Classification performance for a LEVELCLASSIFY approach using Landsat ratio input for the Knabb Tract.

Classes	Classification accuracy
young pine	45.8
old pine	57.4
hardwood	60.0

overall accuracy = 57.4

Hence the ratio of IR to visible Landsat channels has shown usefulness in predicting the ages of pine fields even over areas with no ground truth.

Preliminary results using the generated ratio as a classification channel, however, has shown questionable usefulness. This does not preclude further investigation of levels classification technology. The levels classifier is significantly faster than a maximum likelihood per-point approach and could therefore be beneficial for "first look" evaluations of large areas. Additional investigation into the application of this approach needs to be pursued.

4.0 BENEFIT/COST ANALYSIS

The unique composition of the FRIS Application Pilot Study introduces some complications when conducting a benefit/cost study. Normally, publicly funded projects are evaluated by utilizing a "social" benefit/cost approach where maximizing net social benefit is the dominate objective. Therefore, all benefits and costs are adjusted for market failures or externalities. In addition, the problem of defining what constitutes a benefit or cost and to whom various benefits and cost are accruing is more complex in the social case then in the private analysis. However, since STR is a privately owned corporation, it uses a private benefit/cost analysis approach when making capital investment decision. These decisions are different from the social benefits and costs in that market externalities are not considered (Dasgupta, et al., 1972).

4.1 Social Benefit/Cost - A Conceptual Approach

NASA funded the FRIS project with the expectation that the demonstration of a viable forest information system using LANDSAT imagery will increase the use of LANDSAT in the forest products industry. With the addition of this better information, forest managers should be able to more efficiently manage their resources and increase productivity. This increased productivity could result in decreased costs of production and thus lower consumer prices.

Figure 4.1 is a simple supply/demand model for a hypothetical forest product. The line ABCD is the demand curve for the product. S1 and S2 are supply curves (i.e., cost curves for the production of the product). If before FRIS S1 is the supply curve, the price of the product is P1 and quantity F is demanded. The net consumer surplus in AB P1 (Mishan, 1976, pp. 416-429). Assuming that the information developed by the FRIS project reduces the costs and the supply curve is shifted to S2, the consumer surplus is AC P2. The net increase in consumer surplus is P1 BC P2 (AC Ps - AC P1). The basic question is whether the increased consumer surplus is equal to or greater than the project costs.

Unfortunately, there is no way to estimate the forest products industry's response to the FRIS technology or if in fact a lowering of the supply curve will occur. However, some analysis can be accomplished which will provide a feed for the level of magnitude of a supply curve shift which would justify the initial costs. To develop this estimate, the following assumptions are made:

1) The price elasticity of demand for paper is - 0.2 (Haynes, Holley, and King, 1978).

2) It will take 10 years for enough firms to adopt the FRIS technology to result in a one time reduction in the aggregate supply curve.

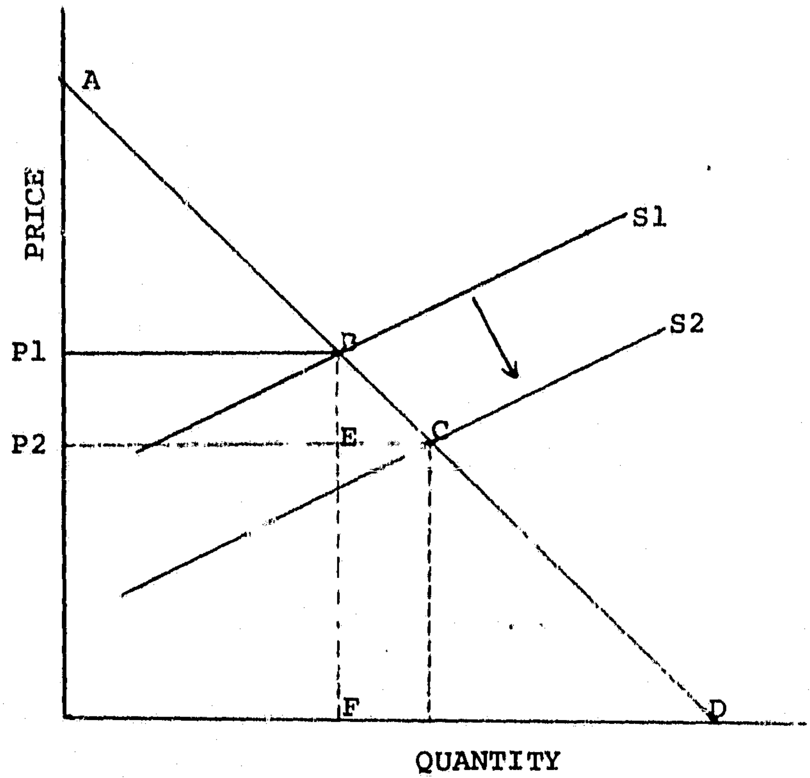


Figure 4.1 The consumer surplus from a decline in the supply curve from S1 to S2 is P1 B C P2.

3) The present trend of 2.4 percent per/annum increase in paper consumption will continue through 1990.

4) The present trend of 6.8 percent per/annum increase in price will continue through 1990.

5) The social rate of discount is 10 percent per/annum.

6) All costs of implementing the future FRIS systems are borne by the firms from private profits, i.e., there are no additional public costs.

The net social benefit of the FRIS project can then be calculated by:

$$NSB = DCS - DC$$

where NSB = net social benefit

DCS = discounted consumer surplus in year 1990

DC = capitalized or discounted costs

Annual project costs are: 1977 - $21,983; 1978 - $214,404; 1979 - $235,008; 1980 - $183,842. The consumer surplus resulting from a shift in the supply curve can be estimated by multiplying the amount of the price decrease times the original quantity (P1 P2 EB, Figure 1) plus one half of the price change times the quantity change (BCE, Figure 1).

Since the price elasticity of demand is known (- 0.20), it is possible to estimate the consumer surplus occurring if the supply curve is shifted. Two shifts are considered here. One shift results in a 0.1% price decrease, the other a 0.01% price decrease. The corresponding quantity increases are 0.02% and 0.002% respectively. The calculation of the consumer surplus assumes an initial price of $740/ton and consumption of 47,282,919 tons in 1990. These levels were calculated by compounding the current price and consumption by the annual rates of increase stated in the assumptions.

Table 4.1 gives the results of this analysis. It can be seen that it will take very little effect on the costs of production to recover the cost of the FRIS profit given the assumptions stated above. These calculations also do not include the potential affects on paperboard, speciality papers, of any solid wood products. Clearly if the FRIS project does lead to increased supply and lower production costs, the public funds were well spent.

Table 4.1. Net social benefit (NSB) calculation.

Year		Item	Actual Cost ($)	Time Adjust.	Present Value ($)
1977		Project Cost	-21,983.	1.3309	-29,259.37
1978		Project Cost	-214,404.	1.2099	-259,428.84
1979		Project Cost	-235,008.	1.1000	-258,596.80
1980		Project Cost	-183,842.	1.000	-183,842.00
1990	a	Consumer Surplus	35,036,836.	0.3855	13,506,700.00
	b	Consumer Surplus	3,503,368.	0.3855	1,350,548.00
NSB	a	From 0.1% price reduction			12,775,573.00
	b	From 0.01% price reduction			619,421.00

4.2 Private Benefit/Cost Analysis

The aggregate supply curve will fall only if a large number of firms adopt a FRIS type system leading to widespread improvement in forest management and productivity. Forest products firms will adopt the technology only on the basis of a private, not social, benefit/cost analysis.

A private benefit/cost analysis is technically identical to social benefit/cost analysis, in that, both discount benefits and costs to obtain net present values or internal rates of return. The major difference between the social and private analysis is in the definition of a cost or benefit. For private firms, there are no externalities by which the benefits or costs are adjusted from the observed market price. For example, in social analysis, taxes are not considered a cost, but simply a redistribution of wealth. To the private firm, taxes are definitely considered a cost of production, and thus reduce profits.

If the objective of the project is the increased use of LANDSAT by the private sector, the critical analysis is that of the private firm. The technology will be accepted or rejected on the basis of the private benefit/cost analysis. Therefore, STR acceptance of the project and implementation of an operational system is the best measure of the success of the project. By this measure the system is acceptable on a private benefit/cost basis. Reports issued by STR on this project will detail that part of the private benefit/cost analysis.

5.0 REFERENCES

Dasgupta, Dartha, Amartya Sen and Stephen Marglin. 1972. Guidelines for Project Evaluation. United Nations, New York. 383 pp.

Mishan, E. J. 1976. Cost-Benefit Analysis. Praeger Publishers, New York. 454 pp.

Haynes, R.W., Holley, D.L. and King, R.A. 1978. A Recursive Spatial Equilibrium Model of the Softwood Timber Sector. Technical Report Number 57, School of Forest Resources, North Carolina State University, Raleigh, N.C. 71 pp.

APPENDIX A

Lists of Preprocessing and LARSYS Software that was transferred to NCC.

Preprocessing Subroutines

ACCNT	DISMAT	FDRITE	LNDERR	LNDSUM
BINSCH	EOT	JTOR	LNDF17	LNDSUP
CASCII	ERRPTR	LARS17	LNDHED	LNDTRA
CHAR	FILEOP	LNDANC	LNDIMA	LNDVAL
CONDMP	FILSRH	LNDANN	LNDINT	LNDWRT
CPTIME	GCTROL	LNDARC	LNDLID	LNDWUP
CTLCBC	GEMANG	LNDBIL	LNDMIL	PAGLOC
CTLSPN	GEMCHK	LNDCOL	LNDPAG	STDHDR
CT01A1	GEMCOM	LNDCOR	LNDPRM	TAPMC
DISK	GEMCOR	LNDCTL	LNDRDR	USAGE
DISKOP	GETACT	LNDDIR	LNDSTR	XMOUNT

Major LARSYS and LARSYSDV Routines

LARSYS	LARSYSDV
PICTUREPRINT	BIPLOT
STATISTICS	COMPARERESULTS
IDPRINT	CLUSTER
LISTRESULTS	SMOOTHRESULTS
PUNCHSTATISTICS	SEPARABILITY
LINEGRAPH	CLASSIFYPOINTS
COLUMNGRAPH	PRINTRESULTS
HISTOGRAM	CHANNELTRANSFORM
GRAPHISTOGRAM	COPYRESULTS
COPYRESULTS	MERGESTATISTICS
EXCOMAND	RATIOMEANS
	SECHO

APPENDIX B

Example of control card reference files and program abstract for the LARSFRIS program COMPARERESULTS.

Note: This program makes comparisons between user specified classes that occur on two classifications.

LARS Program Abstract 380

MODULE IDENTIFICATION

Module Name: COMSUP Function Name: COMPARERESULTS

Purpose: Supervisor for the function.

System/Language: CMS/FORTRAN

Author: John Cain Date: 6/1/79

Latest Revisor: Date:

MODULE ABSTRACT

Supervisor for the COMPARERESULTS function.

PURDUE UNIVERSITY
Laboratory for Applications of Remote Sensing
1220 Potter Drive
West Lafayette, Indiana 47906

1. Module Usage

CALL COMSUP

There are no arguments to COMSUP. It is called from LARSMN when the COMPARERESULTS function is requested. Control returns to LARSMN upon completion of the function.

2. Internal Description

COMSUP receives control from LARSMN to perform the COMPARERESULTS processing. COMSUP calls COMRDR to read and interpret the control cards. Upon return from COMRDR, COMSUP calls CHANGE to finish the processing. Subroutines called by COMSUP: COMRDR
CHANGE

3. Input Description

Not Applicable

4. Output Description

Message numbers are listed below, see User's Manual for text of message.

MESSAGES

INFORMATIONAL

I 26
I 264

5. Supplemental Information

Not Applicable.

6. Flowchart

Not Applicable.

LARS Program Abstract 381

MODULE IDENTIFICATION

Module Name: COMCOM Function Name: COMPARERESULTS

Purpose: Block data

System/Language: CMS/FORTRAN

Author: John Cain Date: 6/1/79

Latest Revisor: Date:

MODULE ABSTRACT

This is the BLOCK DATA subroutine for the COMPARERESULTS common block COMCOM.

PURDUE UNIVERSITY
Laboratory for Applications of Remote Sensing
1220 Potter Drive
West Lafayette, Indiana 47906

LARS Program Abstract 382

MODULE IDENTIFICATION

Module Name: COMRDR Function Name: COMPARERESULTS

Purpose: Reads and interprets function control cards.

System/Language: CMS/FORTRAN

Author: John Cain Date: 6/1/79

Latest Revisor: Date:

MODULE ABSTRACT

COMRDR interprets all function control cards for COMPARERESULTS. Checks are made for complete and valid specifications and the proper input-output devices are attached.

PURDUE UNIVERSITY
Laboratory for Applications of Remote Sensing
1220 Potter Drive
West Lafayette, Indiana 47906

1. Module Usage

CALL COMRDR (Z,NAME)

Output Arguments:

Z-LOGICAL*1 each element initialized to .FALSE.

Z(i,1,j)=.TRUE. - if class i from the first classification is part of user-defined class j.

Z(m,2,n)=.TRUE. - if class m from the 2nd classification is part of user-defined class n.

NAME-I*4 - contains the names of the user-defined classes.

Listed below are the actions taken when the following control cards are read.

FIRSTRESULTS (SECONDRESULTS)	TAPE - the variable TAPE1 (TAPE2) is set to the given tape number. FILE - the variable FILE1 (FILE2) is set to the given file number. DISK - DISKFG is checked to be sure that the DISK option is not already in effect, the tape and file numbers are checked to be sure that both the DISK option and TAPE option are not being used simultaneously. If they are, then an error message will be printed and the DISK will be used. RESLT1 (RESLT2) is set equal to CLASSR.
NEWRESULTS	TAPE - the variable TAPE3 is set equal to the given tape number. FILE - the variable FILE3 is set equal to the given file number. INIT - the variable INITFG is set equal to 1. DISK - the same checks are made as above in addition to a check to see whether the INIT and DISK option were used simultaneously. DISKFG is set equal to one and RESLT3 is set equal to CLASSR.
BLOCK	RUN - the variable RUNNUM is set equal to the given run number. LINE - STALIN is set equal to the first entry (the starting line of the area to be investigated). LASLIN (last line) is set equal to the second entry and finally LININT (line interval) is set equal to the last entry. COL - same as above where the variables are: STACOL - first entry, LASCOL - second entry and COLINT - final entry.

DATA	A check is made for the presence and validity of all information.
CLASS name	The name given is stored in the array NAME.
FIRST N1, N2,... SECOND M1, M2,...	Using the given class numbers the appropriate locations in the Z(64,2,64) array are set .TRUE. (i.e. if these are the FIRST and SECOND cards for the jth user-defined CLASS, then the following assignments are made for array Z:

```
     Z(N1,1,j) = .TRUE.
     Z(N2,1,j) = .TRUE.
       .
and    .
     Z(M1,2,j) = .TRUE.
     Z(M2,2,j) = .TRUE.
       .
       .
```

2. Internal Description

COMRDR uses the standard card reader logic in using CTLWRD, CTLPRM and IVAL to read and interpret control cards.

COMRDR begins by initializing all flags and arrays that are used to convey control card information. It then goes into a loop of reading and interpreting the input specifications and the BLOCK card. When the DATA card is read COMRDR checks for the presence of all information and its validity. Another loop is entered and the CLASS cards and their corresponding FIRST and SECOND cards are read. The class numbers from the FIRST and SECOND cards are used to set appropriate values in the Z array to a logical .TRUE.

Z(i,1,j)=.TRUE. if the Ith class from the FIRST results file is part of user-defined class j.

Z(k,2,m)=.TRUE. if class k from the SECOND results file is part of user-defined class m.

This loop is exited when an END card is read. Once this card is read, COMRDR calls CHTAPE to mount the specified tapes. If a disk was specified as an input device, COMRDR first checks to be certain both a tape and disk were not specified for a single input. It then reads from the results file to be sure it exists on the disk. If a disk was specified as an output device, checks are made to be sure there is sufficient space for the output results. TSPACE makes a search for a larger disk if necessary. COMRDR finally returns control back to COMSUP. Subroutines called by COMRDR:

CTLWRD	CTLPRM	TSPACE
BCDFIL	CHTAPE	RTMAIN
IVAL	ERPRNT	

3. Input Description

Function control cards for COMPARERESULTS are read via CTLWRD.

4. Output Description

Message numbers are listed below, see User's Manual (vol. 3) for text and explanation of message.

MESSAGES

INFORMATIONAL	ERROR
I0261	E620-E633
I0262	

5. Supplemental Information

Not applicable.

6. Flowchart

Not applicable.

LARS Program Abstract 383

MODULE IDENTIFICATION

Module Name: COMPAR Function Name: COMPARERESULTS

Purpose: Compare 2 lines of classification results

System/Language: CMS/FORTRAN

Author: Susan Schwingendorf Date: 3/28/79

Latest Revisor: Date:

MODULE ABSTRACT

COMPAR compares two lines of classification results (presumably from two different classifications which are registered to each other) against user defined change classes in a logical array, and writes the output class number in the output vector.

PURDUE UNIVERSITY
Laboratory for Applications of Remote Sensing
1220 Potter Drive
West Lafayette, Indiana 47906

1. Module Usage

CALL COMPAR (NCOLS,NCLASS,Z,BUFF1,BUFF2,BUFF3)

Input Arguments:

NCOLS - INTEGER*4, the number of columns of classified data.

NCLASS - INTEGER*4, the number of classes defined by the user in array Z.

Z - LOGICAL*1 (64, 2, 64) array containing user defined change classes. (Initialized to .FALSE.)

Z(I,1,K) = .TRUE.
Z(J,2,K) = .TRUE. means a point in class I from classification 1 (BUFF1) and in class J on classification 2 (BUFF2) should be assinged to class K in BUFF3.

BUFF1 - LOGICAL*1 (2*NCOLS + 4) vector containing classified data from first classification. First full word is line number. Then the second byte of each halfword contains the next class number.

BUFF2 - LOGICAL*1 (2*NCOLS + 4) vector containing classified data from the second classification. First full word (4 bytes) is the line number. Then the second byte of each halfword contains the next class number.

Output Arguments:

BUFF3 - LOGICAL*1 (2*NCOLS + 4) vector of change classes for this line. The first full word contains the line number. Then the second byte of each halfword contains the assigned change class number.

2. Internal Description

The line number is written in the first word of BUFF3. The next class number is then extracted from BUFF1 and BUFF2 and assigned to integer variables CLASS1 and CLASS2. A loop through the logical array Z determines which output class to assign this point to. If Z (CLASS1,1,J) and Z (CLASS2,2,J) are true, then the point is assigned to class J. If it belongs to none of the defined output classes, then it is assigned a class number NCLASS+1. The output class numbers are written in BUFF3.

3. Input Description

Not Applicable.

4. Output Description

Not Applicable.

5. Supplemental Information

Not Applicable.

6. Flowchart

Not Applicable.

LARS Program Abstract 384

MODULE IDENTIFICATION

Module Name: CHANGE Function Name: COMPARERESULTS

Purpose: Compares two classification results files and outputs the compared results.

System/Language: CMS/Fortran

Author: John Cain Date: 6/1/79

Latest Revisor: Date:

MODULE ABSTRACT

CHANGE is the main subroutine for COMPARERESULTS. It reads from two input tapes (or one disk and one tape), calls COMPAR, then outputs the data in standard LARSYS classification results file format to tape or disk.

PURDUE UNIVERSITY
Laboratory for Applications of Remote Sensing
1220 Potter Drive
West Lafayette, Indiana 47906

1. Module Usage

CHANGE

CALL CHANGE(Z,NAME)

Input Arguments:

Z-Logical*1 - Z(i,1,j)=.TRUE. if class i from the first classification is part of user-defined class j.

Z(m,2,n)=.TRUE. if class m from the 2nd classification is part of user-defined class n.

NAME - I*4. - Contains the names of the user-defined classes.

Output Arguments:

Not Applicable.

2. Internal Description

CHANGE first reads the file numbers from the input tapes, and the tape numbers passed through the common block, and creates a code that takes the place of the CLASSIFICATION STUDY number. The code format is the first tpae and file numbers followed by the second tape and file numbers. CHANGE then reads the area identification record (record type 5) from both input sources and checks to see whether they are valid for the given BLOCK CARD; if not, appropriate error messages are printed and the function is terminated. Record types 1-5 are written to the output tape (DISK). The inputs are positioned to the correct line number and shifted to the correct column number. CHANGE then calls COMPAR to determine which class each point belongs to and this information is used to create file type 6. Finally record types 7 and 8 are written and control is returned to COMSUP. If the output device is a tape, then a final record type 1 and END OF FILE Mark are written before returning to the supervisor. Subroutines called by CHANGE:

COMPAR
RTMAIN
TAPOP

3. Input Description

Record types 1, 5, 6 of the LARSYS classification results files are read from the two input devices, RESLT1 and RESLT2. One of these may be a disk (DSRN CLASSR). Tape drives 181 (CPYOUT) and 182 (SCNDTP defined in COMCOM) are used as inputs.

4. Output Description

The output device RESLT3 initially has a DSRN of MAPTAP. If a disk is used (only if one is not used for input), the LSRN is changed to CLASSR. Tape drive 180 is used for output to facilitate the run of PRINTRESULTS on the output data immediately after the COMPARE-RESULTS run. The output is a classification results file in standard LARSYS format.

Message numbers are listed below, see User's Manual for text and explanation of message.

MESSAGES

INFORMATIONAL	ERROR
I0263	E634-E638

5. Supplemental Information

Not Applicable.

6. Flowchart

Not Applicable.

LARS Program Abstract 385

MODULE IDENTIFICATION

Module Name: CHTAPE Function Name: COMPARERESULTS

Purpose: Mounts and positions results tapes

System/Language: CMS/FORTRAN

Author: E.M. Rodd Date: 9/5/72

Latest Revisor: J. Cain Date: 6/1/79

MODULE ABSTRACT

CHTAPE mounts and positions the results tape (or a tape to be used as output for copying results files.)

PURDUE UNIVERSITY
Laboratory for Applications of Remote Sensing
1220 Potter Drive
West Lafayette, Indiana 47906

1. Module Usage

CHTAPE

CALL CHTAPE (RQTAPE,RQFILE,MODE,UNIT)

Input Arguements

RQTAPE	- I*4.	Tape number of requested tape. A tape number of 0 is a request for a scratch tape.
RQFILE	- I*4.	File number of requested file. If RQFILE is = 0, then the tape will be initialized by writing a record type 1 on the results tape with filetype= 0.
MODE	- I*4.	Flag indicating usage of CHTAPE. MODE = -1 indicates CHTAPE has been called to mount and position a tape to be used for copying results files onto. MODE = 0 indicates that a results tape is being mounted for reading a results file. In this case, the tape is mounted ring out. Also, if MODE = 0, RQFILE = 0 is invalid and will cause an error when an attempt is made to write on the tape. MODE = 1 indicates a tape is being mounted for writing a new results file (or continuing a suspended classificaiton). Since the unit value is passed in the call, Mode(1) = Mode(-1).
UNIT	- I*4.	DSRN of tape being mounted.

Output Arguments

RQTAPE	- I*4.	When MODE = 0, set to -1 if requested tape file was full and user decided to use disk for results. Otherwise, remains unchanged.
RQFILE	- I*4.	When MODE = 1, set to -1 if requested tape file was full and user decided to use disk for results. Otherwise, sends back current file position of tape.

CHTAPE checks the validity of the tape by reading the record type 1 from the tape and verifying the tape and file number as well as checking for the correct type of file. Any attempt to overwrite an existing file causes CHTAPE to ask the user (via the typewriter) if he wishes to overwrite the file, respecify a new results card, or terminate the function. Note, however, that if a request has been made to initialize a tape, no checking is performed on previous contents.

2. Internal Description

See Output Description. Subroutine called by CHTAPE:

TAPOP	RINGIN	IVAL
MOUNT	CTLWRD	ERPRNT
CPFUNC	CTLPRM	RTMAIN

3. Input Description

The record type 1 of the results tape is read for each file up to and including the file needed. That is, if file 4 is requested the record type 1 is read from files 1-4.

4. Output Description

The following information messages are issued under the circumstances listed. The term filetype means the filetype code from record type 1 of results file (the program uses variable CHECK for this number).

I0042 is typed when a tape has been mounted and before CHTAPE positions it. This message is not typed when the tape is being initialized or when the correct tape number was already mounted.

I0043 is typed when MODE = ± 1 and filetype of the requested file = 0.

I0044 is typed when MODE = ± 1 and filetype of the requested file = 1 and the restart flag from GLOCOM (RESTRT) is not = 1.

I0045 is typed when the tape is correctly positioned. This is not typed when initializing a tape.

After I0043 and I0044, the user is asked whether he wishes to overwrite the file, respecify a new results card with a new tape and/or file or disk option, or terminate the function.

I0100 is typed to allow entry of the new results card. This occurs when the user requests to respecify the results card.

I0101 is typed to confirm usage of disk for results and occurs whenever disk is specified on the results card.

The following error messages are typed under the conditions listed.

E361 is written when the tape is being filed forward and a file is encountered with filetype other than zero before the requested file is reached and MODE = 0.

E362 is written when the circumstance for E361 occurs and MODE = 1. It is also written when MODE = 1 and the filetype of the file requested is = -1.

E363 is written if the RESTRT flag is = 1 and the filetype of the requested file is not = 1.

E364 is written when MODE = 1 and the filetype of the file requested = 1.

E365 is written when an EOF is read on the results file. This should never occur with valid results files.

For message texts refer to the User's Manual.

5. Supplemental Information

This section deals with the handling of tapes by CHTAPE.

Input:

If a tape is mounted on the device and it is the incorrect tape number (as noted from the appropriate status words in GLOCOM), TOPRU is called to unload the tape before the correct tape is mounted. If the correct tape is mounted, CHTAPE will check for the ring in if MODE = ± 1. If the ring is not in, the tape in unloaded and MOUNT is called to mount the tape with the ring in. If the correct tape is mounted, CHTAPE assumes that the file number (as recorded in GLOCOM) is correct and moves the tape backwards or forwards to find the requested file. CHTAPE is a modified version of MMTAPE.

Output:

The tape is mounted with ring in for MODE = ± 1 and with ring out for MODE = 0.

The tape is left positioned at the beginning of the requested file. When the tape is initialized a TOPRW is used to do this.

6. Flowchart

Not applicable.

APPENDIX C

Example of Structured English Module

```
PROGRAM lndsup
      (* Landsat to LARSYS program *)

   DECLARATIONS
      control-cards : file;
      end-of-file   : file-condition;
      error-level, abort : error-level-indicator;
      start, stop   : time;
      dummy, clock, totcpu, vircpu : computer-usage;

 BEGIN
    CALL cptime(dummy, clock, totcpu, vircpu);
    CALL getime(start);
    REPEAT
       CALL lndint;  (* initialize *)
       CALL lndrdr;  (* read control card file *)
       IF NOT(end-of-file ON control-cards) THEN
          IF error-level < abort THEN
             CALL lndctl;  (* reformat Landsat *)
             CALL lndwup   (* wrap-up reformatting *)
          ENDIF;
          CALL lndsum   (* summarize the job *)
       ENDIF
    UNTIL end-of-file ON control-cards OR
          error-level >= abort;
    CALL usage(clock, totcpu, vircpu);  (* print computer time used *)
    CALL getime(stop);
    CALL lndf17   (* print FORM-17 *)
END
```

```
C     PROGRAM LNDSUP
C
      IMPLICIT INTEGER * 4 (A-Z)
C
C******************************************************************
C
C LANDSAT REFORMATTING SUPERVISOR
C WRITTEN 05/23/79 BY KEN DICKMAN
C
C******************************************************************
C
C  ROUTINES INVOKED DIRECTLY BY LNDSUP
C
C      CPTIME
C      GETIME
C      LNDCOL
C      LNDINT
C      LNDFIT
C      LNDRDR
C      LNDSUM
C      LNDWUP
C      USAGE
C
C******************************************************************
C
C   THE LANDSAT-EDIPS TO LARSYS REFORMATTING ROUTINES
C
C BINSRH = PERFORM A BINARY SEARCH
C CASCII = CONVERT ASCII CHARACTERS (BYTES) TO EBCDIC CHARACTERS
C CHAR   = ASSIGN A LITERAL CHARACTER STRING
C CONDMP = DUMP CONTENTS OF AN ARRAY WITH ADDITIONAL INFO
C CPTIME = GET VIRTUAL AND TOTAL CPU TIME USAGE
C CTLCBC = INSERT COMMAS IN THE CONTROL CARD SO CTLWRD WILL WORK
C CTLPRM = (ENTRY POINT OF CTLWRD) READ PARAMETERS FROM CONTROL CARDS
C CTLSPN = SPAN OVER COLUMNS (ALSO SO CTLWRD WILL WORK)
C CTOIA1 = CONVERT CHARACTER (IN A1 FORMAT) TO INTEGER
C DARSIN = FORTRAN DOUBLE PRECISION ARC SINE
C DATAN  = FORTRAN DOUBLE PRECISION ARC TANGENT
C DCOS   = FORTRAN DOUBLE PRECISION COSINE
C EOT    = WRITE END-OF-TAPE AND MORE
C ERRPTR = PRINT A DOLLAR SIGN BENEATH THE ERROR IN THE CONTROL CARD
C FILSRH = FIND THE FIRST OCCURANCE OF AN EMPTY FILE ON A TAPE
C GEBYTE = COMPARE TWO BYTES FOR GREATER THAN OR EQUAL TO
C GETIME = GET TIME OF DAY
C GTDATE = GET DATE
C GTUNIT = (ENTRY POINT OF TAPOP) FIND THE DEVICE ADDRESS OF A TAPE
C IDCHEK = CHECK LARSYS HEADER RECORD AND MORE
C IVAL   = (ENTRY POINT OF RDUVAL) READ VALUES ON CONTROL CARDS
C JTOR   = CONVERT A JULIAN DATE TO ROMAN (IE. MONTH, DAY, YEAR)
C LNDANC = PROCESS THE LANDSAT ANCILLARY RECORD
C LNDANN = PROCESS THE LANDSAT ANNOTATION RECORD
C LNDARC = WRITE TO ARCHIVE TAPE
C LNDBIL = OUTPUT PROCESS FOR BIL FORMAT
C LNDBSQ = OUTPUT PROCESS FOR BSQ FORMAT                          *
C LNDCAL = CALCULATE LINE CORRECTIONS                             *
C LNDCOL = CHECK THE COLUMNS SPECIFIED
C LNDCOR = PERFORM LINE CORRECTIONS                               *
C LNDCTL = CONTROL ACCORDING TO THE LANDSAT INPUT TAPE STRUCTURE
C LNDDIR = PROCESS THE LANDSAT DIRECTORY RECORD
C LNDERR = RECORDS ERRORS ENCOUNTERED
C LNDFIT = GENERATE RUN COMPLETION FORMS (EG. FORMITS)
C LNDHED = PROCESS THE LANDSAT HEADER RECORD
C LNDIMA = PROCESS THE LANDSAT IMAGE RECORD
C LNDINT = INITIALIZE VALUES
C LNDLID = CREATE LARSYS ID RECORD FROM LANDSAT-EDIPS DIR & HED RECORDS
C LNDMIL = MERGE AND RETAIN BIL BANDS FOR A LINE
C LNDMSQ = MERGE BSQ BANDS FOR A LINE                             *
C LNDNEG = CONVERT LIGHT BOARDER PIXELS TO DARK BOARDER PIXELS
C LNDPAG = PRINT A HEADER AT THE TOP OF THE OUTPUT PAGE
C LNDPAN = PUNCH ANCILLARY DATA FOR FUTURE LANDSAT CORRECTIONS    *
C LNDPRM = READ CONTROL PARAMETERS ON CONTROL CARDS
C LNDRES = RE-START TO GET SUMMARY OF EXECUTION (HELP DEBUGGING)
C LNDRDR = CONTROL CARD READER
C LNDSET = CREATE LARSYS ID, DECIDE MERGES, WRITE UNITS (FOR BSQ)  *
C LNDSUM = PRINTS INFO FOR THE JOB MOST RECENTLY RUN
C LNDSUP = LANDSAT REFORMATTING SUPERVISOR
C LNDTRA = PROCESS THE LANDSAT TRAILER RECORD                     *
C LNDVAL = READ CONTROL PARAMETER VALUES ON CONTROL CARDS
C LNDWRT = WRITE LINE IMAGE IN LARSYS FORMAT
```

```
C WRAPUP  * WRAP-UP LANDSAT REFORMATTING
C MOUNT   * MOUNT A TAPE ON A TAPE DRIVE
C MOVBYT  * COPY BYTES
C TPWRIN  * (ENTRY POINT OF TAPOPT) CHECK THE WRITE PROTECT RING
C STRVAL  * GET A DELIMITED STRING FROM AN ARRAY IN A4 FORMAT
C TPWEF   * (ENTRY POINT OF TAPOPT) WRITE AN END-OF-FILE TO TAPE
C TPFSF   * (ENTRY POINT OF TAPOPT) FORWARD FILE A TAPE
C TPREAD  * (ENTRY POINT OF TAPOPT) READ A RECORD FROM TAPE
C TPRUN   * (ENTRY POINT OF TAPOPT) REWIND AND UNLOAD A TAPE
C TPWRT   * (ENTRY POINT OF TAPOPT) WRITE A RECORD TO TAPE
C USAGE   * PRINT JOB INFO SUCH AS CONNECT, VIRTUAL, AND TOTAL CP TIME
C XMOUNT  * CALL MOUNT WITH ADDITIONAL INFORMATION
C
C *       * DENOTES THAT THIS SUBROUTINE HAS NOT BEEN FORMALLY CODED
C
C*********************************************************************
C
C     IN GLOBAL VARIABLES DEFINITION      (UPDATED 12/05/79)
C
      COMMON /UTLCOM/
C          ERROR CONTROL...
     1          ABCNT, ERRCNT, SEVRTY,
C
C          LOGICAL FILE UNITS...
     2          MAXUNT, PRINTR, READER, TTYIN, TTYOUT, UNIT(4),
C
C          FREQUENTLY USED LITERAL STRINGS...
     3          ABLANK, EBLANK, MSS, RBV, RI, RC
C---------------------------------------------------------------------
C     VARIABLES SET BY THE CONTROL CARD READER
C     TO AFFECT THE REFORMATTING OPERATION
      COMMON /RFRCTL/
C          COMMENTS...
     1          COMMNT(20,2), NOCOMM,
C
C          CONTROL CARDS...
     2          EOFRDR,
C
C          INPUT TAPE...
     3          INDEN, INDET, INMNT, INTAPE(3), TAPNDX, TOTTAP,
C
C          INPUT TAPE REFORMATTING...
     4          BILSCN, COLUMN(3), LINES(3), MXINCH, NOCHAN, USECHN,
     4          USECOL, USELIN,
C
C          OUTPUT TAPE...
     5          FLIGHT(16), MXOUCH, OUFILE, OUTAPE, OUIDEN, OUTDET,
     5          OUTMNT, RUN, USEFLT, USEOUT,
C
C          ARCHIVE TAPE...
     6          ARCDET, ARCMNT, ARFILE, ARTAPE, USEARC, USEARF,
C
C          SCRATCH TAPE...
     7          SCRDET, SCRMNT, SCRNDX, SCRTAP(2), TOTSCR, USESCR,
C
C          MISCELLANEOUS...
     8          PATCH(4)
C
      LOGICAL * 4 ARCDET, ARCMNT, EOFRDR, INDET, INMNT, OUTDET,
     1          OUTMNT, SCRDET, SCRMNT, USEARC, USEARF,
     1          USECHN(30), USECOL(3), USEFLT, USELIN(3),
     1          USEOUT, USESCR
C---------------------------------------------------------------------
C     USING THE INPUT STRUCTURE TO DETERMINE PROCESSING
      COMMON /CTLCOM/
C          THE INPUT STRUCTURE...
     1          NS, PS, PSNS(16,16), RECTYP(6,3), STATES(16),
C
C          THE INPUT FREQUENCY...
     2          COUNT(16), FREQ(16,16),
C
C          SETTING UP THE REFORMATTING...
     3          DONE, MAXRC, MERFLG, RDSCR(1), SELECT, WRUNIT
C
      LOGICAL *4 DONE, MERFLG
C---------------------------------------------------------------------
C     BUFFER VARIABLES
      COMMON /BUFCOM/
C          INPUT BUFFER VARIABLES...
```

OF POOR QUALITY


```
     1         INCHN, INCNT, INLIN, INREC(1347), MXICNT,
C
C        SCRATCH BUFFER VARIABLES...
     2         MAXSCR, PXSCNT, SCRCNT(1), SCRREC(8886),
C
C        LANSYS OUTPUT TO HEADER RECORD...
     3         LANSIC(200),
C
C        FLAM = 17C...
     4         F17(236), FORMS, NUFCRM
C---------------------------------------------------------------------
C     OUTPUT VARIABLES
      COMMON /OUTCOM/
C        GENERAL...
     1         FLUSH, MXOCNT, OUTCHN, OUTCNT, OUTLIN,
     .         OUTREC(8880), RECCNT, ROLL,
C        CONVERTING LIGHT BOARDER PIXELS TO DARK...
     2         FIRCOL(30), LEF, LSTCOL(30), RIT, TOTMAX(4)
C
      LOGICAL * 4 FLUSH
      INTEGER *2 RECCNT, ROLL
C---------------------------------------------------------------------
      COMMON /JCBCOM/ CLOCK(5), CURENT(3),
     $         DATE(3), DUMMY(2), LNDVER, START(3), TOTCPU(2),
     $         UPDATE(2), USERID(2), USERNA(4), VIRCPU(2),
     $         WRTAPE
C
      REAL *4 LNDVER
C---------------------------------------------------------------------
C     BEGIN LANDSAT-EDIPS RECORDS' GLOBAL VARIABLES
C
         COMMON /DIRCOM/ CCTVOL, DAYGEN, DIRDMP, DIRID(1), DLEVTY,
     1         DSCENE(3), DWRS(2), MONGEN, NUMVOL, VERDOC, VERSCF,
     2         YRGEN
         LOGICAL *4 DIRDMP
C
         COMMON /HEDCOM/ ACTSTA(2), ATMSCA, CLNENH, LOGENH, GCAPPL,
     1         GCPRES, HCALWD, HIDDMP, HLEVTY, HRESMP, HSCENE(3),
     2         HSENTY, HWRS(2), IMAID(3), MAPAPP, PIXLN1, PIXLN2,
     3         QUALAS, RCAPPL, RCPRES, TIMEXP(4), IMPREG
         LOGICAL *4 HEDDMP
         INTEGER * 2 IMPREG(16)
C
         COMMON /ANNCOM/ ANNDMP, CFHHRS, FRMCEN(4), LATLON(4), NRESMP,
     1         PROJ, SCALE, SCNANG(4), TYPCOR
         LOGICAL * 4 ANNDMP
C
         COMMON /IMACOM/ IMADMP, LFILL, RFILL, SCNCNT
         LOGICAL *4 IMADMP
C
         COMMON /CPRAID/ MIL, MSL, NO, YES
C
C      END LANDSAT-EDIPS RECORDS' GLOBAL VARIABLES
C
C   END GLOBAL VARIABLES DEFINITION
C
C*********************************************************************
C
C   LOCAL VARIABLES DEFINITION
C
       INTEGER * 4 STOP(3)
C
C*********************************************************************
C
C   LOCAL VARIABLES DESCRIPTION
C
C  STOP   = ENDING TIME FOR THIS JOB SET
C
C*********************************************************************
C
      CALL CPTIME(DUMMY, CLOCK, TOTCPU, VIRCPU)
      CALL GETIME(START)
C
C     PRINTR & TTYOUT ARE ALSO INITIALIZED IN LNDINT
      PRINTR = 6
      TTYOUT = 16
C
      WRITE (PRINTR, 9100) START
      WRITE (TTYOUT, 9100) START
```

ORIGINAL PAGE IS
OF POOR QUALITY


```
 910  FORMAT(' I0021 LANDSAT REFORMATTING BEGINS ', 3A4,
     1   4X, '(LNDSUP)')
C
C     REPEAT PROCESSING  UNTIL (END-OF-FILE(READER) OR ABORT)
 110  CONTINUE
C
C        INITIALIZE COMMON BLOCKS
         CALL LNDINT
         IF (SEVRTY .GE. ABORT) GOTO 290
C
C           READ CONTROL CARDS
            CALL LNDRCR
            IF (SEVRTY .GE. ABORT) GOTO 260
               IF (EOFRDR) GOTO 230
C
C                 CONVERT LANDSAT TO LARSYS
                  CALL LNDCTL
C
C                    WRAP-UP CONVERSION FROM LANDSAT TO LARSYS
                     CALL LNDWUP
                     IF (SEVRTY .GE. ABORT) GOTO 170
                        WRITE (PRINTR, 9140)
                        WRITE (TTYOUT, 9140)
 9140                   FORMAT(///, ' I0002 LANDSAT TO LARSYS COMPLETE',
     1                        4X, '(LNDSUP)')
  170                CONTINUE
                     ENDIF
C 230             CONTINUE
                  ENDIF
C 260          CONTINUE
               ENDIF
C 290     CONTINUE
         ENDIF
C
C
         IF (EOFRDR) GOTO 720
C
C           WRAP-UP SUMMARY FOR ONE CONTROL CARD SET
C
            CALL LNDSUM
  720    CONTINUE
         ENDIF
C
C     UNTIL (EOF(READER) OR ABORT)
      IF (.NOT. EOFRDR .AND. SEVRTY .LT. ABORT) GOTO 110
C
C*****************************************************************
C WRAP-UP SUMMARY FOR ALL CONTROL CARDS
C*****************************************************************
C
C     PRINT VIRTUAL AND TOTAL CPU USAGE FOR ALL JOBS
      CALL USAGE(CLOCK, TOTCPU, VIRCPU)
C
      CALL GETIME(STOP)
      WRITE (PRINTR, 9991) STOP
 9991 FORMAT(12(/), ' I0003 EXECUTION STOPS ', 3A4, 4X, '(LNDSUP)')
      WRITE (TTYOUT, 9992) STOP
 9992 FORMAT(/, ' I0004 EXECUTION STOPS ', 3A4, 4X, '(LNDSUP)')
C
C*****************************************************************
C PRINT FORM 17C
C*****************************************************************
C
      CALL LNDF17
C
      STOP
      END
```

APPENDIX D

Image Registration Functional Specifications

An image registration capability has been determined to be a necessary part of the FRIS III image preprocessing software. Image registration is general enough to mean grid to grid transformation. Thus, while the system is designed to register two coincident Landsat scenes, registration to alternate grid systems may be accomplished with this software as well. Functional specifications will be as follows:

I. Purpose

A. Primary: Registration of two coincident digital images as two Landsat digital image data sets.

B. Secondary: Provide for the registration of any known two-dimensional grid to another known or defined two-dimensional grid.

II. Input images are assumed to be in LARSYS format.

III. Checkpoint Acquisition

A. Manual checkpoint acquisition is possible.

B. Cross-correlation of two coincident digital images may be accomplished by implementation of a numerical integration image correlator.

C. Control may be by set line and column intervals.

D. Alternate control will be from a set of inputted control correlation point locations where a cross correlation is desired, i.e., arbitrary point by point correlation.

IV. Registration transformation

A. Coefficient determination will be calculated for affine, biquad, and bicubic transformation.

B. Transformations through bicubic will be implemented for the registration transformation.

C. Block registration technique will be utilized.
   1. Optimum rectangular block size will be determined for biquadratic and bicubic registrations.

D. Interiors of all blocks will be registered with an affine or linear transformation.

V. Radiometric interpolation

A. Nearest neighbor will be the default.

B. Cubic interpolation will be optimally implemented.

VI. Output images will be produced in LARSYS format.

APPENDIX E

Cubic Interpolation Used in the Image Registration System

The algorithm used in the current image registration system for cubic interpolation of data values is based on a thrid order Lagrange interpolation. The general Langrangian interpolating polynomial for three dimensions is:

$$P_{mn}(X,Y) = \sum_{i=0}^{m} \sum_{j=0}^{n} L_i(X)L_j(Y)f(X_iY_j)$$

where

$$L_i(X) = \prod_{\substack{k=0 \\ k\neq i}}^{m} \frac{X-X_k}{X_i-X_k} \qquad i = 0,\ldots,m$$

and

$$L_j(Y) = \prod_{\substack{\ell=0 \\ \ell\neq j}}^{n} \frac{Y-Y_\ell}{Y_j-Y_\ell} \qquad j = 0,\ldots,n$$

The image registration system uses the above equations with $m = 3$, $n = 3$. Therefore, we need $m+1=4$ different X_i values and $n+1=4$ different Y_i values. The X_i's and Y_i's used are 0,1,2,3 and 0,1,2,3. Then the general equation reduces to:

$$\begin{aligned}
P_{33}(X,Y) = {} & L_0(X)L_0(Y)f(0,0)+L_1(X)L_0(Y)f(1,0)+ \\
& \quad L_2(X)L_0(Y)f(2,0)+L_3(X)L_0(Y)f(3,0)+ \\
& L_0(X)L_1(Y)f(0,1)+L_1(X)L_1(Y)f(1,1)+ \\
& \quad L_2(X)L_1(Y)f(2,1)+L_3(X)L_1(Y)f(3,1)+ \\
& L_0(X)L_2(Y)f(0,2)+L_1(X)L_2(Y)f(1,2)+ \\
& \quad L_2(X)L_2(Y)f(2,2)+L_3(X)L_2(Y)f(3,2)+ \\
& L_0(X)L_3(Y)f(0,3)+L_1(X)L_3(Y)f(1,3)+ \\
& \quad L_2(X)L_3(Y)f(2,3)+L_3(X)L_3(Y)f(3,3)
\end{aligned}$$

where:

$$L_0(X) = \frac{(X-1)(X-2)(X-3)}{(0-1)(0-2)(0-3)} = \frac{X^3 - 6X^2 + 11X - 6}{-6}$$

$$L_1(X) = \frac{(X-0)(X-2)(X-3)}{(1-0)(1-2)(1-3)} = \frac{X^3 - 5X^2 + 6X}{2}$$

$$L_2(X) = \frac{(X-0)(X-1)(X-3)}{(2-0)(2-1)(2-3)} = \frac{X^2 - 4X^2 + 3X}{-2}$$

$$L_3(X) = \frac{(X-0)(X-1)(X-2)}{(3-0)(3-1)(3-2)} = \frac{X^3 - 3X^2 + 2X}{6}$$

and $L_j(Y)$'s have the same equations with Y substituted for X

and f(X,Y) is the data value associated with pixel (X,Y).

To save computation time, the L_i's are calcualted according to the above equations for specific points in the (X,Y) grid. These points were chosen at quarter pixel intervals as shown in figure 1. The calucalted $L_i(X)$'s are;

	$L_0(X)$	$L_1(X)$	$L_2(X)$	$L_3(X)$
X = 1.00	0.0	1.0	0.0	0.0
X = 1.25	-0.0546875	0.8203125	0.2734375	-0.0390625
X = 1.50	-0.0625	0.5625	0.5625	-0.0625
X = 1.75	-0.0390625	0.2734375	0.8203125	-0,0546875
X = 2.00	0.0	0.0	1.0	0.0

The same table applies for Y=1.00, 1,25,1.50, 1.75, 2.00.

In the image registration process, an input point A (see Figure 1) is approximated to its nearest quarter pixel. To calculate the data value associated with A, the Lagrange polynomial coefficients for that quarter pixel location are used in the $P_{33}(X,Y)$ equation. To further save on computation, the products $L_i(X)L_j(Y)$ for all combinations of the quarter pixel locations ((1.0, 1.0), (1.25,1.0), (1.50,1.0), (1.75,1.0), (2.0,1.0), (1.0,1.25), (1.25,1.25), etc.) have been stored in a table. Then when $P_{33}(X,Y)$ is calculated, a table lookup locates the appropriate $L_i(X)L_j(X)$'s.

When this algorithm was implemented for cubic interpolation of data values, it was determined that the error introduced by this method of using discrete intervals versus continuous intervals was negligible. It was negligible because the intervals involved were quarter pixels and the final data values were integer values between 0 and 255.

References:

"Multitemporal Image Registrations of Multispectral LANDSAT Data of Finney and Ellis Co.'s, Kansas by Nearest-Neighbor and Third Order Lagrangian Interpolation Methods." Prepared by Charles R. Smith, LARS, September 20, 1976.

Source listing of OVERLA subroutine used in current Image Registration System.

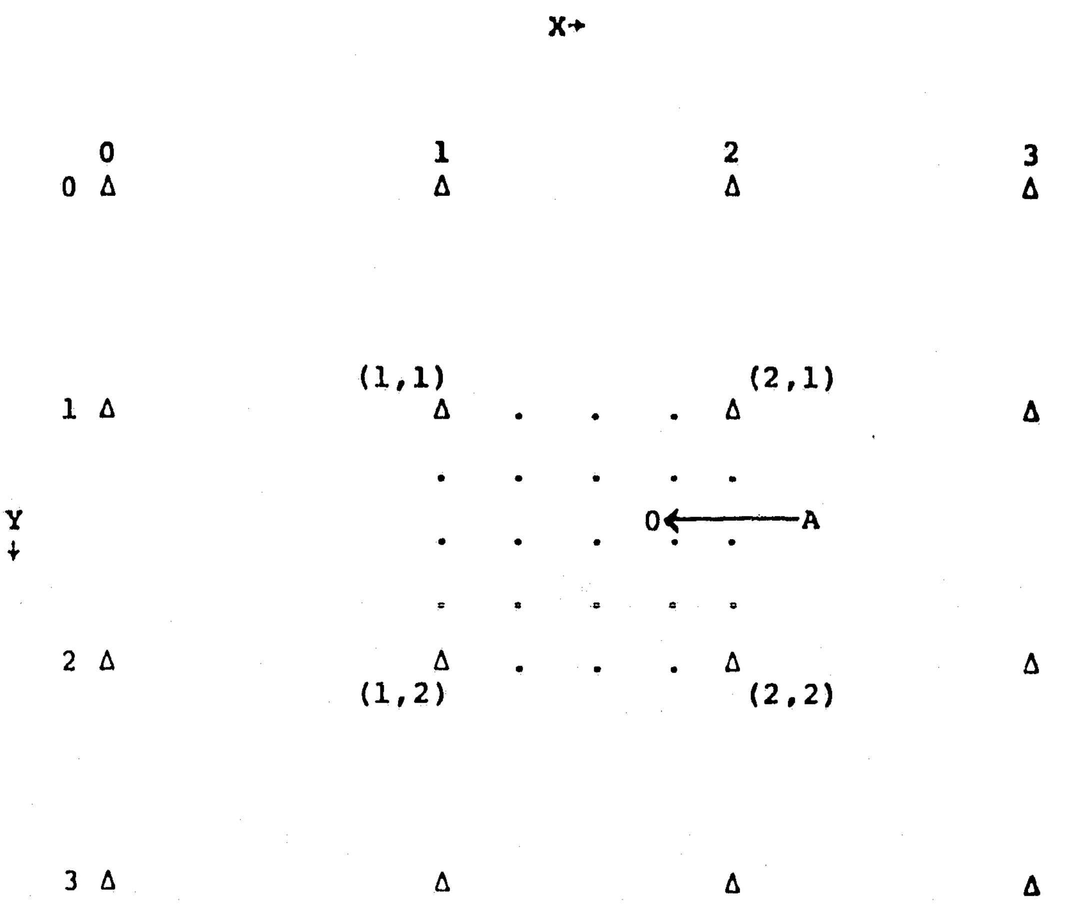


Figure 1. 4 x 4 Data matrix surrounding point to be interpolated (point A). Example: Since point A is nearest grid coordinates (1.5, 1.75), the Lagrange coefficients for this x and y are taken from the table and used in the interpolating polynomial.

APPENDIX F

Reformatting Documentation Standards

Preface

This guide supplements the LARSYS Standards Report Section III Programming Standards. Programmers writing software for the Reformatting group should read the LARSYS report as well as this guide; wherever this guide conflicts with the LARSYS report, this guide should be followed. Programmers should take particular note of the paragraphs in the LARSYS Standards Report Section III on Assembler and EXEC organizations and comments, and on programming techniques.

The main emphasis of the guide is on the documentation of program source code. Program logic must flow downward, and comments must reflect that flow. Within the source code, all global and local variables must be identified in variable description lists. The source code also must contain a general description of the algorithm used and input/output requirements. Specific coding and commenting practices are recommended for improving the legibility of source code.

This guide contains the following information.

I. DOCUMENTATION OUTSIDE OF SOURCE CODE LISTING

A. Any program with a control card reader must have a separate description of its control cards. The description must include all keywords and all parameters with an indication of which keywords and parameters are required and which are optional. All default values must be indicated. It is also useful to include one or two sample control card decks. For an example of a control card description, see Appendix A.

B. Any program designed for use by non-reformatting staff should have a user's guide. This guide should include several example user sessions.

C. Any program using routines that depend on non-trivial algorithms, calculations, or data structures must have an abstract. The abstract may be for an entire system or for specific subroutines. The abstract must describe the algorithms, calculations, and/or data structures in sufficient detail for a person unfamiliar with the source code to understand the implementation. For a major program, it may be appropriate to have two levels of documentation abstracts. One abstract would be directed at the interested user, and the other at the programmer responsible for program maintenance. For an example of a program abstract, see Appendix B.

II. DOCUMENTATION WITHIN THE SOURCE CODE LISTINGS

A. Overall System Standards

1. Each Routine must flow logically downward. See Appendix C for examples of routines that flow logically downward.

2. The names of all routines for a specific software system must have the same three-letter prefix. The last three letters should be unique for each routine and represent the main function of the routine. See the example below.

```
MEAD   - main routine for processing a MEAD product.
MEACC  - read MEAN control cards.
MEAINT - initialize MEAD variables and common
         blocks.
MEAMTX - set up MEAD scaling matrix
MEATRA - translate one line of input values into
         one line of output values.
```

Example 1

3. Use variables for constants. In the example below, constants such as Fortran unit numbers and the buffer sizes are declared as variables. Such a convention facilitates program maintenance and revision.

```
C***********************************************************************
C
C         LOCAL VARIABLES
C
C
      REAL * 4 TIME
C
      INTEGER * 4 BLANK/'    '/,     F1UNT/21/,   F2UNT/22/,   F3UNT /23/,
     1            HEX3F /Z3F/,       HEXFF /ZFF/, INBCNT /1008/,
     2            INUNT/12/,         MAXCHN/3/,   MAXIN/500/,
     3            MAXLC/10000/,      NO/'NC  '/,  CUTID(200),  CUTLNT/11/,
     4            TRK7 /'7TRK'/
C
      LOGICAL * 4 IDFLG
C
      INTEGER * 2 LARDAT(5000),      ROLL /Z7FFF/
C
      LOGICAL * 1 INBUF(1008),       ZERO/Z00/
C
C***********************************************************************
C
C            LOCAL VARIABLE DESCRIPTIONS
C     BLANK   THE CONSTANT BLANK.
C     F1UNT   DISK UNIT WHERE FIRST TAPE FILE IS TRANSFERRED.
C     F2UNT   DISK UNIT WHERE SECOND TAPE FILE IS TRANSFERRED.
C     F3UNT   DISK UNIT WHERE THRID TAPE FILE IS TRANSFERRED.
C     HEXFLG  EQUALS THE ID FLAG FOR THE INPUT TAPE (DEPENDS ON
C             FORMAT OF INPUT TAPE).
C     HEX3F   CONSTANT EQUAL TO 3F HEXIDECIMAL.  AN INPUT RECORD IS
C             AN ID RECORD IF THE FIRST BYTE EQUALS HEX 3F (7 TRACK FORMAT)
C     HEXFF   CONSTANT EQUAL TO FF HEXDECIMAL.  AN INPUT RECORD IS
C             AN ID RECORD IF THE FIRST BYTE EQUALS HEX FF (9TRACK FORMAT)
C     INBCNT  NUMBER OF BYTES IN AN INPUT RECORD.
C     INUNT   UNIT NUMBER OF INPUT TAPE.
C     MAXCHN  MAXIMUM NUMBER OF DATA CHANNELS THIS ROUTINE CAN HANDLE
C     MAXIN   MAXIMUM NUMBER OF DATA VALUES IN ONE INPUT RECORD.
C     MAXLC   MAXIMUM NUMBER OF BYTES ALLOWED IN ONE LINE OF LARSYS DATA.
C     OUTUNT  UNIT NUMBER FOR OUTPUT TAPE.
C     NC      CONSTANT EQUAL TO 'NC'.
C     TRK7    CONSTANT EQUAL TO '7TRK'.
C     ZERO    CONSTANT BYTE EQUAL TO 00 HEXIDECIMAL.
C
```

Example 2

In the above example, local variable descriptions have been provided only for the "constant" variables. See the example software system in Appendix C for descriptions of all local variables.

4. Block commons must be named and they must have variables listed in the order:

   REAL * 8
   REAL * 4
   INTEGER * 4
   LOGICAL * 4
   INTEGER * 2
   LOGICAL * 1

Within each type of variable, the variables must be listed alphabetically. Large common blocks must be spaced for legibility.

```
C     VARIABLE NAMES FOR AGRONOMIC ID AND FOR SOILS IS
C
         COMMON /IDNAME/  AITE,     AZIR,     AZVI,     BAPR,     BRLE
         COMMON /IDNAME/  CARU,     CATN,     CLCO,     CLTY(4),  COB2
         COMMON /IDNAME/  COMM(37),DACO,     DADA,     DAPL,     DBBL
         COMMON /IDNAME/  DBFR,     DBGL
         COMMON /IDNAME/  DBST,     DBTO,     DBWE,     DBVL,     DEEQ
         COMMON /IDNAME/  DENA(2),  DERA,     DIGR,     DIIN,     DQF1(2)
         COMMON /IDNAME/  DQF2(2),  DQF3(2),  DQF4(2),  DQF5(2),  DQF6(2)
         COMMON /IDNAME/  DQF7(2),  DRCL,     EP01,     EP02,     EP03
         COMMON /IDNAME/  EP04,     EP05,     EP06,     EP07,     EP08
         COMMON /IDNAME/  EP09,     EP10,     EXNA(4),  EXNU,     FANA(4)
         COMMON /IDNAME/  FIAC,     FINU,     FIVI,     FLLI(2),  FOCA
         COMMON /IDNAME/  FRBI,     FRCO,     GMCS,     GRLE,     HAWI
         COMMON /XDNAME/  HEIG,     HERF,     HISC,     HORI(2),  ILLU(2)
         COMMON /IDNAME/  ININ,     INNA(4),  INST,     JUDA
         COMMON /IDNAME/  LAID,     LEAR,     LEPL,     LF(6),    LF7
         COMMON /IDNAME/  LF8,      LOCA(4),  LODA,     LOLA(2),  LOLO(2)
         COMMON /IDNAME/  LOSQ,     MATU(4),  MODA,     MOFI(4),  MOLA
         COMMON /IDNAME/  MOST,     MUCO(4),  NMAT,     NUDE,     NUSA
         COMMON /IDNAME/  NUSG,     OBNU,     OTST,     PECL,     PEGR
         COMMON /IDNAME/  PESA,     PESI,     PHFR(2),  PHRO,     PHSE(4)
         COMMON /IDNAME/  PLCO,     PLDA,     PLMO,     PLNU,     PMOW
         COMMON /IDNAME/  PRIN(4)   RATE
         COMMON /IDNAME/  RECA,     REDA,     REHU,     RENU,     RODI
         COMMON /IDNAME/  ROWI,     RUSE,     SAGR,     SCRA,     SCTY(4)
         COMMON /IDNAME/  SENA(4),  SENU,     SPEC(4),  STCO(10),SUCO(4)
         COMMON /IDNAME/  TALE,     TATE,     TAWI,     TCR1,     TCR2
         COMMON /IDNAME/  TEXT(4),  TIDA,     TSWT,     UNCA,     VARI(4)
         COMMON /IDNAME/  VI
         COMMON /IDNAME/  WABA(4),  WBTE,     WEED,     WIDI,     WISP
         COMMON /IDNAME/  YEDA,     YELD,     YELE,     ZEIR,     ZEVI
C
C        VARIABLE NAMES EXCLUSIVELY FOR SOILS ID
C
         COMMON /IDNAME/  ACTI,     ALUM,     ASHO(2),  AVPH,     AVPC
         COMMON /IDNAME/  BASA,     BUDE,     BUFH,     CAEX,     CALC
         COMMON /IDNAME/  CHRO,     CLAY,     COCO,     COIN,     COPA
         COMMON /IDNAME/  COSA,     COSI,     CSNU,     ELCO,     ELNU
         COMMON /IDNAME/  EP11,     EP12,     EP13,     ERFA,     EROS
         COMMON /IDNAME/  EXAC,     FINE,     FISA,     FISI,     FSAN
         COMMON /IDNAME/  GRGR(2),  HUE1,     HUE2,     IRON,     LIIN
         COMMON /IDNAME/  LILI,     LISH,     LOF(6),   MAGN,     MANG
         COMMON /IDNAME/  MESA,     MICL,     MOTE,     MOZC(2),  MSAN
         COMMON /IDNAME/  MSNU,     OMOD,     ORCA,     ORDR,     PAMA
         COMMON /IDNAME/  PASI,     PHYS,     PLIN,     PLLI,     PCTA
         COMMON /IDNAME/  SAND,     SAPO,     SHLI,     SHRA,     SILI
         COMMON /IDNAME/  SILT
         COMMON /IDNAME/  SLOP,     SODI,     SOEL,     SPGR,     STAB
         COMMON /IDNAME/  STLN,     SUBO,     SUDE(10),SUNA(4),  TERE(2)
         COMMON /IDNAME/  UNIF,     VALU,     VCSA,     VFSA,     VCSH
         COMMON /IDNAME/  WACO,     WAPH,     WIER,     YEAR
C
C     INTERMEDIATE VARIABLES USED IN CALCULATION OF ID VALUES
C
C        REAL * 4 VARIABLES
C
         COMMON /IDNAME/  METROW,   XFRCO,    XPLCO
C
C        INTEGER * 4  VARIABLES
C
         COMMON /IDNAME/  AGOC,     BIOPLT,   FWR,      GRG,      HEAD
         COMMON /IDNAME/  MODE,     MOZ,      ORD,      SIDE,     SUBR
         COMMON /IDNAME/  SURFST,   SWING,    TEMR,     TEX,      INTBUF(50)
C
C
         REAL * 4  METROW, XFRCO, XPLCO
```

Example 3

Although the common block above does not list the variables in the order REAL, INTEGER, LOGICAL, it is a good example of spacing for legibility. (The variables are arranged by usage in this common block).

Common block variables must be described in a BLOCK DATA routine or in an initialization subroutine. The variable descriptions must be alphabetic. See Appendix D for an example.

5. Do not use Fortran entry points unless the use of them is clearly the best solution to an implementation problem.

6. Information and error messages should be informative to the user as well as the programmer. Each message must include the name of the routine printing the message.

```
C               *ELSE   ERROR
 4200              OK = .FALSE.
                   WRITE (TYPWTR, 94210) I, ID(STRESS(I))
                   WRITE (PRNTR, 94210) I, ID(STRESS(I))
94210              FORMAT(' E00A0 STRESS(', I2, ') DOES NOT EQUAL',
     1                ' N, Y, OR BLANK. ',
     2             ' INSTEAD IS SET TO (', A4, ')',
     3             4X, '(XTKA)')
```

Example 4

It may be numbered either sequentially (1 to n) or for the labeled Fortran statement nearest the message in the code. In the example above the message is numbered sequentially. In the example software system in Appendix C the messages are numbered for labeled Fortran statements.

7. Labels for code statements must be assigned in ascending order within the body of each routine. For examples see Appendix C.

8. Labels for FORMAT statements must be assigned in ascending order within the body of each routine. The FORMAT labels should be sufficiently different from the code labels that they stand out. For example, code labels in a routine could range from 100 to 900 and FORMAT labels from 9100 to 9900. The FORMAT statements may be interspersed with the executable code or they may be just before the END statement. However, within one routine, they must be either all interspersed or all at the end. The software system shown in Appendix C is an example of FORMAT statements interspersed with executable code.

9. Do not use unnecessary EQUIVALENCE statements. However, there are some data structures for which EQUIVALENCE statements are necessary. For example, a LARSYS ID record contains real data values and integer data values. In order to correctly access both data types, the ID record must be declared as:

```
REAL * 4 RID(2000)
INTEGER * 4 ID(200)
EQUIVALENCE (ID(1), RID(1))
```

10. Use standard LARSYS and Reformatting routines whenever possible. For example, often used LARSYS routines are CTLWRD and BCDVAL (for interpreting control cards), and often used Reformatting routines are IDRITE and EOT (for mounting LARSYS data tapes, writing ID records, and writing end-of-tape records).

11. Document all revisions to routines by adding your name and date to the comments. Include a version number if appropriate. If the revision is appropriate for only a special application, add a comment near the revision comment stating exactly what the special applications is.

```
C
C    WRITTEN 07/19/79 BY CATHERINE KOZLOWSKI FOR FY70
C                                 SR&T CONTRACT
C
C    REVISED 11/20/79 BY CATHERINE KOZLOWSKI FOR FY79
C                                 SR&T CONTRACT
C
```

Example 5

12. Indent (horizontal) and space (vertical) the source code to improve readability and/or logical flow of each routine. See the software system in Appendix C for examples.

13. When reading or writing a long string of variables, space the variable names the same in the READ/WRITE statement as in the FORMAT statement.

```
C
C***********************************************************************
C
C READ AGRONOMIC RECORD SHEET NUMBER 2 OF THE GROUP OF 7
C
C***********************************************************************
  200     READ  (AGSHE, 9200, END=110)
     1       AGOC2, PAGE2, RID(HEIG), RID(LEPL), ID(GRLE),
     2       ID(YELE), ID(BRLE), XPLCO, XFRCO,
     3       METROW, ID(PEGR), RID(DBGL), RID(DBYL),
     4       RID(DBBL), RID(DBST), RID(DBFR), RID(DBTO),
     5       RID(FRBI), BIOPLT, RID(LEAR), ID(PLMO)
C
 9200     FORMAT(I3, I1, F3.0, 1X, F4.1, 1X, 3I2,
     1         F4.0,F3.0,
     2        F4.0, 1X, I2, 1X, 5F5.0,
     3        2F6.0, I1, F5.0, 1X, I2)
C
           PTRA  = PTRA - 1
           READ(AGSHE, 9201, END=110)
     1         BLKID(HEIG), BLKID(LEPL), BLKID(GRLE), BLKID(YELE),
     2         BLKID(BRLE), BLKID(PLCO), BLKID(FRCO), BLKID(PEGR),
     3         BLKID(DBGL), BLKID(DBYL), BLKID(DBBL), BLKID(DBST),
     4         BLKID(DBFR), BLKID(DBTO), BLKID(FRBI), BLKID(LEAR),
     5         BLKID(PLMO)
C
 9201      FORMAT(T5, A3, 1X, A2, T14, 3A2, 1X, 2A3, T32, A2,
     1             1X, 5(1X,A4), 2(2X,A4), T76, A2, 1X, A2)
C
```

Example 6

14. If possible, use the following convention for FILEDEFing and assigning tape units:

```
FILEDEF 11 TAP1
FILEDEF 12 TAP2
FILEDEF 13 TAP3
FILEDEF 14 TAP4
FILEDEF 10 TAP5
```

where Fortran unit 11 is the output tape and units 12-14, 10 are input tapes.

ORIGINAL PAGE IS
OF POOR QUALITY

15. Several suggestions about labels and CONTINUE statements:

    a. It is easier to revise routine if each DO loop has its own CONTINUE statement.

```
      DO 120 K = 1, 20
         DO 100 J = 1, 3
            ARRAY(J,K) = J + K
100      CONTINUE
126   CONTINUE
```

Example 7

    b. It is easier to revise a routine if all of its non-FORMAT labels are on CONTINUE statements.

16. Debugging convention

C-2

B. Layout of Individual Routines

```
C   routine name
C
C*****************************************************************
C   routine name  one-line description
C
C   WRITTEN date BY name FOR CONTRACT name or number
C   REVISED data BY name
C
      SUBROUTINE name
C
      IMPLICIT INTEGER * 4 (A ~ Z)
C
C   detailed description
C
C   special features and/or limitations
C
C   input
C
C   output
C
C   subroutines used (include one-line description of
      each subroutine)
C
      COMMON /name/ declarations
                  .
                  .
                  .
      COMMON / name/ declarations
C
C*****************************************************************
C
C         LOCAL VARIABLES
C
      local variables declarations by type, then alphabetic
         (include parameters as necessary)
C
C   local variable descriptions including parameters, listed
       alphabetically
C
      body of routine
      END
```

Example 8

All routines should follow the general format outlined above. See Appendix C for a complete system following this layout.

1. The first several lines of the source code should identify the routine.

```
C    SPCSCN
C
C*************************************************************************
C
C    SPCSCN   REFCRMATS CNE SPECSCAN TAPE TO CNE LARSYS RUN.
C
C    WRITTEN 02/14/79 BY CATHERINE KOZLOWSKI    FCR SRET FY79 CCNTRACT
C    REVISED C4/04/79 BY CATHERINE KCZLOWSKI
C    REVISED C7/02/79 BY CATHERINE KOZLOWSKI
C
C*************************************************************************
C
```

Example 9

2. After the IMPLICIT INTEGER * 4 statement, there should be a detailed description of the routine.

```
C
C    SPCSCN
C       THIS PROGRAM AND ITS SUBROUTINES REFCRMAT A 7-TRACK (MODE 3)
C    OR 9-TRACK 8CO BPI SPECSCAN TAPE TO A 9-TRACK 1600 BPI LARSYS
C    DATA RUN.  THE CRIGINAL SPECSCAN TAPE FCR WHICH THIS SCFTWARE WAS
C    WRITTEN WAS RECEIVED FROM RCBERT A. GOODING, TECHNICCLCR GRAPHIC
C    SERVICES INC. (TEXAS OPERATICNS, LYNDCN B. JCHNSCN SPACE CENTER,
C    P.O. BOX 58863, HOUSTON, TEXAS 77C58).
C    THE INPUT TAPE HAS CNE OR MCRE FILES, EACH FILE CCRRESPCNDING TO 1
C    CHANNEL OF DATA ON THE LARSYS TAPE.  ALL RECCRDS CN THE INPUT TAPE
C    ARE 1008 BYTES LONG.  THE FIRST INPUT RECORD OF EACH FILE MAY
C    BE AN ID RECCRD -- IF IT IS, THE FIRST BYTE EQUALS HEXIDECIMAL
C    '3F' OR 'FF' AND THE RECORD IS SKIPPED DURING PRCCESSING OF DATA.
C    IT IS ASSUMED EACH FILE HAS THE SAME NUMBER  OF RECORDS AND,
C    IF ONE FILE HAS AN ID, THEY ALL HAVE ID RECCRDS.
C    IT IS ALSO ASSUMED THAT THE ONLY SIZE OF CNE SPECSCAN INPUT
C    RECORD IS 10C8 BYTES.  HOWEVER, ONE LINE OF SPECSCAN INPUT MAY
C    BE SEVERAL INPUT RECORDS LCNG.
                          CONTINUE
C    THE PROGRAM REQUIRES ONE TEMPORARY DISK AND TWO TAPE DRIVES.
C    THE EXEC CALCULATES THE AMOUNT OF TEMP SPACE NEEDED FOR THE DATA
C    TO BE TRANSFERRED.  THE INPUT TAPE DRIVE IS ASSIGNED UNIT NUMBER 12
C    IT MAY BE A 7 OR 9 TRACK DRIVE.  THE OUTPUT TAPE DRIVE IS ASSIGNED
C    UNIT NUMBER 11  IT IS A 9 TRACK DRIVE.
                          CONTINUE
C    THE PROGRAM FIRST TRANSFERS THE INPUT TAPE FILES TO DISK.  THEN
C    IT SETS UP THE LARSYS ID RECORD.  INPUT DATA IS REFCRMATTED LINE
C    BY LINE.  EACH INPUT LINE GOES THROUGH THE FOLLOWING PROCESSINGO
C       1)  INPUT FOR CHANNEL N IS READ FROM DISK N INTO A TEMP BUFFER.
C       2)  THE DATA IN THE TEMP BUFFER IS REFORMATTED FROM 2 BYTES
C           PER VALUE TO 1 BYTE PER VALUE.  THE DATA IS INVERTED (BLACK
C           TO WHITE) AT THIS TIME.  IF NECESSARY, THE DATA IS ALSO
C           FLIPPED LEFT TO RIGHT.
C       3)  THE DATA IS MOVED TO A LARSYS-FORMAT DATA LINE
C           AND WRITEN TO TAPE.
```

Example 10

In the above example, special features and limitations of the routine have been noted. Special features are 1) the input can be on either a 7-track or 9-track tape, and 2) the data can be flipped left to right. Limitations are 1) if one input file has an id record, all input files must have ids, and 2) the routine requires two tape drives and one temporary disk.

3. Input requirements must be specified.

```
C        THE INPUT IS AN ARRAY OF DATA VALUES IN SPECSCAN TAPE
C        FORMAT.
C        EACH DATA VALUE IS ASSUMED TO BE ONE 3-BIT FIELD IN
C        AN 8-BIT BYTE AND ONE 6-BIT FIELD IN AN 8-BIT BYTE,
C        THESE TWO FIELDS REPRESENTING ONE DATA VALUE RANGING
C        FROM 0 TO 511.
```

Example 11

4. Output from a routine must be described.

```
C        THE OUTPUT IS AN ARRAY OF DATA VALUES WITH EACH 2 BYTES
C        REPRESENTING ONE 8-BIT DATA VALUE (THE FIRST BYTE IS SET TO ZERO
C        AND THE SECOND BYTE CONTAINS THE DATA).  OUTPUT VALUES RANGE
C        BETWEEN 0 AND 255.
```

Example 12

5. The source listing must include all non-system subroutines called.

```
C     THE NON-SYSTEM SUBROUTINES USED ARE:
C           EOT      WRITES END-OF-TAPE RECORD ON OUTPUT TAPE.
C           GTDATE   RETURNS TODAYS'S DATE IN CHARACTER FORMAT.
C           IDRITE   MOUNTS OUTPUT TAPE AND WRITES OUTPUT RUN ID RECORD.
C           MOUNT    MOUNTS INPUT TAPE.
C           MOVBYT   MOVES BYTES FROM INPUT BUFFER TO OUTPUT BUFFER.
C           SPCDAT   TRANSLATES DATA FROM A 9-BIT FORMAT TO AN 8-BIT
C                    FORMAT.
C           SPCSAM   CALCULATES THE NUMBER OF SAMPLES PER CHANNEL IN THE
C                    OUTPUT.
C           TAPOP (ENTRY POINTS TOPEF, TCPFF,
C                    TOPRD, TOPWR)  PERFORMS TAPE I/O FUNCTIONS.
C
C
C****************************************************************
C
C
```

Example 13

6. All local variables must be declared (as necessary) and described.

```
C*****************************************************************************
C
      INTEGER * 4 BLFCNT /5C4/, MAXDAT /500/, NCNDAT /4/, RLIMIT /5/,
     1      ZERO /0/
C
      INTEGER * 2 BUFFER(504)
C
      LOGICAL * 4 IIFLG
C
      LOGICAL * 1 INBUF(10C8)
C
      EQUIVALENCE (BUFFER, INBUF)
C
C*****************************************************************************
C
C                    LOCAL VARIABLE DESCRIPTIONS
C
C    ACJUST TEMPORARY VARIABLE USED TO ACJUST SAMPLE CCUNT TC BE EVENLY
C           DIVISIBLE BY 4.
C    BUFCNT NUMBER OF ELEMENTS IN INPUT BUFFER.
C    BUFFER INPUT BUFFER IN INTEGER * 2 FORMAT.
C    DISP   DISPLACEMENT INTO INPUT BUFFER.
C    ICFLG  FLAG INDICATIANG WHETHER FIRST INPUT RECORD OF THE FILE IS
C           AN ID RECORD.  IF ICFLG IS SET, FIRST RECCRD IS AN ID.
C    INBUF  INPUT BUFFER IN LOGICAL * 1 FORMAT.
C    LSTVAL LAST CATA VALUE IN INPUT BUFFER.
C    MAXDAT MAXIMUM NUMBER OF DATA VALUES POSSIBLE IN CNE INPUT RECORD.
                                 CONTINUE
C    NCNCAT NUMBER OF NON-DATA VALUES IN INPUT BUFFER.
C    NREAD  NUMBER OF CONSECUTIVE RECORDS READ WHEN SEARCH FCR ZERO DATA
C           VALUES.
C    NSAMP  NUMBER OF SAMPLES PER CHANNEL THAT WILL BE IN LARSYS CUTPUT.
C    PREVAL PREVICUS DATA VALUE IN INPUT BUFFER.  USED TC SEARCH
C           BACKWARDS IN INPUT RECORD.
C    RLIMIT UPPER LIMIT ON THE NUMBER OF CONSECUTIVE READS TC PERFORM
C           BEFORE TERMINATING SEARCH FOR ZERC DATA VALUES.
C    TOTAL  RUNNING TOTAL USED TC CALCULATE NUMBER OF SAMPLES.
C    UNIT   DISK UNIT FROM WHICH TO READ INPUT RECORDS.
C    ZERO   THE CCNSTANT ZERO.
C
C*****************************************************************************
C*****************************************************************************
C
```

Example 14

C. Comments Within The Body of a Routine

1. Highlight comments that describe large sections of code. See Appendix C for examples.

2. Comments by themselves should describe the flow of the routine in sufficient detail so a reader can understand the routine without looking at the code.

3. Inobvious programming "tricks" must be explained in detail including the reason for the trick.

4. Specific suggestions:

   a. Comment a control card computed GO TO so that it is apparent which label corresponds to which key word.

```
      IMPLICIT INTEGER * 4 (A - Z)
C
      INTEGER * 4 KEYLST(7)
C
      DATA KEYLST / '*INP', 'REFO', 'INPL', 'SCRA', 'OUTP',
     1              'END',  '-COM'/
C
      DATA KEYSZ /7/
C
C
C
      CALL CTLWRD(CARD, COL, KEYLST, KEYSZ, CODE, READIN, ERROR)
C
C              *INP   REFO   INPL   SCRA   OUTP   END    -COM
      GOTO (1000,  2000,  3000,  4000,  5000,  6000,  7000), CODE
C
 1000 CONTINUE
 2000 CONTINUE
 3000 CONTINUE
 4000 CONTINUE
 5000 CONTINUE
 6000 CONTINUE
 7000 CONTINUE
      STOP
      END
```

Example 15

b. Comment logical program structures with statements such as:

```
C
C   WHILE NOT END-OF-FILE PROCESS DATA
C

C
C   REPEAT LINE PROCESSING UNTIL END-OF-FILE
C

C
C   IF GOOD DATA THEN PROCESS IT
C                ELSE PRINT ERROR MESSAGE
C
```

APPENDIX G

A BEGINNING GUIDE TO OPERATING LARSYS

July 1979

Developed for

St. Regis Paper Company
Southern Timberlands Division
Jacksonville, Florida

by

Susan Schwingendorf
Bud Goodrick

Laboratory for Applications of Remote Sensing
Purdue University
West Lafayette, Indiana 47906

Prepared for

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Johnson Space Center
Earth Observation Division
Houston, Texas 77058

Contract: NAS 9-15325

Technical Monitor: R.E. Joosten/SF5

TABLE OF CONTENTS

INTRODUCTION

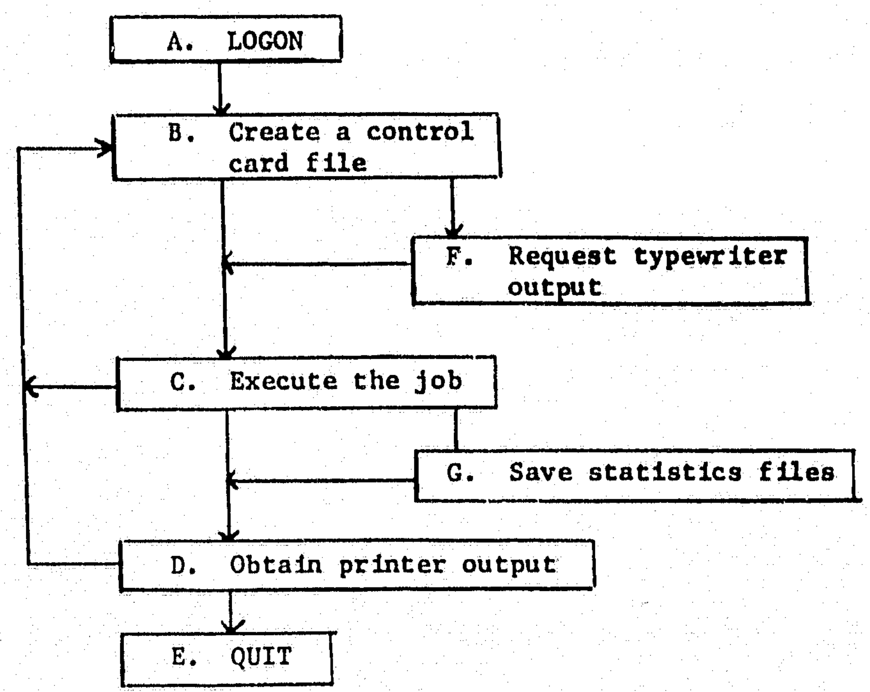
This guide is intended to serve as a handy reference for new (or infrequent) users of the LARSYS software available at Purdue/LARS, and has been divided into two sections. The first section describes the sequence of commands required to access the Purdue/LARS computer, create input files for the LARSYS processors, execute the LARSYS job and receive printer output. Section II briefly discusses the LARSYS processors used in an analysis sequence, with examples of setting up the control card files and executing the jobs.

Before using this guide, the new LARSYS user should witness a demonstration of the terminal equipment, the procedure for accessing the Purdue/LARS computer and the steps involved in executing a LARSYS function. This person should also know who is available at his location to answer questions and assist with unexpected problems.

The new analyst should also have some background in remote sensing, such as is presented in the monthly LARS Short Course on "Numerical Analysis of Remote Sensing Data" or minimally, by reading the LARS Information Note 110474, "An Introduction to Quantitative Remote Sensing."

This may be followed by an overview of the LARSYS processors and their capabilities. Brief descriptions can be found in the "LARSYS User's Manual," Volume 1; Sections 1, and 2.1 through 2.4. More detailed descriptions of each processor are located in Volume 2 of the "LARSYS User's Manual" and may be reviewed as the processor is used in the analysis sequence.

The general steps required to access the Purdue/LARS computer and execute a LARSYS processor are flowcharted below. The letters refer to the part in Section I where that step is discussed, not the required sequence.



Section I

Procedures for Accessing LARSYS

A. LOGON AND INITIATE THE LARSYS SYSTEM

In order to access the LARSYS system, it is necessary to have a computer ID and a password. If you do not know what computer ID you are to use, check with _________________. The initial procedure for accessing the computer at Purdue/LARS from a typewriter terminal is called "logging-on" which is illustrated below. Before logging on locate the Attention(ATTN) or Break (BRK or BREAK) key on your terminal. Pressing this key signals to the computer you are ready to gain access to the system. Also locate the RETURN (or CR) key, which you will press each time you have completed typing a line. The command "i lsdv370" initiates the LARSYS system.

Lines to be entered by the user are outlined with boxes. The > means the keyboard is unlocked and waiting for user input.

```
VM/370 ONLINE  <------------  (Press BREAK key)
!
>logon jax  <------------  (Substitute your ID for 'jax')
ENTER PASSWORD:
>XXXXXXXX
ENTER NAME:>schwingendorf
LOGMSG - 08:37:18 EST MONDAY 07/09/79
* YOUR OPERATORS THIS MORNING ARE ROSS AIKEN AND GREG RICHARDSON
LOGON AT 09:48:54 EST MONDAY 07/09/79
>i lsdv370
DEVELOPMENTAL LARSYS READY:
  SYSTEM IS BEING INITIALIZED...

... LARSYS IS READY FOR YOUR FIRST COMMAND
T=0.40/1.07 09:49:48
```

At this point (after a message beginning with "T=" and a ">") the system will accept any of the LARSYS commands listed in the reference file listing at the back of this notebook. We will discuss the ones you will use most often.

Note: Character delete - If you happen to make a typing error, you can type the @ to "erase" the previous character(s)

e.g. i lad@@sdv370

would be interpreted by the computer as

i lsdv370

Line delete - If you need to "erase" an entire line, use the [and the computer will ignore what you have already typed on that line.

CP begin - If the letters CP are typed unexpectedly, you will need to type "begin". This can happen if you try to type a command before the ">" appears or if there is noise on the line.

B. PREPARE CONTROL CARD FILES

To use LARSYS, a person must first decide which LARSYS processor is to be executed, and then create a control card file containing the necessary information for that processor. To help the user know what to put into a control card file, LARSYS maintains lists or menus (called REFERENCE Files) of all the possible control parameters for each processor. A composite list of all processors, including initialization functions and control commands, also exists. For your convenience the list has been printed and included in this notebook. These control card files are "shown" to the LARSYS software system on cards, from disk files, or interactively from a typewriter terminal. In these examples, we will make use of disk files.

Associated with your computer ID is a private disk area on which you can store one or more files of information, such as control card files. Before discussing the usual processors used in an analysis project, we will look at how you can enter a control card file onto your disk storage area. You will repeat this process for each LARSYS processor you run.

A typical sequence, then, might be as follows: 1) determine the desired function, such as wanting to produce a grayscale map, 2) from experience or by reviewing the function of each processor in the "LARSYS User's Manual", determine the processor that can perform the desired function, (for example PICTUREPRINT will produce a grayscale map), 3) by reviewing the control card list (provided at the back of the notebook) for the selected processor, write out the keywords and control parameters necessary to execute the processor, and 4) enter the keywords and control parameters into a disk file. Now let's look at how to obtain and enter parameters into a control card file.

GET Having completed the steps from Section I for logging-on, we select the LARSYS processor IDPRINT, to execute. This processor produces a one-page listing of identification information for your data. Use the GET command and the first three letters of the processor's name, (in this case IDP) to obtain a skeleton control file for IDPRINT.

```
>get idp
THE FILE -- IDP CC -- HAS BEEN COPIED TO YOUR PRIVATE DISK.
IT'S CONTENTS ARE:

-RUNTABLE
DATA
RUN(XX), TAPE(YY), FILE(ZZ)
END
*IDPRINT
PRINT RUN(XX)
END

EDIT:
```

(A file has two names - only the first was needed for the GET command. See Note at end of this section.)

EDIT As you can see from the above messages, the file named IDP CC is copied onto your disk storage area, its contents are typed, and your computer ID enters the Edit environment. This means that you can enter a variety of Edit commands which will add, change or delete lines from this control card file.

In order to make changes to the file, the user must move to the line to be changed. Four Edit commands help you position yourself in the file. These are:

TOP	*(point to the Top of the file)*
BOTTOM	*(point to the last line of the file)*
UP n	*(move up 'n' lines in the file)*
NEXT n	*(move down 'n' lines in the file)*

TOP points you to the place before the first line of the file, BOTTOM moves the pointer to the last line of the file and UP and NEXT move the pointer one or more lines within the file. When Edit is first entered, you are at the spot just before the first line of the file. If you type NEXT you will be at the first line of the file. Note that the computer types out the line you are on.

```
>next
-RUNTABLE
>next 2
RUN(XX), TAPE(YY), FILE(ZZ)
>up 2
-RUNTABLE
```

Now let's look at some commands to make changes to the file. Four commands you can use are:

DELETE n	*(delete 'n' lines from the file)*
CHANGE .XXX.YYY.	*(Change the letters 'XXX' to the letters 'YYY' in current line)*
REPLACE 'new line'	*(replace current line with 'new line')*
INPUT 'new line'	*(insert 'new line' after the current line)*

One other command, TYPE allows us to verify which line we are on.

TYPE	*(type the current line of the file)*

Remember that we are currently at the first line of the file, which is the line -RUNTABLE. We decide that the first four lines of the file are not needed and type DELETE 4. Using the TYPE command reassures us we are no longer at the line -RUNTABLE (because it was removed from the file) and are at the new first line of the file, *IDPRINT.

```
>delete 4
>type
*IDPRINT
```

We proceed to the next line of the file and use the CHANGE command to supply a run number in place of 'XX'.

```
>next
PRINT RUN(XX)
>change .xx.76020106.
PRINT RUN(76020106)
```

We decide to input a comment line at the beginning (TOP) of the file using the INPUT command. (TOF stands for Top Of File)

```
>top
TOF:
>input  -comment test comment line
```

To replace this entire line with a new comment, the REPLACE command can be used.

```
>replace -comment print id record for fargo data
```

Now we can check the contents of the file by going back to the top and typing all the lines.

```
>top
TOF:
>type 10
TOF:
-COMMENT PRINT ID RECORD FOR FARGO DATA
*IDPRINT
PRINT RUN(76020106)
END
EOF:
```

When the Edit session has been completed there are two Edit commands for returning to LARSYS. They are FILE and QUIT.

FILE	*(save the current file on disk)*
FILE 'name'	*(save the current file on disk with the new name 'name')*
QUIT	*(don't save the changes made during the current Edit session)*

To save our IDPRINT control card file, we can type

```
>file
T=0.50/4.79 10:09:18
```

and the file IDP CC will be stored on our computer ID with all the changes which have been made. This completes the Edit session and returns us to the LARSYS Environment. If we wanted to make further changes to this same file in the future the EDIT command should be used instead of the GET command. The format of this command is:

EDIT name1 name2

```
>edit idp cc
      (Issue any Edit commands to make changes desired)
>file
T=0.10/0.45 11:06:40
```

<u>Note on disk files</u>: All files which you store on your private disk have two names which identify the file to the computer. (In the above example the file was IDP CC). You must remember this name for later commands. It is generally easier to remember the names if you establish a naming pattern (such as always making the second name of a control card file CC or DECK, and making the first name an abbreviation of the analysis area location or processor name). The names may consist of up to 8 alphanumeric characters.

A QUICK REFERENCE TO EDIT COMMANDS

Edit name1 name2	
Bottom	*(move to the last line of the file)*
Change /string1/string2/	*(change 'string1' to 'string2' in this line, where a string is a sequence of characters.)*
DELete n	*(delete 'n' lines from the file, starting with the current line. If 'n' is omitted, the current line is deleted.)*
Getfile name1 name2	*(insert the file 'name1 name2' after the current line)*
Input new-line	*(insert line 'new-line' after the current line)*
Next n	*(move down 'n' lines in the file. If 'n' is omitted, move to the next line.)*
Replace new-line	*(replace the current line with 'new-line')*
TOP	*(move to the top of the file)*
Type n	*(type 'n' lines, starting with the current line. If 'n' is omitted the current line is typed.)*
Up n	*(move up 'n' lines in the file. If 'n' is omitted, move up 1 line.)*
FILE	*(store the current file, with all changes made, on the private disk and return to LARSYS)*
QUIT	*(stop editting the current file without storing changes which have been made)*

Note: Once you are familiar with the Edit commands, it is usually faster to use the abbreviations for the commands. These are indicated above by the letters which are capitalized and underlined. Hence, the abbreviation for 'delete' is 'del' and the abbreviation for 'next' is 'n'.

C. EXECUTE THE LARSYS JOB

To run a LARSYS job, the control cards must be passed to the computer by reading them into a card reader or by telling the computer which disk file contains the needed control cards, using the CCINPUT command. The format of the CCINPUT command is

CCINPUT CCINPUT name1 name2

```
>ccinput idp cc
 T=0.11/0.60 10:09:32
```

You must know both names of your control card file, in this case IDP CC. In case you have forgotten the name(s) of your control card file(s), you may get a list of file names with the LISTFILE command.

```
>listfile * cc
```

This indicates you want to list all files on your private disk with a second name of "cc". If you have forgotten the contents of a file, use the TYPE command, supplying the two names of the file.

```
>type idp cc
```

Here, the contents of the file IDP CC will be typed.

RUN Next type:

```
>run
 EXECUTION BEGINS...
  I0065  IDPRINT FUNCTION HAS BEEN REQUESTED.  (RUNSUP)
  I0114  IDPRINT FUNCTION COMPLETED.  (RUNSUP)
  I0103  CPU TIME USED WAS    6.148 SECONDS.  (LARSMN)
  I0004  END OF INPUT DECK - RUN COMPLETED  (LARSMN)
  I0050  TOTAL CPU TIME FOR THIS RUN WAS  6.288 SECONDS.  (LARSMN)
 PRT FILE 6458  TO RSCS          COPY 01 NOHOLD
 T=2.06/10.17  10:10:25
```

Notice that LARSYS prints a series of informational messages indicating when the function begins processing and ends. A print file is created as normal output from a LARSYS processor and procedures for obtaining the printer output from your site can be found in the next section.

You have now completed the general sequence for running a LARSYS job. The next step is to review your printer output and decide if you need to run any more LARSYS jobs. If so, go back to Section B, and create a new control card file. Then, using the name of this new file, enter the CCINPUT command and the RUN command. When you have executed the last LARSYS processor, proceed to Section E.

Note on halting execution: To terminate a job which is executing, press the BREAK key (computer types 'CP'), and type 'i lsdv370' to re-initiate the system.

D. OBTAIN PRINTER OUTPUT AT JACKSONVILLE

The steps for receiving printer output from LARS are as follows:

1. On the DECWRITER terminal type:

    m cp please start stregis

    (or call LARS computer room and ask them to check whether STREGIS is STARTED. Call: 317-749-2052 and ask for computer operator)

2. Flip the LARS/DALLAS switch to position A

Move to the IBM 3776 terminal for the following steps:

3. Flip the reset switch on the IBM 3776

4. Enter "local mode" (press "code" and "start job")

5. To get 8 Lines Per Inch (LPI), press "code" and "K"

6. Press: "start job", "s", "3", "0", "EOM"
    (NPR should display 320)

7. Put the SIGNON card in the reader and press START

8. Receive print files.

9. After all print files have been received, put the DRAIN card in the card reader, press START, and then flip the LARS/DALLAS switch back to B.

E. TERMINATE THE LARSYS SESSION

QUIT When you have completed your work for the current terminal session, the QUIT command is used to let the computer know you are finished.

```
>quit
YOU TYPED QUIT.  DO YOU NEED TO RETURN TO LARSYS TO SAVE ANY
STATISTICS OR HISTOGRAM FILES?
>no
CONNECT= 00:16:38 VIRTCPU= 000:17.75 TOTCPU= 000:33.58
LOGOFF AT 10:05:34 EST MONDAY 07/09/79
```

F. EXECUTE A LARSYS JOB, RECEIVING OUTPUT AT THE TYPEWRITER TERMINAL

PRINT If for some reason you would like to receive the printer output from a LARSYS job at your typewriter terminal, then there is a new LARSYS command to use called PRINT TYPEWRITER. Its format is:

PRINT TYPEWRITER linewidth

where "linewidth" is the number of characters which can be printed across your terminal. For the Decwriter terminal, use 120. This command must be entered shortly before the RUN command. For example, if a control card file called JAXPRI CC had been created, the following command sequence would execute the job and return the output for printing on the Decwriter.

```
>print typewriter 120
T=0.09/0.27 10:17:41
>ccinput jaxpri cc
T=0.11/0.20 10:19:22
>run
EXECUTION BEGINS...
 I0102  COMMENT - CLASSIFICATION RESULTS FROM RUN 76020106
 I0113  PRINTRESULTS FUNCTION REQUESTED   (PRISUP)

 I0071  PRINTRESULTS FUNCTION COMPLETED   (PRISUP)
 I0103  CPU TIME USED WAS  19.651 SECONDS.   (LARSMN)

 I0004  END OF INPUT DECK - RUN COMPLETED  (LARSMN)
 I0050  TOTAL CPU TIME FOR THIS RUN WAS  19.725 SECONDS.  (LARSMN)
TAPE 181 DETACHED
T=13.47/21.90 10:23:03
```

OUTPUT After the job completes execution, the print file is stored on a temporary disk file while you decide whether to print it out or run another LARSYS job. If you are ready to print the output, use the OUTPUT START command:

```
>output start
EXECUTION BEGINS...
```
(printer output is typed)

Warning: Do not try to print large outputs (such as all 4 channels in PICTUREPRINT) on the typewriter terminal, as it could take hours to print.

PRINT To shift the output from the typewriter to the printer, then use the command:

PRINT 'location'

to specify where the output should go.

```
>print stregis
T=0.09/0.27 10:35:01
>ccinput newprint cc
T=0.12/0.30 10:37:23
>run
EXECUTION BEGINS...
```

Note: The print location stays in effect for all jobs executed, until it is changed by another PRINT command.

G. SAVE STATISTICS FILES

One important data file which LARSYS creates and later uses as input to other processors is called the LARSYS Statistics File. When you execute a LARSYS processor which creates a Statistics File, it is important to save it on your computer ID's private disk storage after the job has finished executing. To do this, use the STATDECK command:

```
>statdeck save "name"
```

You should supply a name for the Statistics File which is meaningful to you. If you forget what names you have already assigned, use the command:

```
>statdeck status
```

to get a listing of all of your saved Statistics Files. Later, when you want to execute a LARSYS processor which requires the input of a Statistics File, use the command:

```
>statdeck use "name"
```

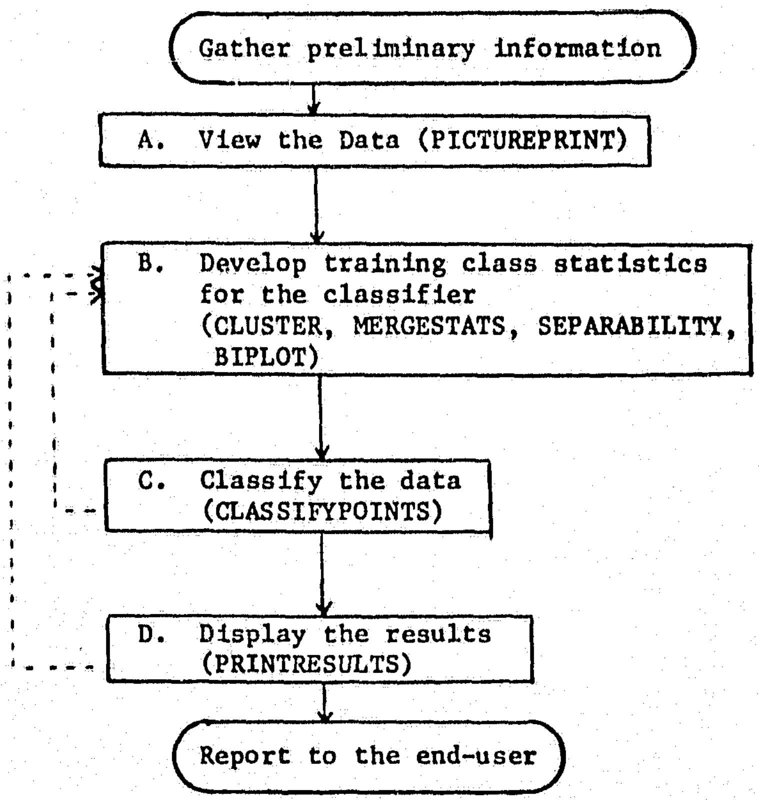
before typing the RUN command. Examples of these commands may be seen in the next section.

Section II

An Example LARSYS Analysis Sequence

Before beginning an analysis of remotely sensed data, you should be familiar with the names and general functions of the LARSYS processors, have a computer ID and password, know what data you will be analyzing, and collect the "ground truth" (any information you can get on the ground cover for parts of your analysis area, such as maps, photos, statistical reports, etc.) you will use. The digital data used by LARSYS is identified to the computer by a run number, which you should be informed of when the data is entered into the LARS Data Library. For the examples on the following pages the run number assigned to the data was 76020204. This data is over an area south-east of Columbus, Georgia.

The typical analysis sequence has been illustrated below.



A. OBTAIN A GRAYSCALE MAP OF THE DATA

PICTUREPRINT

The first step in the LARSYS Analysis sequence is to get a pictorial representation of the data to check data quality and geographically orient yourself in the data. The PICTUREPRINT function in LARSYS reads data from the LARSYS Multispectral Image Storage tape and produces alphanumeric pictorial printouts of the data for each channel that is specified. In this example maps of channels 2 and 4 (one visible and one infrared channel) will be printed. (Note: It is not a good idea to receive the output from PICTUREPRINT at the typewriter terminal unless you are looking at one channel of a small area.)

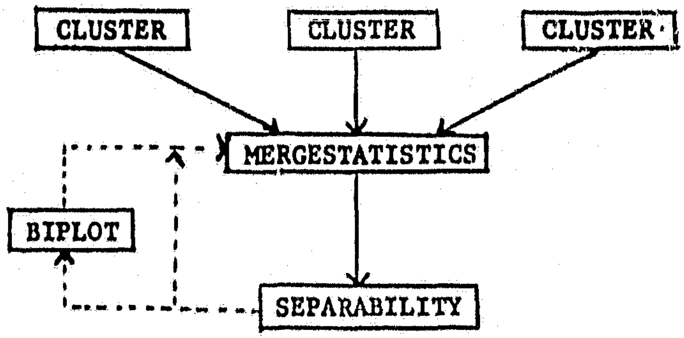
You should now turn to the control card reference page for PICTUREPRINT and write down the control cards you will need to use, checking the example which follows if necessary.

```
VM/370 ONLINE
!
>l stregis                                        (Logon procedure)
ENTER PASSWORD:
>XXXXXXXX
ENTER NAME: bud goodrick
LOGMSG - 08:01:17 EST TUESDAY 07/10/79
* YOUR OPERATOR THIS MORNING IS CINDY....
LOGON AT 08:44:35 EST TUESDAY 07/10/79
>i lsdv370
DEVELOPMENTAL LARSYS READY:
   SYSTEM IS BEING INITIALIZED....

... LARSYS IS READY FOR YOUR FIRST COMMAND
T=0.42/4.17 08:47:29
>get pic
THE FILE -- PIC CC -- IS READY TO BE EDITTED.
IT'S CONTENTS ARE:

-RUNTABLE                                         (Create the control
DATA                                               card file)
RUN(XX), TAPE(YY), FILE(ZZ)
END
-COMMENT GRAYSCALE MAP OF RUN XX
*PICTUREPRINT
DISPLAY  RUN(XX), LINES(A,B,C), COL(I,J,K)
CHANNELS 2,4
BLOCK    RUN(XX), LINES(A,B,C), COL(I,J,K)
END

EDIT:
>next
-RUNTABLE
>delete 4
>type
-COMMENT GRAYSCALE MAP OF RUN XX
>change /xx/76020204/ *  <------------------------ (Note: the * causes this
-COMMENT GRAYSCALE MAP OF RUN 76020204               change to be made in
                                                     every line)
```

```
DISPLAY  RUN(76020204), LINES(A,B,C), COL(I,J,K)
BLOCK    RUN(76020204), LINES(A,B,C), COL(I,J,K)
EOF:
>top
TOF:
>next
-COMMENT GRAYSCALE MAP OF RUN 76020204
>change /4/4 columbus, georgia test site/
-COMMENT GRAYSCALE MAP OF RUN 76020204 COLUMBUS, GEORGIA TEST SITE
>next 2
DISPLAY  RUN(76020204), LINES(A,B,C), COL(I,J,K)
>change /a,b,c/60,120,1  *
DISPLAY RUN(76020204), LINES(60,120,1), COL(I,J,K)
BLOCK  RUN(76020204), LINES(60,120,1), COL(I,J,K)
EOF:
>up
END
>up 3
DISPLAY RUN(76020204), LINES(60,120,1), COL(I,J,K)
>c /i,j,k/335,475,1/ *
DISPLAY RUN  (76020204), LINES(60,120,1), COL(335,475,1)
BLOCK   RUN(76020204), LINES(60,120,1), COL(335,475,1)
>top
TOF:
>type 99
TOF:
-COMMENT GRAYSCALE MAP OF RUN 76020204  COLUMBUS, GEORGIA TEST SITE
*PICTUREPRINT
DISPLAY  RUN(76020204), LINES(60,120,1), COL(335,475,1)
CHANNELS 2,4
BLOCK    RUN(76020204), LINES(60,120,1), COL(335,475,1)
END
EOF:
>file
T=0.66/8.51 09:08:22

>ccinput pic cc                                  (Execute the LARSYS
T=0.12/0.81 09:09:52                              PICTUREPRINT function)
>run
EXECUTION BEGINS....

 I0102  COMMENT - GRAYSCALE MAP OF RUN 76020204  COLUMBUS, GEORGIA TEST
SITE            (LARSMN)
 I0092  PICTUREPRINT FUNCTION REQUESTED       (PICSUP)
 I0237  ALL CONTROL CARDS FOR PICTUREPRINT HAVE BEEN READ     (PICRDR)
TAPE 182 ATTACHED
 I0002 TAPE 4523 HAS BEEN REQUESTED ON UNIT 182      (TAPMOUNT)
 I0003 TAPE READY...   EXECUTION CONTINUING     (TAPMOUNT)
 I0036  DESIRED RUN FOUND ... 76020204       (GADRUN)
 I0093  PICTUREPRINT FUNCTION COMPLETED       (PICSUP)
 I0103  CPU TIME USED WAS     36.012 SECONDS.     (LARSMN)

 I0004  END OF INPUT DECK - RUN COMPLETED      (LARSMN)
 I0050  TOTAL CPU TIME FOR THIS RUN WAS    36.197 SECONDS.      (LARSMN)
PRT FILE 6418  TO RSCS       COPY 01 NOHOLD
TAPE 182 DETACHED
T=11.74/41.47 09:21:16
```

B. DEVELOP TRAINING CLASS STATISTICS

This phase is the heart of the analysis process, and is a series of several steps, which may be re-iterated as needed. A general scheme of the order the processors may be used is illustrated below.



B1. CLUSTER - The cluster function implements an unsupervised classification (clustering) algorithm which groups data points into a user-specified number of clusters. The processor creates a LARSYS Statistics File of means and covariances which is needed as input to the classification processor and other LARSYS processors which use statistics information. The Statistics File must be saved on the user's private disk after the CLUSTER job has executed with the command 'STATDECK SAVE name', where 'name' is assigned by the user. Typically, the 'name' is chosen to be meaningful to the user, such as a location name. NOTE: The name must consist of only letters and numerals - no special characters are allowed.

Before setting up the control card file you need to select several training areas in the data containing points that represent all the cover types. Use your grayscale map from PICTUREPRINT to help you do this. Note the starting and ending lines and columns for each area. You will now execute a separate CLUSTER job on each training area.

Turn to the CLUSTER control card reference page and try to set up the first file you should use. The user-specified number of classes 'YY' is entered on the OPTIONS MAXCLAS(YY) card. In this example, 12 classes are requested for an area from line 50 to line 94 and column 420 to column 476 of the Columbus data.

```
>get clu                                          (Create the control
THE FILE -- CLU CC -- IS READY TO BE EDITTED.      card file for *CLUSTER)
IT'S CONTENTS ARE:

-RUNTABLE                  }                      (These lines are only
DATA                       }                       needed if the data is
RUN(XX), TAPE(YY), FILE(ZZ)} -------------->       copied to a user's tape.)
END                        }
-COMMENT CLUSTER ON RUN XX
*CLUSTER
OPTIONS  MAXCLAS(YY), CONV(97.5)
PUNCH  STATS
CHANNELS 1,2,3,4
```

```
DATA
RUN(XX), LINE(A,B,C), COL(I,J,K)
END

EDIT:
>next
-RUNTABLE
>delete 4
>type
-COMMENT CLUSTER ON RUN XX
CHANGE /XX/76020204/*
-COMMENT CLUSTER ON RUN 76020204
RUN(76020204), LINE(A,B,C), COL(I,J,K)
EOF:
>top
TOF:
>next
-COMMENT CLUSTER ON RUN 76020204
>next 2
OPTIONS MAXCLAS(YY), CONV(97.5)
>change .yy.12
OPTIONS MAXCLAS(12), CONV(97.5)
>change .7.9.
OPTIONS MAXCLAS(12), CONV(99.5)
>next
PUNCH    STATS
>delete
>next
DATA
>next
RUN(76020204), LINE(A,B,C), COL(I,J,K)
>change .a,b,c.50,94,1.
RUN(76020204, LINE(50,94,1), COL(I,J,K)
>change .i,j,k.420,276,1.
RUN(76020204), LINE(50,94,1), COL(420,476,1)
> top
TOF
> type 15
TOF:
-COMMENT CLUSTER ON RUN 76020204
*CLUSTER
OPTIONS  MAXCLAS(12), CONV(99.5)
CHANNELS 1,2,3,4
DATA
RUN(76020204), LINE(50,94,1), COL(420,476,1)
END
EOF:
> file
T=0.61/3.00 10:36:25
```

(These are the line and column coordinates for this training area)

```
>ccinput clu cc                                         (Execute the CLUSTER
T=0.08/0.61 10:08:16                                     job)
>run
EXECUTION BEGINS...
 I0102  COMMENT - CLUSTER ON RUN 76020204
                  (LARSMN)
 I0165 CLUSTER FUNCTION REQUESTED.      (CLUSUP)
 I0034  ALL CONTROL AND DATA CARDS HAVE BEEN READ.     (CLURDR)
 I0002 TAPE 4523 HAS BEEN REQUESTED ON UNIT 182     (TAPMOUNT)
TAPE 182 ATTACHED
 I0003 TAPE READY...  EXECUTION CONTINUING    (TAPMOUNT)
 I0036  DESIRED RUN FOUND ... 76020204      (GADRUN)
TAPE 182 DETACHED
TIME IS 10:09:34 EST TUESDAY 07/10/79
CONNECT= 01:25:00 VIRTCPU= 000:19.08 TOTCPU= 001:17.41
TIME IS 10:27:41 EST TUESDAY 07/10/79
CONNECT=01:43:06 VIRTCPU= 003:22.61 TOTCPU= 004:37.44
 I0171  FLAG=  0 NOMOD=  12 INTOT=  2565 INTV=   1 ITER=  16 TIME=  2
05 SECS
          NCHAN=    4 CHAN=  1  2  3  4
 I0166  CLUSTER FUNCTION COMPLETED.  (CLUSUP)
 I0103  CPU TIME USED WAS  220.473 SECONDS.     (LARSMN)
 I0004  END OF INPUT DECK - RUN COMPLETED      (LARSMN)
 I0050  TOTAL CPU TIME FOR THIS RUN WAS  221.120 SECONDS.     (LARSMN)
PRT FILE 6457  TO RSCS    COPY 01 NOHOLD
T=194.37/226.23 10:32:38

>statdeck save columbus                                  (Save the Statistics
T=0.36/2.40 10:39:28                                      File)

>statdeck status                                         (List all saved
 I0007                                                    Statistics Files)
FILENAME  FILETYPE  FM  FORMAT   RECS  BLKS   DATE    TIME
COLUMBUS  STATDECK  A1  F    80    57     6   7/10/79  10:29
COMBO     STATDECK  A1  F    80   119    12   6/19/79  17:49
DEC76     STATDECK  A1  F    80    70     7   5/23/79   9:29
DEC77     STATDECK  A1  F    80    76     8   4/20/79   9:33
MOD       STATDECK  A1  F    80    74     8   6/19/79  14:39
MODNULL   STATDECK  A1  F    80    65     7   6/19/79  13:58
NONUL     STATDECK  A1  F    80    65     7   7/06/79  14:07
PICAMOD   STATDECK  A1  F    80   103    11   5/10/79  11:03
T=0.21/2.00 10:41:32
```

Now repeat the above procedure for each training area you selected, by creating another CLUSTER control card file with the lines and columns for the next training area. Then issue the CCINPUT and RUN commands. After execution save the Statistics File with the STATDECK SAVE command, being sure to use a different name. For this example we created two other control card files using the above file called CLU CC and the EDIT command to change the line and column numbers.

ORIGINAL PAGE IS
OF POOR QUALITY

2nd CLUSTER job

```
>edit clu cc
EDIT:
>type 7
TOF:
-COMMENT CLUSTER ON RUN 76020204
*CLUSTER
OPTIONS  MAXCLAS(12), CONV(99.5)
CHANNELS 1,2,3,4
DATA
RUN(76020204), LINE(50,94,1), COL(420,476,1)
>change /50,94/79,113/
RUN(76020204), LINE(79,113,1), COL(420,476,1)
>change /420,476/333,364/
RUN(76020204), LINE(79,113,1), COL(333,364,1)
>file
T=0.23/4.72 07:32:47

>ccinput clu cc
T=0.12/1.01 07:33:16
>run
EXECUTION BEGINS...
```

(When execution is completed use the statdeck save command, using a different name than the first saved file)

3rd CLUSTER job

```
>edit clu cc
EDIT:
>type 7
TOF:
-COMMENT CLUSTER ON RUN 76020204
*CLUSTER
OPTIONS  MAXCLAS(12), CONV(99.5)
CHANNELS 1,2,3,4
DATA
RUN(76020204), LINE(79,113,1), COL(333,364,1)
>change /79,113/60,90/
RUN(76020204), LINE(60,90,1), COL(333,364,1)
>change /333,364/420,475/
RUN(76020204), LINE(60,90,1), COL(420,475,1)
>file
T=0.26/1.03 15:48:26
>ccinput clu cc
T=0.08/0.23 15:57:13
>run
EXECUTION BEGINS ...
```

(When execution is completed, use the statdeck save command, using a name different from the above two saved files)

(If you are done for now, type QUIT.)

```
>quit
YOU TYPED QUIT.  DO YOU NEED TO RETURN TO LARSYS TO SAVE ANY
STATISTICS OR HISTOGRAM FILES?
>no
CONNECT= 02:00:04 VIRTCPU= 003:30.29 TOTCPU= 004:59.94
LOGOFF AT 10:44:41 EST TUESDAY 07/10/79

VM/370 ONLINE
```

Obtain the print files for these CLUSTER jobs and compare the cluster maps in the output to any ground truth (maps or photos) you have. Give a name to as many of the cluster classes as possible in each of the three areas. Don't worry if you can't name all the classes, since we have several other programs to aid us.

Remember that you created three separate Statistics Files. It is very probable that there are statistical values in each Statistics File that represent the same cover type on the ground. To combine these classes and allow us to better compare the other classes, it is necessary to merge the Statistics Files.

B2. BMERGESTATISTICS - The mergestatistics processor provides the capability of combining several LARSYS Statistics Files or decks and editting the resulting new file. Classes from the input Statistics decks may be combined (pooled), deleted and renamed. This processor is frequently used in conjunction with the SEPARABILITY processor (discussed next) to identify redundant or unnecessary classes, and then combine or delete them as required. The 'Separability-Mergestatistics' processor combination may be repeated if the analyst feels class redundancy justifies its use.

The version of this program we use is called BMERGESTATISTICS, so we create our control card file and execute the job as follows:

```
>logon stregis                                   (Logon)
ENTER PASSWORD:
>XXXXXXXX
ENTER NAME:>bud goodrick
LOGMSG - 13:00:06 EST THURSDAY 07/19/79
* YOUR OPERATOR THIS AFTERNOON IS CINDY....
* NEXT SCHEDULED SHUTDOWN IS SATURDAY JULY 21 AT 17:00...
LOGON AT 13:00:58 EST THURSDAY 07/19/79
>i lsdv370
DEVELOPMENTAL LARSYS READY;
   SYSTEM IS BEING INITIALIZED....

... LARSYS IS READY FOR YOUR FIRST COMMAND
T=0.35/4.01 11:47:41                        (If you have forgotten the names
>get bme cc                                  of your statistics files use the
THE FILE -- BME CC -- IS READY TO BE EDITTED. STATDECK STATUS command before
IT'S CONTENTS ARE:                           the GET command.)

*BMERGE                                     (Create the control card file)
OPTIONS NOFIELD, COSPEC
CLASSES ENTIRE(1,2)
SCALE SPCINT(1)
DISK READ
DATA  (USE GETFILE COMMAND TO INSERT YOUR STATDECK AFTER THIS CARD)
DATA  (USE GETFILE COMMAND TO INSERT YOUR STATDECK AFTER THIS CARD)
END

EDIT:
>next 3
CLASSES ENTIRE(1,2)
>change /2/2,3/
CLASSES ENTIRE(1,2,3)
>next
SCALE SPCINT(1)
>next
DISK READ
>delete
>type
DATA  (USE GETFILE COMMAND TO INSERT YOUR STATDECK AFTER THIS CARD)
>getfile csg statdeck
EOF REACHED
EOS              *****     LAST CARD OF STATISTICS DECK     *****
```

```
>next
DATA     (USE GETFILE COMMAND TO INSERT YOUR STATDECK AFTER THIS CARD)
>getfile au49 statdeck
EOF REACHED
EOS                  *****     LAST CARD OF STATISTICS DECK     *****
>input data
>getfile columbus statdeck
EOF REACHED
EOS                  *****     LAST CARD OF STATISTICS DECK     *****
>next
END
>top
TOF:
>type 8
TOF:
*BMERGE
OPTIONS NOFIELD, COSPEC
CLASSES ENTIRE(1,2,3)
SCALE SPCINT(1)
DATA     (USE GETFILE COMMAND TO INSERT YOUR STATDECK AFTER THIS CARD)
LARSYS VERSION 3 STATISTICS FILE          0
CLASS TPL7601
>file
T=0.54/7.48 11:59:17
>ccinput bme cc
T=0.54/7.48 11:59:41
>run                                            (Execute the BMERGE job)
EXECUTION BEGINS...
 IOXXX  MERGESTATISTICS FUNCTION REQUESTED      (MERSUP)

 I0034  ALL CONTROL AND DATA CARDS HAVE BEEN READ      (MERSUP)
 IOXXX  STATISTICS BEING MERGED      (MERSTT)
 I0032  REDUCED STATISTICS COMPUTED.     (REDSAV)
 I0032  REDUCED STATISTICS COMPUTED      (REDSAV)
 I0032  REDUCED STATISTICS COMPUTED.     (REDSAV)
 I0209  COINCIDENT BI-SPECTRAL PLOT PRINTED.     (COSPEC)
 IOXXX  MERGESTATISTICS FUNCTION COMPLETED      (MERSUP)
 I0103  CPU TIME USED WAS     16.155 SECONDS.     (LARSMN)

 I0004  END OF INPUT DECK - RUN COMPLETED     (LARSMN)
 I0050  TOTAL CPU TIME FOR THIS RUN WAS   16.400 SECONDS.     (LARSMN)
PRT FILE 4952  TO RSCS    COPY 01 NOHOLD
T=9.96/22.11 12:03:14
 >statdeck save csg3deck
T=0.61/1.96 12:12:01
```

B3. SEPARABILITY - The separability processor helps the user to decide how well the individual classes in a LARSYS Statistics File may be distinguished from one another. i.e. What is the "separability" between the classes. It is also used to select a subset of channels in a Statistics File which will produce an accurate classification. The first reason is why we will use the separability function now. We want to compare all of our cluster classes which have been combined into one Statistics File during the BMERGESTATISTICS job.

Now create your SEPARABILITY control card file. If you have forgotten the names of your Statistics Files, type 'statdeck status' before issuing the 'get' command.

```
>get sep
THE FILE -- SEP CC -- IS READY TO BE EDITTED.
IT'S CONTENTS ARE:

-COMMENT SEPARABILITY ON CLASSES FROM RUN XX
*SEPARABILITY
COMBINATIONS 4
SYMBOLS  A,B
CARD     READSTATS
CHANNELS 1,2,3,4
DATA
* PUT STATISTICS DECK HERE IF A CARDS READSTATS CARD IS ABOVE *
   DON'T FORGET TO DELETE THESE TWO LINES.
END

EDIT:
>next
-COMMENT SEPARABILITY ON CLASSES FROM RUN XX
>change /XX/76020204/
-COMMENT SEPARABILITY ON CLASSES FROM RUN 76020204
>next 3
SYMBOLS A,B
>delete 6
>up
COMBINATIONS 4
>input print div(1200)
>top
TOF:
>type 6
TOF:
-COMMENT
*SEPARABILITY
COMBINATIONS 4
DIV (1200)
END
>file
T=0.54/1.28 13:45:21
```

(The 4 indicates we want to use all 4 channels in computing class separabilities)

(Class pair separabilities are assigned a number ≤ 2000, the higher the number, the more separable the classes. This line causes a table to be printed of classes with separability less than 1200 which are candidates for combining.)

```
>statdeck use csg3deck
T=0.59/1.56 12:04:31
>ccinput sep cc
T=0.08/0.20 12:05:50
>run
EXECUTION BEGINS...
 I0111  SEPARABILITY FUNCTION REQUESTED    (SEPSUP)
 I0032  REDUCED STATISTICS COMPUTED.   (REDSAV)
 I0034  ALL CONTROL AND DATA CARDS HAVE BEEN READ     (SEPINT)
 I0022  DIVERGENCE CALCULATIONS COMPLETED--READY TO ORDER AND PRINT RESULTS
 I0011  SEPARABILITY FUNCTION COMPLETED    (SEPSUP)
 I0103  CPU TIME USED WAS    22.083 SECONDS.     (LARSMN)

 I0004  END OF INPUT DECK - RUN COMPLETED     (LARSMN)
 I0050  TOTAL CPU TIME FOR THIS RUN WAS    22.339 SECONDS.    (LARSMN)
PRT FILE 4954  TO RSCS    COPY 01 NOHOLD
T=13.49/27.30 12:10:34
```

After obtaining the printer output from this SEPARABILITY job, turn to the table of suggested class groupings. Compare this with the names you assigned to the classes in the CLUSTER output. Based on how well the grouped classes agree with respect to cover type, you should decide whether to combine or delete classes.

Your next step is to create another BMERGESTATISTICS control card file with the class combinations indicated on the POOL card, and then re-run the SEPARABILITY processor, changing the DIV value to 1400. Do this as many times as is necessary. Don't forget to save the Statistics File after BMERGESTATISTICS is executed.

After the 2nd BMERGESTATISTICS job has been executed, you may also choose to run the BIPLOT processor, as follows:

B4. BIPLOT - The Biplot processor provides the user with a graphic presentation of the relationships of the classes in a statistics deck. Two-channel plots of means, ellipsoids of concentration, and classification space for training classes may be requested. The plots are used to examine the spatial relationship of training classes in two channel space. Any two channel combinations may be plotted.

```
>get bip
THE FILE -- BIP CC -- IS READY TO BE EDITTED.
IT'S CONTENTS ARE:

-COMMENT BIPLOT OF CHANNELS X,Y
*BIPLOT
PLOT MEANS(X,Y),ELLIPSE(X,Y),CLASS(X,Y)
SCALE ORIG(X,0.0), ORIG(Y,0.0), UNIT(X,0.8), UNIT(Y,0.6)
END

EDIT:                      (issue Edit commands here to create a file
                            which looks like this:)
>type 5
TOF:
*BIPLOT
PLOT MEANS(3,2),CLASS(3,2)
SCALE ORIG(3,0.0),ORIG(2,0.0),UNIT(3,0.5),UNIT(2,0.5)
END
>file
T=0.09/1.17 17:05:12
>statdeck use csg          (our final Statistics File is named 'csg')
T=0.42/0.61 17:07:49
>run
EXECUTION BEGINS...
 I0000  BIPLOT FUNCTION SELECTED                 (BIPSUP)
 I0000  READ STATISTICS COMPLETED                (BIPRDR)
 I0000  ALL CONTROL AND DATA CARDS HAVE BEEN READ (BIPSUP)
 I0000  DOING CLASSIFY PLOT OF  2 VS 3           (BIPLTR)
 I0000  DOING    MEANS PLOT OF  2 VS 3           (BIPLTR)
 I0000  BIPLOT FUNCTION COMPLETED                (BIPSUP)
 I0103  CPU TIME USED WAS  12.108 SECONDS.       (LARSMN)

 I0004  END OF INPUT DECK - RUN COMPLETED     (LARSMN)
 I0050  TOTAL CPU TIME FOR THIS RUN WAS  12.228 SECONDS.    (LARSMN)
PRT FILE 5943  TO RSCS    COPY 01 NOHOLD
T=11.22/13.85 17:08:56
```

C. CLASSIFY THE DATA

Having completed the necessary iteration of steps in part B. to develop your training class statistics, you are ready to classify all the data points in your analysis area.

CLASSIFYPOINTS - The classifypoints processor uses the maximum likelihood classification rule which classifies multispectral data on a point-by-point basis. The processor assigns each data point to a class in the training set (LARSYS Statistics Files) for which the data values give the maximum likelihood for statistically correct classification. The classification file is written onto tape or disk.

Now create a control card file for CLASSIFYPOINTS and execute the job.

```
>get cla
THE FILE -- CLA CC -- IS READY TO BE EDITTED.
IT'S CONTENTS ARE:

-RUNTABLE
DATA
RUN(XX), TAPE(YY), FILE(ZZ)
END
-COMMENT PERPOINT CLASSIFICATION OF RUN XX
*CLASSIFYPOINTS
RESULTS  TAPE(X),FILE(Y)
CLASSES  MM(P1/C1,C2/)  <----------  (The 'classes' card is not needed if
CHANNELS 1,2,3,4                      you have already combined your clas-
DATA                                  es in the BMERGE jobs)
RUN(XX), LINE(A,B,C), COL(I,J,K)
END

EDIT:                   <----------  (Issue Edit commands here to get a
                                      file such as the one typed below)
*CLASSIFYPOINTS
RESULTS DISK  <---------------------  (If you have a larger area, you will
CHANNELS 1,2,3,4                      need to put it on tape instead of disk)
DATA
RUN(76020204),LINE(140,180,1),COL(60,200,1)
END
>file
T=0.15/0.49 14:51:35
>ccinput cla cc
T=0.09/0.18 14:52:03
>statdeck use csg
T=0.09/0.23 14:56:39
>run
EXECUTION BEGINS...
```

At this point the data has been classified and you can proceed to display the data.

D. DISPLAY THE RESULTS

PRINTRESULTS - The Printresults processor produces printed outputs describing the classification results, in the form of map images, tabular summaries, or both. The user can assign various symbols to the classes for the maps, and produce several types of tables. All output products are optional and various combinations of products may be produced, with multiple copies produced if requested.

If you choose to produce a map, then you must first select symbols to assign to each class. Many special symbols may be used, such as 'blank', ., /, +, =, -, etc. in addition to letters and numbers.

```
>get pri
THE FILE -- PRI CC -- IS READY TO BE EDITTED.
IT'S CONTENTS ARE:

-COMMENT CLASSIFICATION RESULTS FROM RUN XX
*PRINTRESULTS
RESULTS  TAPE(TT), FILE(FF) <-- (specify here where the classification was
PRINT    TEST(P)                 written)
SYMBOLS -,+,.  <---------------- (symbols are required for a map)
GROUP    GG(G1/P1,P2/)
DATA                            (data cards are needed if tables are requested)
TEST
RUN(XX), LINES(A,B,C), COL(I,J,K)
END

EDIT:                           (Issue Edit commands here to create your file.
                                 The following file just requests map output.)
*PRINTRESULTS
RESULTS DISK
SYMBOLS  .,.,.,.,.,.,/,/,/,/,X,X,X,H,H,H,W
END
>file
```

Execute the job using the CCINPUT and RUN commands. Obtain your printer output and look for problem areas in the classification. It may be necessary to repeat several steps from part B to redefine the training class statistics. If no problems are apparent, then you have completed the analysis sequence and may produce various other maps or tables, grouping similar class types as desired.

APPENDIX H

Instructions to Convert Linear Parameters to 4-Parameter Non-Conformal Transformation for use by the LARS Image Registration System.

(This algorithm is for use with Landsat-EDIPS formatted geometrically corrected tapes).

1. Set up the distortion matrix $\underline{M}$ such that

$$\underline{M} = \underline{M}_1 \underline{M}_2 \underline{M}_3 \underline{M}_4 \underline{M}_5$$

where

$$\underline{M}_1 = \begin{bmatrix} 1.0 & 0.0 \\ 0.0 & 1.0 \end{bmatrix}$$

for pixel scale (assumes 57m pixel)

$$\underline{M}_2 = \begin{bmatrix} \cos\Theta & \sin\Theta \\ -\sin\Theta & \cos\Theta \end{bmatrix}$$

for rotation, where Θ is the rotation required to a north-south orientation.

$$\Theta = \sin^{-1}\left(\frac{\cos\ (80.985)}{\cos\ \phi}\right)$$

where ϕ is run center latitude

$$\underline{M}_3 = \begin{bmatrix} 1.0 & 0.0 \\ 1.0 & 0.0 \end{bmatrix}$$

for skew (already corrected by EDIPS)

$$\underline{M}_4 = \begin{bmatrix} 0.8 & 0.0 \\ 0.0 & 1.0 \end{bmatrix}$$

for a line printer aspect of 8 lines per inch, 10 columns per inch

$$\underline{M}_5 = \begin{bmatrix} 1.3368421 & 0.0 \\ 0.0 & 1.3368421 \end{bmatrix}$$

for scale correction to 1:24000 from a scale of 1:17952.7

$$\underline{M} = \begin{bmatrix} 1.0694737 \cos \Theta & 1.3368421 \sin \Theta \\ -1.0694737 \sin \Theta & 1.3368421 \cos \Theta \end{bmatrix}$$

This distortion matrix is set up for the transformation

$$\underline{Y} = \underline{M}\ \underline{X}$$

where $\underline{X}$ is the original coordinates and
$\underline{Y}$ is the new coordinate matrix.

2. To put this into a form suitable for the registration equation:

$$\underline{Y} = \underline{X} + \underline{\Delta}$$

we must use the form

$$Y = \underline{A}\ \underline{X}$$

Expanding: $\underline{Y} = \underline{A}\ \underline{X} = \underline{X} + \underline{\Delta}$

or

$$Y_1 = a_{11}x_1 + a_{12}x_2 = ax_1 + bx_2 + c + x_1$$

$$Y_2 = a_{21}x_1 + a_{22}x_2 = dx_1 + ex_2 + f + x_2$$

let $c = f = 0$,

$$a_{11}x_1 + a_{12}x_2 = (a + 1)x_1 + bx_2$$

$$a_{21}x_1 + a_{22}x_2 = dx_1 + (e + 1)x_2$$

or,

$$a_{11} = a + 1$$

$$a_{12} = b$$

$$a_{21} = d$$

$$a_{22} = e + 1$$

or, $\underline{A} = \begin{bmatrix} a + 1 & b \\ d & e + 1 \end{bmatrix}$

3. To convert $\underline{A}$ in terms of overlay line (CLD) and column (CJD) coefficients of the image registration system, it is only necessary to observe

$$
\begin{aligned}
a &= CJD3 = 1.0694737 \cos \Theta - 1 \\
b &= CJD2 = 1.3368421 \sin \\
d &= CLD3 = -1.0694737 \sin \\
e &= CLD2 = 1.3368421 \cos \Theta - 1
\end{aligned}
$$

IMAGE REGISTRATION PERFORMANCE REPORT

ORIGINAL PAGE IS
OF POOR QUALITY

WP= 556-PR

DATE: 9/25/79 REFORMATTER: Smith

MACT= 796

CPU=

INPUT

RUN A 77010200 LINES 1,600 COLS 75,435 CHAN -
RUN B 77010201 LINES 1,750 COLS500,1100CHAN All
BUFFER USED:

OUTPUT

RUN 77010202 TAPE 2668 FILE 2
INTERPOLATION: NN

DATA TRANSFORMATION INFORMATION

TRANSFORMATION: FORWARD BACKWARD ORDER 1

LINE RMS : 0.70806

COL RMS : 1.03212

N : 14

DISTRIBUTION R: 0.99

Registered to USGS Quad Maps

CORRELATION NA

RATE OF ACCEPTANCE ACCEPT______ REJECT______ RATE______

AVERAGE RHO :

AVERAGE EUCLIDEAN ERROR :

TRANSFORMATION COEFFICIENTS:

	LINE COEF.		COL COEF.
1:	32.494452	1:	543.055448
2:	0.015229	2:	-0.005780
3:	0.005092	3:	-0.013307
4:		4:	
5:		5:	
6:		6:	

THE FOLLOWING FIRST ORDER DISTORTIONS WERE CORRECTED:

SCALE X: 1.0152 Y: 1.0000

ROTATION 0.326 (DEGREES)

SKEW: 0.0299 (DEGREES)

ORIGINAL PAGE IS OF POOR QUALITY

IMAGE REGISTRATION PERFORMANCE REPORT

WP= 583-IR

DATE: 10/11/79 REFORMATTER: Smith

MACT= 796

CPU=

INPUT

RUN A 77010202 LINES 1,600 COLS 1,428 CHAN 1-4
RUN B 79000201 LINES 1,739 COLS 1,504 CHAN 1-4
BUFFER USED: 7246 bytes

OUTPUT

RUN 79000202 TAPE 2848 FILE 1
INTERPOLATION: NN

DATA TRANSFORMATION INFORMATION

TRANSFORMATION: FORWARD BACKWARD ORDER 2

LINE RMS : 0.099

COL RMS : 0.283

N : 185

DISTRIBUTION R: NA distribution appears excellent

CORRELATION

RATE OF ACCEPTANCE ACCEPT 185 REJECT 85 RATE

AVERAGE RHO : 0.69

AVERAGE EUCLIDEAN ERROR : 0.67

TRANSFORMATION COEFFICIENTS:

LINE COEF. 1:	129.20298180	COL COEF. 1:	66.39532730
2:	-0.00008529	2:	0.00264544
3:	-0.00133339	3:	0.00066831
4:	-0.00000039	4:	-0.00000293
5:	0.00000173	5:	-0.00000105
6:	0.00000115	6:	0.00000014

THE FOLLOWING FIRST ORDER DISTORTIONS WERE CORRECTED:

SCALE X: 0.9999183 Y: 1.0006683

ROTATION -0.1515851 (DEGREES)

SKEW: 0.075 (DEGREES)

ORIGINAL PAGE IS OF POOR QUALITY

IMAGE REGISTRATION PERFORMANCE REPORT

WP= 572-GC

DATE: 10/5/79 REFORMATTER: Smith

MACT= 796

CPU=

INPUT

RUN A 79000300 LINES 762,1500 COLS 1200,1698 CHAN none
RUN B 79000200 LINES 600,1700 COLS 1000,2150 CHAN 1-4
BUFFER USED: 1,500,000

OUTPUT

RUN 79000201 TAPE 2836 FILE 1
INTERPOLATION:

DATA TRANSFORMATION INFORMATION

TRANSFORMATION: FORWARD BACKWARD ORDER 1

LINE RMS :

COL RMS : NA: systematic distortions corrected for rotation, scale, aspect only

N :

DISTRIBUTION R:

CORRELATION NA

RATE OF ACCEPTANCE ACCEPT______ REJECT______ RATE______

AVERAGE RHO :

AVERAGE EUCLIDEAN ERROR : NA

TRANSFORMATION COEFFICIENTS:

LINE COEF. 1:	0.0	COL COEF. 1:	0.0
2:	0.314711	2:	0.2422435
3:	-0.1937948	3:	0.0517688
4:		4:	
5:		5:	
6:		6:	

THE FOLLOWING FIRST ORDER DISTORTIONS WERE CORRECTED:

SCALE X: 1.3368421 1:24000 LP Y: 1.0694737 1:24000 LP

ROTATION $10^{\circ}.44$ (DEGREES)

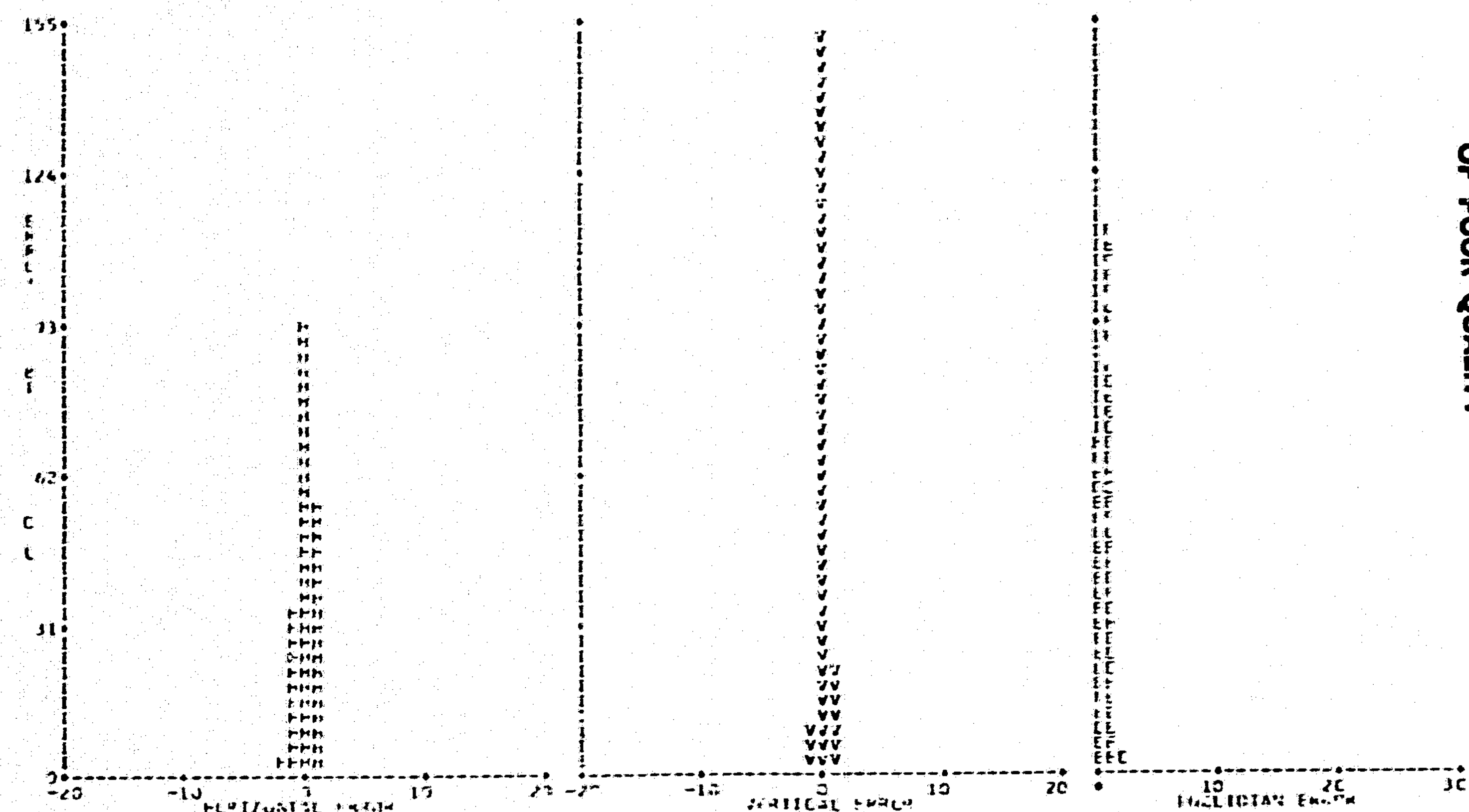
SKEW: $0^{\circ}.0$ (DEGREES)

ORIGINAL PAGE IS
OF POOR QUALITY

RMS PEAK CORRELATION VALUE....... = 0.7[illegible]
NUMBER OF ACCEPTABLE CORRELATIONS = 14[illegible]
CORRELATION WINDOW = 32 BY 32 SQUARE

AVERAGE PEAK CORRELATION VALUE = 0.67
NUMBER OF REJECTED CORRELATIONS = 85
CORRELATION DELTA............... = 4

****** HISTOGRAMS OF CORRELATION DISTANCE ERRORS ******



3.4 Ratio Evaluation

During the course of the FRIS Project LARS Project personnel became aware of forest managements need to quantitatively relate Landsat and forest inventory data. One approach that was especially noteworthy involved the application of regression analysis to Landsat MSS reflectance values. The predicted variable was the age of pine plantations, which is an indirect measure of crown closure. Crown closure is a measure of stand stocking which is an inventory measure.

More precisely the ratio of the infrared to visible band responses are assumed to be affected by stand occupancy, which is reflected in crown closure. As stands mature, individual tree crowns occupy a greater proportion of the site (figure 3.4.1). The increasing crown closure affects the ratio, which in preliminary tests corresponds well to a measure of age.

3.4.1 Knabb and Picayune Ratio Results

The ratio of IR channels to visible channels from December 1977 Landsat data for the Knabb and Picayune tracts were used to predict the age of selected pine fields. The exact ratio used, the method of picking pine fields, the analysis used to predict the fields' ages and the results of these predictions are outlined below.

The exact data ratio generated was as follows:

$$\text{ratio} = 40.0(C3 + C4)/(C1 + C2 + 0.1)$$

where

$C1$ = channel 1

$C2$ = channel 2

$C3$ = channel 3

$C4$ = channel 4

The multiplier 40.0 and the constant 0.1 were needed to enhance the range of and information in the data, and to prevent a divisor of zero.

The Knabb Tract was first categorized into pine and non-pine classes. From this classification fourteen fields of seemingly homogeneous pine were selected and the average ratio for each field was determined. Due to the proximity of this tract to the Fargo test site and their similar physiography, a regression equation developed for Fargo was used to predict the ages of the selected Knabb fields. Four of the Knabb fields were dropped from further analysis. Two of these discarded fields were accidently picked outside the Knabb boundaries and the other two dropped fields were inaccessible for checking ground truth. Of the ten pine fields left, a ground inspection of the area established that (1) all ten fields were pine, and (2) nine of the ten fields had ages within the ninety percent confidence interval for each predicted age. Ages were derived by taking increment cores and counting growth rings of randomly selected dominant trees.

Exhibit 1

NASA SATELLITE TO AID TIMBER INDUSTRY

HON. DON FUQUA

OF FLORIDA

IN THE HOUSE OF REPRESENTATIVES

Monday, April 28, 1980

● Mr. FUQUA. Mr. Speaker, I would like to call to the attention of my colleagues a recent release by NASA depicting a case in cost sharing between a Government agency and private concern. This is another example of the benefits being derived by the technology being developed in our space program.

An Earth resources NASA satellite has found a new use: Gathering data to improve management of America's forest lands. The project reflects a unique relationship between the Government agency and the private sector, one in which the initiating company is sharing the total cost but the technology developed will be available to all other timber companies. The satellite is Landsat-3. The company is the St. Regis Paper Co.

Since 1977, St. Regis has been working with NASA in a test program to see if Landsat data could improve the paper industry's information base on forest lands. St. Regis wants to use the data for planning timber harvests, leasing and buying new timber lands, and to monitor more than 920,000 hectares—2.3 million acres—across the South.

The project has been so successful that the St. Regis Southern Timberland Division, Jacksonville, Fla., recently authorized over $300,000 of new capital investment for a forest resource information system to use Landsat data to supplement conventionally acquired data in its general operations.

The entire forestry industry stands to gain from the venture because technology developed by the St. Regis experiment is in the public domain and available to other companies. The company and NASA plan to conduct a symposium in 1981 to demonstrate Landsat interpretation methods to timber industry managers.

St. Regis was the first private company to act as a user in NASA's resource observation applications test program. The project established a unique relationship between NASA and the private sector for St. Regis initiated the project—rather than NASA—and the company is sharing in the total cost.

St. Regis will use techniques developed in the project at its Dallas computer facility to process the data coming from Landsat which will complement the automated data base at St. Regis divisional remote sensing center in Jacksonville. This combined data will assist the company in estimating timber volume and productivity, as well as changes in the conditions of the forests.

Landsat data will be used to determine distribution of the major types of trees on St. Regis-owned timberland in Texas, Florida, Georgia, Alabama, Mississippi, and Louisiana and other land available through lease or purchase.

The project is being conducted by NASA's Johnson Space Center, Houston; the Laboratory for Applications of Remote Sensing at Purdue University in Lafayette, Ind.; and the St. Regis Paper Co. It is scheduled to end in September of this year.

Landsat-3 orbits the globe 14 times a day at an altitude of 900 kilometers—560 miles. Its electronic multispectral scanner returns data which is processed into film and computer tape format. This resource information may be used to characterize different types of terrain, vegetation, soils, rocks, and other surface features.●