~~028~~
022882

Remote Sensing Image Processing on the COMTAL Vision One/20

prepared for
COMTAL Corporation

Philip H. Swain
and
Shirley M. Davis

LARS Contract Report 022882

Purdue University
Laboratory for Applications of Remote Sensing
West Lafayette, Indiana 47907 USA

PREFACE

Under sponsorship of the COMTAL/3M Corporation, the Laboratory for Applications of Remote Sensing (LARS) at Purdue University has initiated development of a COMTAL Vision One-based system for remote sensing image processing. This system may be viewed as a primary component in a georeferenced information system capable of integrating a spectrum of remote sensing, geophysical, geopolitical, meteorological and other forms of data in a tool for resource monitoring, mineral exploration, urban planning and numerous other applications.

This three-month project had two specific objectives, both of which have been accomplished as described below.

OBJECTIVE #1: To implement an interactive clustering algorithm for multispectral remote sensing data on the COMTAL Vision One in order to test the feasibility of using the COMTAL Vision One for remote sensing image processing; and, along with this, to establish a baseline approach for implementing and supporting with documentation and user aids such a COMTAL-based remote sensing system.

An ISODATA-type iterative clustering algorithm has been programmed to run on the Vision One system computer. The prototype code has been tested on a variety of Landsat data and, in terms of processing speed and ability to cluster large areas of multispectral data, may be deemed a successful demonstration of the feasibility of using the Vision One in this context. By means of this algorithm, the remote sensing data analyst can interact with the algorithm and the remote sensing data to achieve the desired unsupervised classification results both effectively and expediently. It is expected that this capability will be dramatically enhanced when the Vision One system computer is upgraded from the present LSI-11/02 to an LSI-11/23.

The documentation developed to support the clustering algorithm includes:

- An introduction to the georeferenced information system concept, sketching the overall system in general terms and, more specifically, the role of clustering in the overall data processing scheme.

- The Cluster processor component of a user's reference manual, documenting the use of the processor for remote sensing image processing.

- A self-taught hands-on tutorial module designed to teach effective use of the Cluster processor, containing step-by-step instructions and a typical array of supporting reference data.

- System documentation which, together with the internally documented program source code, describes the algorithm implementation from a programmer's viewpoint.

- Test procedures for establishing the validity of the code as installed locally.

- A distribution tape containing the source code, assembled code, test patterns, and Landsat imagery supporting the tutorial module.

All of these items, except the distribution tape which is supplied separately, are included in this report.

OBJECTIVE #2: <u>To formulate for COMTAL Corporation a system concept and implementation proposal for a Vision One-based system to support remote sensing applications and research</u>.

Drawing on its longstanding and multidisciplinary experience within the remote sensing community, the staff of Purdue/LARS prepared and delivered to COMTAL in December, 1981, a proposal to "Develop an Interactive Georeferenced Information System Based on the COMTAL Vision One/20." In recent years, the potential role of remote sensing, including multispectral image processing, in monitoring and managing the earth's renewable and nonrenewable resources has become clear. Remote sensing provides only a subset, albeit a key subset, of the forms of information essential for such applications as land-use planning, geologic mapping and mineral exploration, agricultural inventory, forest/wildland/wetland monitoring, and so on. The LARS proposal, therefore, encompasses a generalized georeferenced information system capable of managing and analyzing the diverse forms of data required by such applications. The Vision One system is cast as the primary interactive device in the system, optimally utilizing its unique architecture and capabilities for providing an effective interface between the user/analyst and the data and processing facilities of the georeferenced information system. It is believed that such a system is in great demand world-wide by industrial, governmental, educational and research organizations anxious to make more effective use of the staggering array of data now available to them.

Since the current economic climate apparently does not support pursuit of such an ambitious development project at this time, LARS submitted an informal alternative proposal of more limited scope in January 1982. In view of the success of the project documented herein and the great potential for meeting real users' needs with COMTAL-based systems such as those proposed, we hope that COMTAL Corporation will be able to find resources available to continue these efforts, even if necessarily at the more limited level.

The authors are indebted to Mr. Jeffrey Welch, Systems Engineer at LARS, who programmed the Cluster processor for the Vision One/20; and to Mr. Ronald J. Clouthier, Marketing Support Manager at COMTAL/3M, for his support and continued encouragement.

TABLE OF CONTENTS

INTRODUCTION TO THE GEOREFERENCED INFORMATION SYSTEM

Georeferenced information systems are multi-faceted, computer-based information systems which input, store, and manipulate geographically encoded data to produce information pertinent to specific geographic locations or regions. The input data may range from satellite-obtained weather data, such as that derived from the GOES System, to census data, to in situ measurements of physical phenomena. Because of its repetitive global coverage, Landsat is often used as the source of base maps in geographic information systems. Because of the complexity of the data involved and the operations which need to be applied to these data, interactive facilities, including image display systems such as the COMTAL Vision One, are essential components of the most useful georeferenced information systems.

Georeferenced information systems can be represented conceptually as five inter-related component subsystems, as shown in Figure 1. The input subsystem includes data entry and file maintenance operations. Since the data typically are of many types and from diverse sources, techniques for reformatting digital tapes and digitizing image and map information are integral parts of the input subsystem. Assessment of the quality of the input is also an operation in this subsystem, carried out to ensure the dependability of the data accessed from the data base. Perhaps the most important operations in the input subsystem are image digitization, registration and rectification, operations that allow diverse data types to be accessed and used in a coordinated manner. The COMTAL Vision One may aid in implementing these operations by providing a channel for digitized video input and an interactive facility for locating and defining the required geographic registration control points.

Low-level information extraction, another of the five component subsystems in Figure 1, includes a wide range of data processing and analysis functions. Data enhancement may be carried out at this stage, utilizing such operations as edge enhancement, ratio transformation, illumination normalization, principal components transformations, to name a few. These operations also include classification operations ranging from single-channel level slicing to implementation of the more complex multi-channel classification rules which may incorporate several types of data from which to derive the classification. In some cases the results from this step may be the final product desired from the georeferenced information system; in others they may be used as input to high-level information extraction, another subsystem shown in Figure 1. The interactive Vision One system enhances the execution of the low-level information extraction operations by providing facilities for direct image interpretation, parameter determination and results evaluation.

Processors included in the high-level information extraction subsystem perform area mensuration, statistical analyses, and spatial analysis based

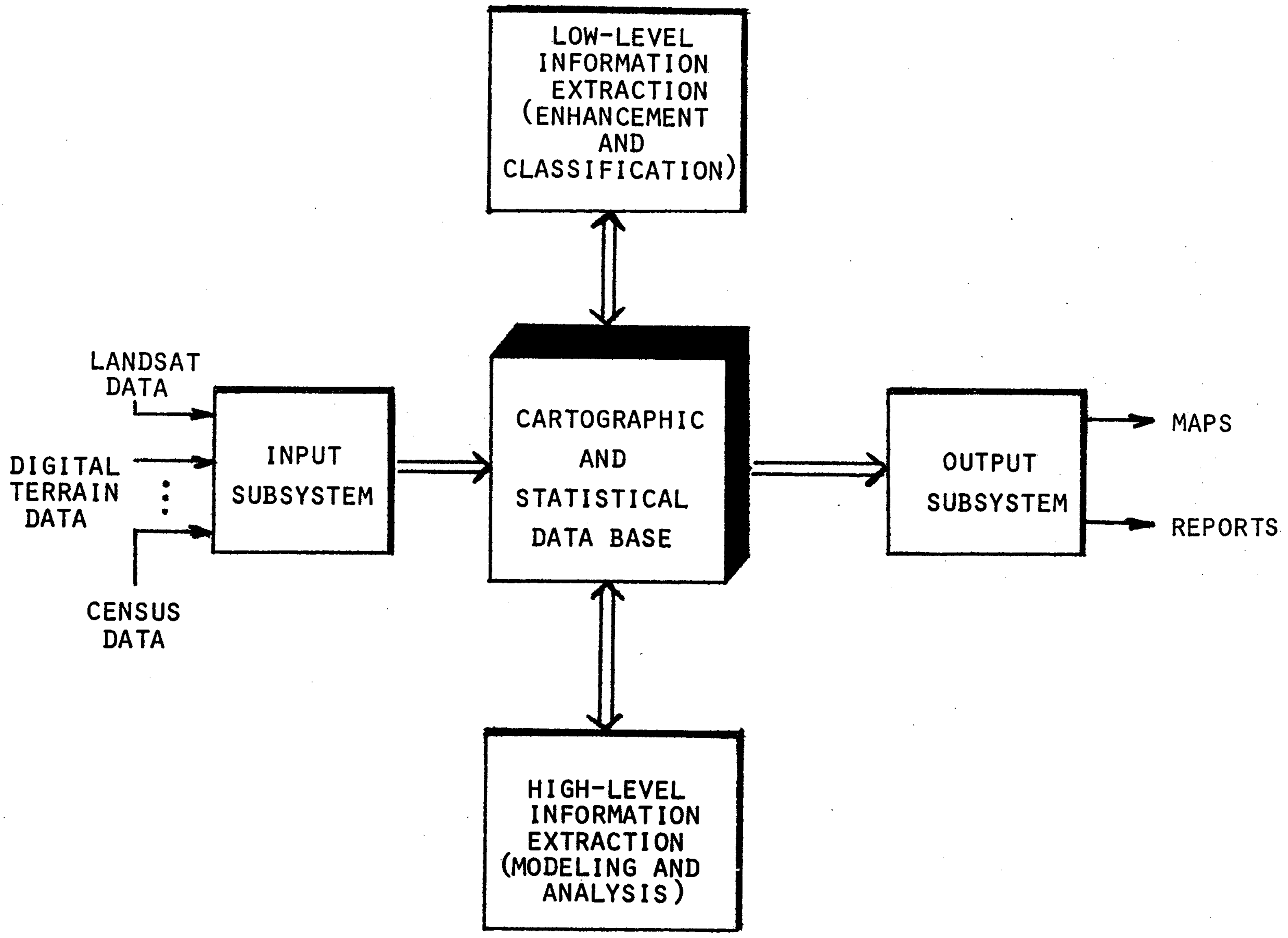

LOW-LEVEL INFORMATION EXTRACTION (ENHANCEMENT AND CLASSIFICATION)
LANDSAT DATA
DIGITAL TERRAIN DATA
CENSUS DATA
INPUT SUBSYSTEM
CARTOGRAPHIC AND STATISTICAL DATA BASE
OUTPUT SUBSYSTEM
MAPS
REPORTS
HIGH-LEVEL INFORMATION EXTRACTION (MODELING AND ANALYSIS)

Figure 1. Conceptual organization of a georeferenced information system.

on logical combinations of data to yield information for site selection or suitability analyses. Modeling to assess effects of natural or man-made events or to predict future effects is an important aspect of high-level information extraction. The results of these steps may also be returned to the data base to be used in later analyses.

In the output subsystem, information is prepared for hard-copy output in a number of image and tabular formats, as required by the user of the information for the particular applications.

The fundamental difference between "low-level information" and "high-level information" (and the operations which deal with them) is related to the degree of refinement of the data inputs. Low-level information is usually drawn from raw measurements or data on which very little processing has been performed. High-level information is often derived by applying numerous operations to the input data, perhaps even combining several forms of lower-level information. In general, the output of a processing operation may be said to be at a somewhat higher information level than the inputs to the operation. The distinction between "low-level" and "high-level" is by no means clear-cut, but it is convenient to subdivide the universe of information extraction operations into two categories such that operations in one category have more in common with each other than with operations in the "other" subset.

The COMTAL Vision One supports the operations of the high-level information extraction subsystem and the output subsystem by permitting the user to view both the progress and the final results of these operations. Monitoring the operations in this way often enhances the user's insight with respect to both the data and the processing operations and may also prevent costly errors from going undetected until the final hard-copy results are produced.

The following paragraphs describe in somewhat more detail the operations performed in each of the four processing subsystems, thus providing an overview of the system and the perspective from which the reader may understand the following detailed documentation of individual processors.

The Input Subsystem

(to be written later)

Cartographic and Statistical Data Base

(to be written later)

Low-level Information Extraction

The low-level information extraction subsystem incorporates principally two types of operations: image enhancement and classification. Some image enhancement operations are for the purpose of enhancing the image nature of the data to facilitate visual interpretation by the data analyst. They may emphasize certain features in the image such as edges or lineaments, or combine images in special ways such as taking ratios of various spectral bands. Other image enhancement operations improve the quantitative characteristics of the data, often utilizing available ancillary information concerning the data or the scene from which it originated. Examples include line-by-line calibration, illumination level normalization, and atmospheric correction.

The purpose of classification operations is to assign the pixels in the image to classes which have particular meaning in the context of the application at hand. The classes may be fairly general, e.g., "urban" and "nonurban." Or they may be very detailed, e.g., species of agricultural crops. In any case, however, the classes must be meaningful to the user and characterizable by the data in the data base.

The classificaton process may be viewed as a series of five steps which are common to the most simple as well as the most sophisticated classification schemes:

1. Pictorial display of the image data,
2. Characterization of the training classes,
3. Classification of the area of interest,
4. Pictorial and tabular display of the classification,
5. Evaluation of the classification

The first step is pictorical display of the data, or representation of the data in image format so that the analyst may become familiar with the data, assess its quality, and select the specific areas to be analyzed. The COMTAL provides a means for viewing up to three images simultaneously in a congruent format by displaying a truecolor image of three geometrically registered data sets. The most critical part of the analyst's task at this point is to select a sample of the data that will be used to design ("train") the classifier so that later each data element or pixel may be assigned to a specific class. The analyst makes use of all reference data available, such as aerial photographs and maps, to select the training points and to be sure that the sample chosen is representative of all the classes that are present in the area.

There are two major approaches to defining the training classes into which the entire data set will be classified: the supervised approach and the unsupervised approach. For the former, the analyst chooses samples from homogeneous areas of known identity, e.g., samples of deciduous forest, or coniferous forest, of bare soil, of clear water or turbid water, and so on, until all classes are represented in this way. In the non-supervised approach, the analyst selects hetereogeneous areas that together represent the important classes present in the scene and relies on a clustering processor to separate the training data into spectrally differentiable classes.

The second step, characterization of the training classes, draws heavily on both the analyst's understanding of the area and on the analysis software available. Once the samples are selected they are submitted to a clustering processor, through which the pixels selected are subdivided into distinct classes based on their values in two or more data channels. The clustering algorithm finds the natural clusters or groupings within the data and characterizes these classes in terms of statistical parameters such as the means and covariances of the classes. After the initial classes are determined through clustering, the analyst must evaluate and possibly refine the cluster classes to be sure that they together represent all the ground cover materials of interest and that they are distinguishable by the classifier. Several complex criteria may be used to measure distinguishability among classes, however the concept of interclass distance, based on the distance between the class means (i.e., how far apart in measurement space the class centers lie) and the spread or variance of the data associated with a single class in relation to near-by classes, is the fundamental measure of distinguishability. Higher classification accuracy is achieved when classes do not overlap in measurement space. Characterizing the training classes is the most critical step in the classification process, one which is greatly facilitated when the analyst can monitor and interact with the data and the analysis processes.

In the third step each pixel in the data set is assigned to one of the previously defined classes. In this step the computer-implemented classifier does practically all the work once it has been given two kinds of information: the coordinates of the area to be classified and the statistical description of the training classes. The classifier itself is a decision-making algorithm that assigns each pixel in the scene to a class according to a certain classification rule. There are several classification rules that give their names to the classifier; minimum distance and maximum likelihood are two frequently used classification rules.

Once the classification is complete, we move to step four, in which the classification is presented pictorially or in tabular format, depending on the needs of the user. In the context of the georeferenced information system, these output products often serve as intermediate tools for the analyst to evaluate the accuracy of the results. Determining classifier performance over a set of test areas is an especially effective means for quantifying the accuracy of the results.

Once the analyst is satisfied that the results of the classification are adequate to meet the requirements of the application, these results may be stored in the cartographic data base for use in further information extraction operations.

High-level Information Extraction

(to be written later)

Output Subsystem

(to be written later)

USER'S GUIDE TO CLUSTER ON THE COMTAL VISION ONE

An essential aspect of any pattern-recognition based approach to classification (supervised or unsupervised) is the development of training statistics. These statistics characterize the training classes and enable the classifier to assign each point in the data to one of the classes.

The Cluster processor provides one method of determining the sets of multidimensional data vectors to be used to represent the training classes. When applied to the training samples (usually a subset of the data to be classified), this processor isolates spectrally similar "clusters" of data vectors from which the training statistics may be computed. When the training statistics are determined in this manner, based on spectral similarity, the training classes are often referred to as "spectral training classes" or simply "spectral classes."

Implemented on the COMTAL Vision One, the Cluster processor allows the data analyst to interactively develop training classes by applying an iterative clustering algorithm to multivariate image data. The area processed may consist of up to three channels of data (on a four-image-plane system), 512-by-512 pixels in extent. The results are stored in an image plane as a classification map (also called a cluster map) which may be viewed in black-and-white or pseudocolor, permitting the analyst to make a visual evaluation of the results. The classification map may also be transferred back to the host computer for further processing such as the computation of training class statistics.

GENERAL DESCRIPTION OF THE CLUSTERING PROCESS

The first step in defining spectral training classes in a data set is to identify the natural spectral groupings of pixels in a sample of those data, that is, to find the clusters or groups of pixels that occur together in measurement space. Pixels derived from similar materials on the ground tend to group together in measurement space; for example, data vectors from clear water will usually group together in measurement space, as will those from turbid water, deciduous forest, urban areas, etc. The procedure for locating these clusters is implemented in the processor named "Cluster."

How can a computer find clusters in the training sample? Basically, the Cluster processor used a "guided trial-and-error" approach to assign the pixels in the image to disjoint classes. The objective is to make the assignment in such a way that pixels within any given class or "cluster" are as similar as possible while the pixels in different clusters are as different as possible. Many iterations of the assignment process are made and on each the objective is more closely met until on two successive iterations no change occurs, i.e., the process reaches "convergence."

The Cluster processor first requires the following inputs: the analyst specifies which pixels are to be submitted to the processor for clustering, how many clusters are desired, and how many iterations the processor should run through. The final input is several pixels (one per cluster), selected so that together they are representative of the spectral variations in the scene. This initializes the cluster centers.

The processing then follows a sequence of operations which is repeated until the stated maximum number of iterations or convergence is reached, whichever occurs first. The sequence, summarized in Figure CLU-1, can be described as follows:

1) the processor calculates the multidimensional distance between each pixel in the sample and each specified cluster center and assigns each pixel to the cluster whose center is nearest in measurement space;

2) for each cluster, the mean vector of measurements for all the pixels currently assigned to that cluster is calculated, and the mean vector becomes the new cluster center;

3) if all the new means are identical with the previous means (meaning that convergence has been reached), the Clustering operation halts; otherwise, the processor sets the cluster centers equal to the new means, checks to see if the maximum number of iterations has been reached and either stops or repeats steps 1) and 2).

This process of distance calculation, point assignment, and migration of the means can be demonstrated graphically (Figure CLU-2). The shaded area in Figure CLU-2(a) is the location in two-dimensional measurement space (coordinate axes x_1 and x_2) of the measurements in the training sample. The two crosses mark the initial locations of the cluster centers, with a separating boundary equidistant from the centers. Figure (b) shows the new locations for the cluster centers, each determined by calculating the mean of all the data points in the corresponding cluster. The boundary between the clusters again lies equidistant from the new centers. With the second iteration, the cluster centers again move to a new location in the measurement space, shown in Figure (c), and the boundary is re-positioned. The final iteration, Figure (d), shows no change in the location of the cluster centers from the previous iteration and no change in the position of the decision boundary. Convergence has been reached.

The distance measure used for determining cluster assignments is L_1 distance. In essence, L_1 distance is the sum of the individual component distances. As shown in Figure CLU-3, the L_1 distance between points A and B in two-dimensional feature space is calculated by determining the difference between a_1 and b_1, the difference between a_2 and b_2, and adding the two values. The procedure can easily be extended to accommodate higher-dimensional situations.

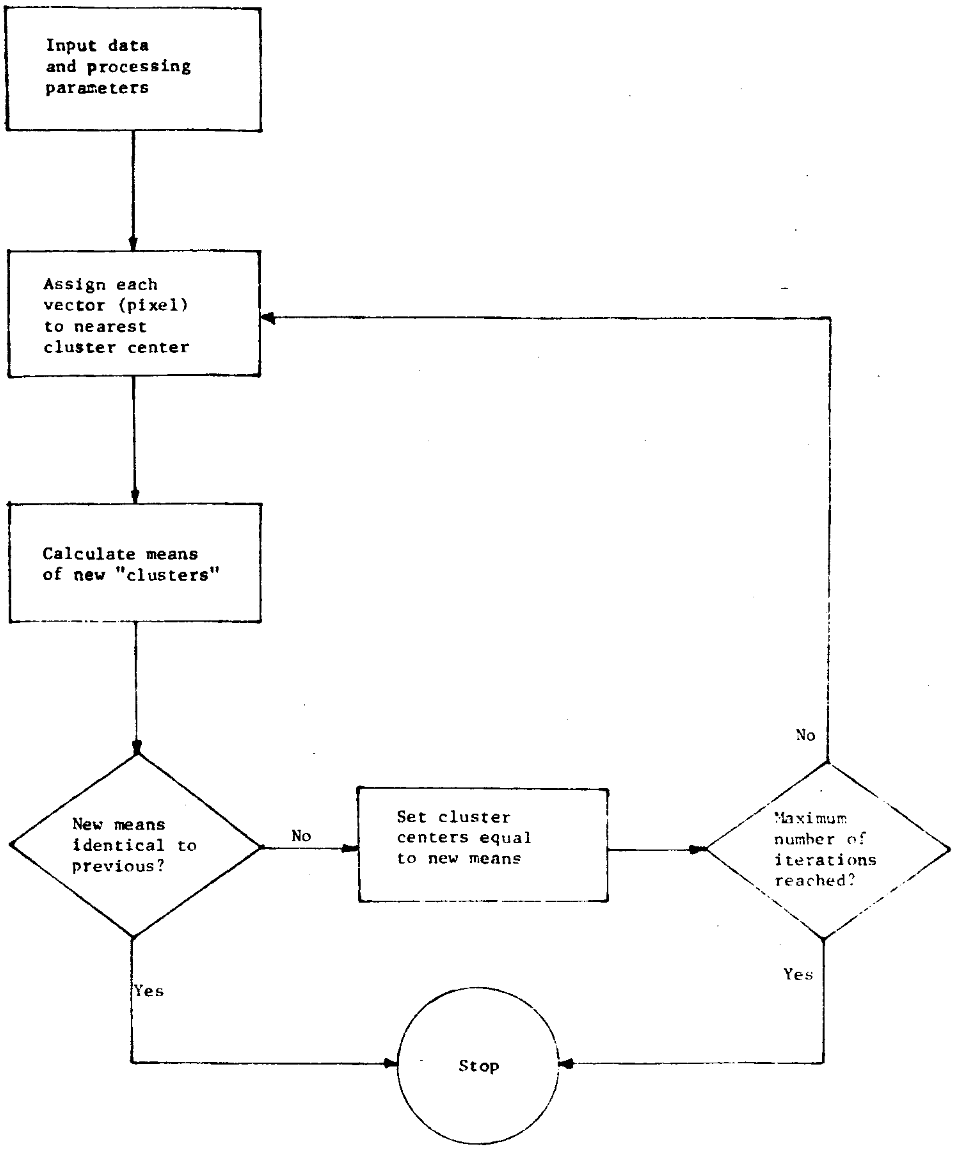

Input data and processing parameters
Assign each vector (pixel) to nearest cluster center
Calculate means of new "clusters"
New means identical to previous?
No
Set cluster centers equal to new means
Maximum number of iterations reached?
No
Yes
Yes
Stop

Figure CLU-1. Basic flowchart of clustering algorithm.

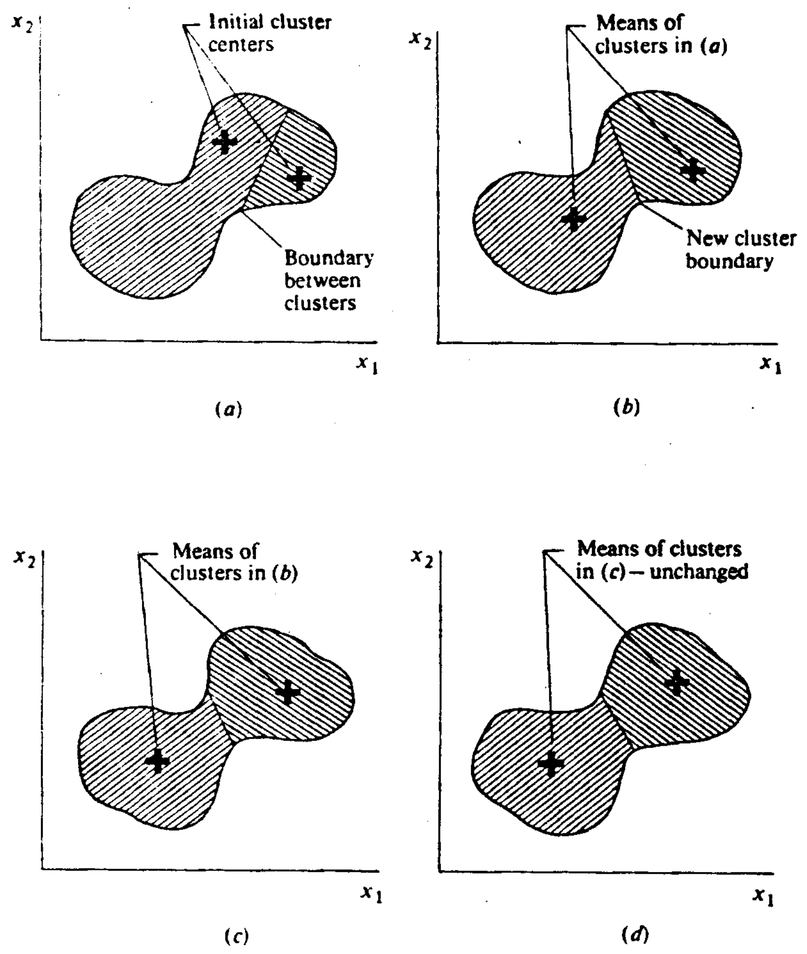
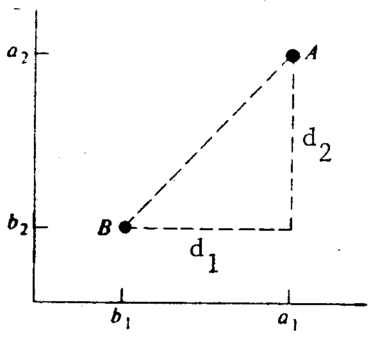


Figure CLU-2. A sequence of clustering iterations. (from Swain and Davis)



L_1 distance

$$D_{AB} = \sum_{i=1}^{n} |a_i - b_i|$$

In two-dimensional space, $n = 2$ and $D_{AB} = d_1 + d_2$.

Figure CLU-3. Interpoint distance measure using L_1 distance.

The clustering algorithm is called a nonsupervised classifier because it groups or classifies pixels strictly on the basis of their multidimensional data values. Neither the locations of the pixels relative to one another (spatial information) nor the actual surface materials that the pixels represent are considered when the algorithm determines the clusters. The result of the processing is a classification of each data vector in the sample into one of the clusters, with the decision stored as a classification map. Figure CLU-4 shows an example of three channels of input spectral data and a classification map derived from those data.

USER INPUTS TO CLUSTER

To run the Cluster processor, the analyst must first enter the image data set into the COMTAL image memory, call the program and then provide the required information: (a) the area in the image data on which the processor is to operate, generally a subset of pixels from a larger scene; (b) the maximum number of clusters that the processor should produce from the data; (c) the locations of pixels to be used as initial cluster centers, one per cluster; and (d) the maximum number of iterations through which the program should run.

Procedures for entering the data, setting up the display and running the processor are discussed in detail below.

1. Enter the image data

   Data to be clustered must be in the image memory of the COMTAL. If the system has three image planes, then two channels of data may be clustered; if there are four image planes available, then up to three channels of data may be clustered. One image plane must be available to receive a new image, the results of the classification performed by the Cluster processor.

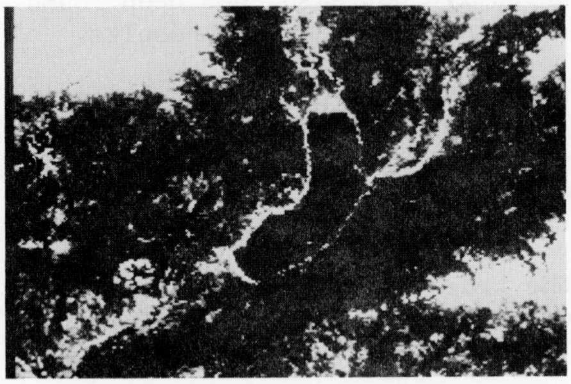
   The following sequence of operations will set up the image data in an optimal way:

   A. Initialize the COMTAL: type (SHIFT RS) R (see COMTAL Notation Conventions at the end of this document for interpretation of the notation conventions adopted.)

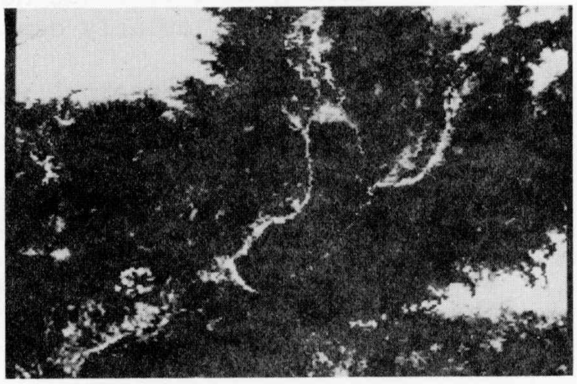
   B. Load the first image to be used in the clustering into Image Plane 2; enhance the displayed image by histogramming the image and using the equalized histogram as Function Memory 2. Display Image 2 with Function Memory.

   C. Load the second image to be used in the clustering (e.g., the second channel of a multispectral image) into Image Plane 3; enhance and display Image 3 as above.

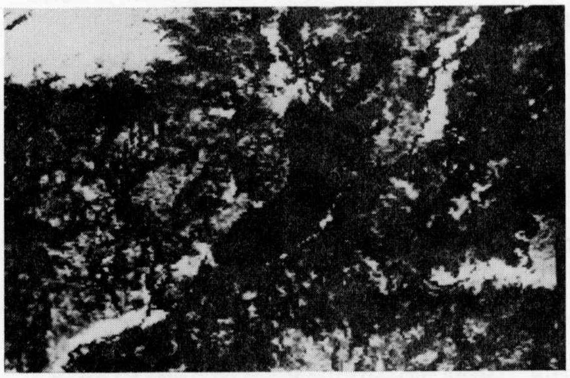
   At this point two channels of data have been specified for viewing on the monitor. If the system has four image planes a third image



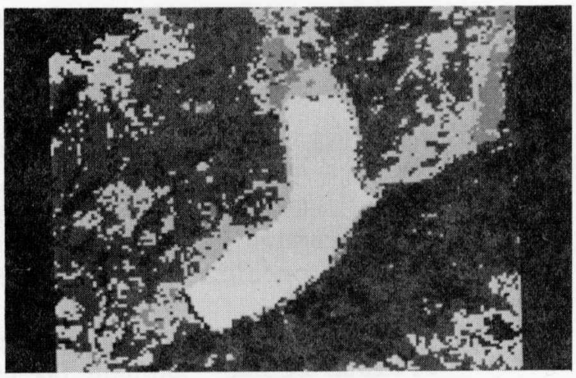
(a) Feature 1: Landsat data in .5-.6 μm band



(b) Feature 2: Landsat data in .6-.7 μm band



(c) Feature 3: Landsat data in .8-1.1 μm band



(d) Classification of three-channel image data into 7 classes using the Cluster processor.

Figure CLU-4. Display of Landsat data in three channels and an example of a Cluster results image for the same area.

may be entered in the same way for display and processing if desired.

Note: The Cluster processor allows the input images and the results image to be in any of the available image planes. The arrangement followed here is illustrative but also has been found to be particularly convenient for viewing both data and results.

D. Set up a truecolor image that allows you to look at the input images simultaneously. An appropriate sequence of commands is:

```
Assign Truecolor 8 red 4 green 3 blue 2
ck N
```

After the input images are enhanced and the truecolor image is displayed, the next step is to call the Cluster processor and provide the information requested by the prompts displayed on the monitor.

2. Call the Cluster processor

Access the COMTAL Utilities Program by typing on the host computer:

```
RUN COMTAL
```

A menu provides a list of options; you should select:

```
CLUSTR
```

If the image display is set up as you wish to have it during the following interactive session, type on the COMTAL:

```
EXecute Code
```

You will be prompted for required information.

Note: While Cluster is running, the command keys are inoperative, and the keyboard may be used only to respond to the questions asked or to end the processing. The display cannot be changed.

3. P-ovide requested information

Prompts appear on the COMTAL requesting you to supply information as needed.

A. Specify the input and output image planes

The image plane in which the results are to be stored should be different from the image planes containing the data to be clustered. The input images ("features") may be all or any subset of the image planes loaded previously.

Note: The processing time required per iteration is proportional to the number of input images.

B. Specify the area to be processed

The target appears on the screen and you are prompted to specify the area to be clustered. Use the trackball to position the target at the upper-left corner of the rectangular area to be clustered and depress function switch 1 (fs 1). Position the target at the lower-right corner of the area and depress (fs 2). The rectangle shown on the screen outlines the area to be clustered. It may be modified by moving the target and depressing the appropriate function switch (upper-left corner: fs 1; lower right corner: fs 2). To accept the area defined by the rectangle, hit the space bar.

Note: The processing time required per iteration is proportional to the size of the area clustered.

C. Enter the maximum number of clusters

When prompted, type an integer between 2 and 16 (inclusive), followed by a space bar.

Estimating the right number of clusters is a critical part of the analyst's job. To make a good estimate, first look at the imagery and at maps and photographs of the area to decide how many information classes occur in the outlined training area. Information classes to count are those whose identity is important to the analysis, such as all primary earth surface materials, perhaps with sub-classes for features of particular importance. Next, look closely at the displayed truecolor image to find characteristic samples of each of the classes. Based on the appearance of the imagery, you may need to adjust the number of classes, decreasing it to account for the possible spectral similarity of different materials and increasing it when obvious subclasses of materials are present. If too few clusters are requested, spectrally different ground covers may be lumped into the same cluster, making the clusters ineffective as training classes. If too many clusters are requested, the spectral classes may be difficult to correlate with information classes, an important step later in the process. When uncertain, it is preferable to ask for too many rather than too few clusters since later some of the clusters may be deleted or combined.

D. Select initial cluster centers

The target will appear on the screen and a prompt will ask you to select the initial cluster centers. Move the target to the first pixel and then hit the space-bar to input the pixel. Pixels may be specified anywhere on the image, either inside or outside of the rectangle surrounding the area to be clustered.

Processing will be more efficient if the initial cluster centers are chosen carefully to represent each spectral class expected. While the processor will run even if the initial cluster centers are not chosen in this way, convergence will be reached sooner if the initial cluster-center locations approximate the final ones. Any identical cluster centers will be replaced by a single cluster center, thereby reducing the total number of clusters in the result.

E. Request number of iterations
You are prompted to specify the maximum number of iterations for the clustering sequence, any integer from 1 through 99, followed by a space. If the processor reaches 100% convergence before it reaches the specified number of iterations (that is, if the cluster centers do not change location in feature space and there are no shifts in pixel assignment on two successive iterations), the processor will stop.

RUNNING CLUSTER

Hitting the space bar after typing the desired number of iterations starts the actual clustering operations. The number of the iteration in progress is displayed on the monitor as well as the number of pixels that have changed class assignment during the previous iteration. (The value that appears there during the second iteration is the total number of pixels in the area being clustered.)

The length of time required to reach the maximum number of iterations is a function of the number of clusters requested, the number of pixels submitted to the processor, and the number of input images; as any of these quantities increases, processing time increases. Moveover increasing any of these quantities tends to increase the number of iterations required to reach 100% convergence.

If you want to stop the processor, depressing fs 1 will cause the Cluster processor to complete the iteration in progress and then return to the point at which the number of clusters is requested; alternatively, depressing ESC will stop the clustering immediately and return to the same point. If no further clustering is desired, you may exit Cluster by simply hitting the space bar in response to the question.

After exiting Cluster, you may run another cluster job on the same data set by typing EXecute Code on the COMTAL; this is possible as long as system reset has not been executed and no other processing code has been loaded from the host computer.

OUTPUTS FROM CLUSTER

With each iteration, the Cluster processor creates a new image of the processed portion of the input image, storing the results in the results image plane you previously designated for that purpose. Each new results image will be written over the previous results image. The results image will appear in the same screen location as the input data and will contain data values from 1 to the number of clusters; all other pixels in that image plane are set to zero.

The results image may be enhanced in several ways for further visual analysis; for example, you could histogram the sub-image and equalize the related function memory, or you could use an integer function to create a suitable function memory as follows:

```
Set INteger function 1
INteger function 1 = Integer (X * Constant 255 / Constant NC #)
Function memory 1 = Integer (integer function 1 #) #
```

where NC equals the number of spectral classes in the clustered data. The cluster image can also be colored effectively by using pseudocolor memory.

The final results image may also be read back to the host computer by means of the COMTAL Utilities Program.

A hands-on self-guided tutorial introduction to Cluster is available. See "A Hands-on Experience with Cluster on the COMTAL Vision One."

References

P.H. Swain and S.M. Davis, 1978. Remote Sensing, The Quantitative Approach, pp. 177-184, "Clustering."

COMTAL Notation Conventions

Notation	Connotation
Display Graphic 1 INteger function 1 = Integer (X + Y #) R	Commands you enter by typing only the capital letters, numerals, and operands, each followed by a space. Lower case letters, equal signs, and parentheses are added by the system.
#	Add an extra space. Ex: for INItialize PSeudocolor memory #, user would type INI, space, PS, space, space.
___	(Underline) A single key represented by a group of letters or numbers; e.g., ESC Escape Key fs 1 Function Switch 1 ck A Command Key A
n,g	(Lower case letters) A number that can vary, depending on the size of the system. n refers to an image number or a memory area related to an image; g refers to a graphic number.
* /	multiplication sign division sign

A HANDS-ON EXPERIENCE WITH CLUSTER ON THE COMTAL VISION ONE

The following exercise leads you through a hands-on, tutorial session with the Cluster processor using a multispectral data set from the Landsat satellite.

Before doing this exercise, you should have read the following materials, all a part of this documentation for Cluster:

a) Introduction to the Georeferenced Information System
b) User's Guide to Cluster on the COMTAL Vision One.

With this background describing what Cluster contributes to a Georeferenced Information System, you should be ready to proceed. It is assumed that you are already familiar with the concepts of COMTAL-based image processing, with loading data into the COMTAL image memory and issuing basic commands on the COMTAL. (See Image Processing on the COMTAL Vision One Series - A Beginner's Guide by S.M. Davis, D.M. Freeman, and P.H. Swain, 1981, to acquire this background.)

When you have finished this exercise (including the user documentation listed above), you should be able to successfully carry out the following:

1. Describe the role of clustering in the analysis of multispectral data.

2. Explain with the aid of a sketch how the clustering algorithm works; include in your explanation the following terms: initial cluster centers; migration of cluster centers; iterations; class means; L_1 distance; and convergence.

3. Define the following terms: feature space, cluster class, cluster map, and convergence.

4. Call and run the Cluster processor, selecting an area to be clustered, specifying the number of clusters to be formed, providing the locations of the initial cluster centers, and stating the number of iterations to be used.

5. Display the results of the Cluster processor, and, with the aid of reference data, infer the information class most closely associated with each spectral or cluster class.

6. Transfer Cluster results back to the host computer for storage or further processing.

Step 1 - Load the data into the COMTAL image memory

The loading instructions given here are for use with the COMTAL Utilities Program installed on the PDP-11/34 system at Purdue/LARS and may require modification for other systems. See the COMTAL Notation Conventions at the conclusion of the "User's Guide to Cluster."

Initialize the COMTAL by typing on the keyboard:

```
        [SHIFT RS] R
```

Load image data from the PDP using the COMTAL Utilities Program. On the PDP, enter the following commands, each followed by CAR RET:

```
        RUN COMTAL     (to call the COMTAL Utilities Program)
        DATA           (to request the routine for transferring data)
        DATATO         (for subroutine that sends data to the COMTAL)
```

When requested, load the following images into the image planes shown:

```
        [300,300]VALLECITO.CH1    into Image 2
        [300,300]VALLECITO.CH2    into Image 3
        [300,000]VALLECITO.CH4    into Image 4
        ESC                       (to return to sub-menu)
        END                       (to return to main menu)
```

Image 1 is left unused so that it can be used later to receive the results of the clustering, an arrangement that will be helpful in interpretation and comparison of the images.

Step 2 - Call the Cluster processor

The main menu of the COMTAL Utilities Program gives you access to the Cluster processor. On the PDP, type:

```
        CLUSTR
```

With this command, the code for running Cluster is transferred to the COMTAL, and you may log off the PDP.

The PDP screen will prompt you to type "EXECUTE CODE"; however, before doing this you should enhance the image data and set up a truecolor image made up of the three images. (If you wish to do so now, you may alter Function Memory 1 to enhance the Cluster results; specific instructions for doing this are given as part of Step 6, a more convenient time to perform this operation.) These enhancement steps may not be carried out while Cluster is running; the command keys are inoperative then, and the keyboard may be used only to respond to specific prompts. Enhancements must therefore be done before you type "execute code" or after the processing is complete.

Step 3 - Enhance the image data

Begin this operation by histogramming each of the three images containing data and equalizing the histograms to create the respective function memories. You may do this by entering the following commands on the COMTAL:

```
Function memory 2 = Histogram of image
Equalize Function memory 2
ck H
```

Repeat this sequence for Images 3 and 4, and then display each image in turn to become familiar with the general characteristics of the data. Remember that there is no command key that allows you to display Image 4. You must instead use the commands:

```
Display Image 4
Add Function memory 4
```

Create a truecolor image now, using images 2, 3, and 4. This will allow you to see the three channels of data simultaneously as a color composite. If you wish to display the scene with colors that approximate those of a color-IR photograph, make the following color assignments:

```
ASsign Truecolor 8 red 4 green 3 blue 2
```

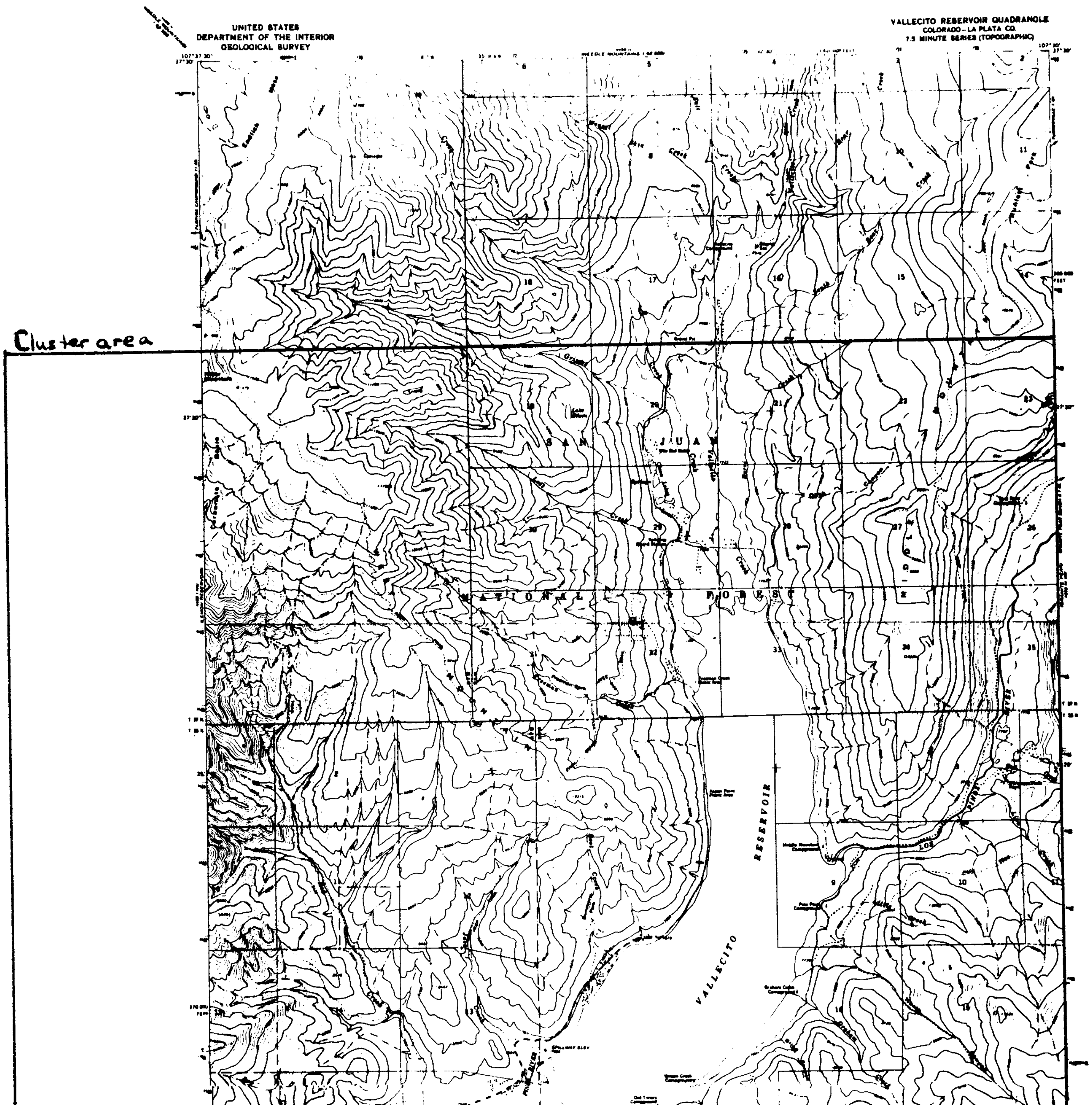
Use ck N to see the truecolor image with function memories added.

Step 4 - Become familiar with the data and region

Look at the three enhanced images separately and also together in a truecolor image to gain a general sense of the quality of the data and the primary features of the scene. Answer the following questions:

a) Is there any obvious noise in any of the images?
Look for both linear patterns that may have been caused by the sensor and for other abberations in the data, including clouds and cloud shadows. Any significant amount of "noise" may degrade the clustering results.

b) What are the major geographic features of the imaged area?
Several kinds of reference data are included to help you answer this question. A portion of a U.S.G.S. topographic map, Figure 1, shows the part of the scene near Vallecito Reservoir, the largest body of water. Figure 2 is a Level 1 Cover Type map of the same area showing the primary materials


UNITED STATES
DEPARTMENT OF THE INTERIOR
GEOLOGICAL SURVEY
VALLECITO RESERVOIR QUADRANGLE
COLORADO - LA PLATA CO.
7.5 MINUTE SERIES (TOPOGRAPHIC)
Cluster area
SAN JUAN
NATIONAL FOREST
VALLECITO
RESERVOIR

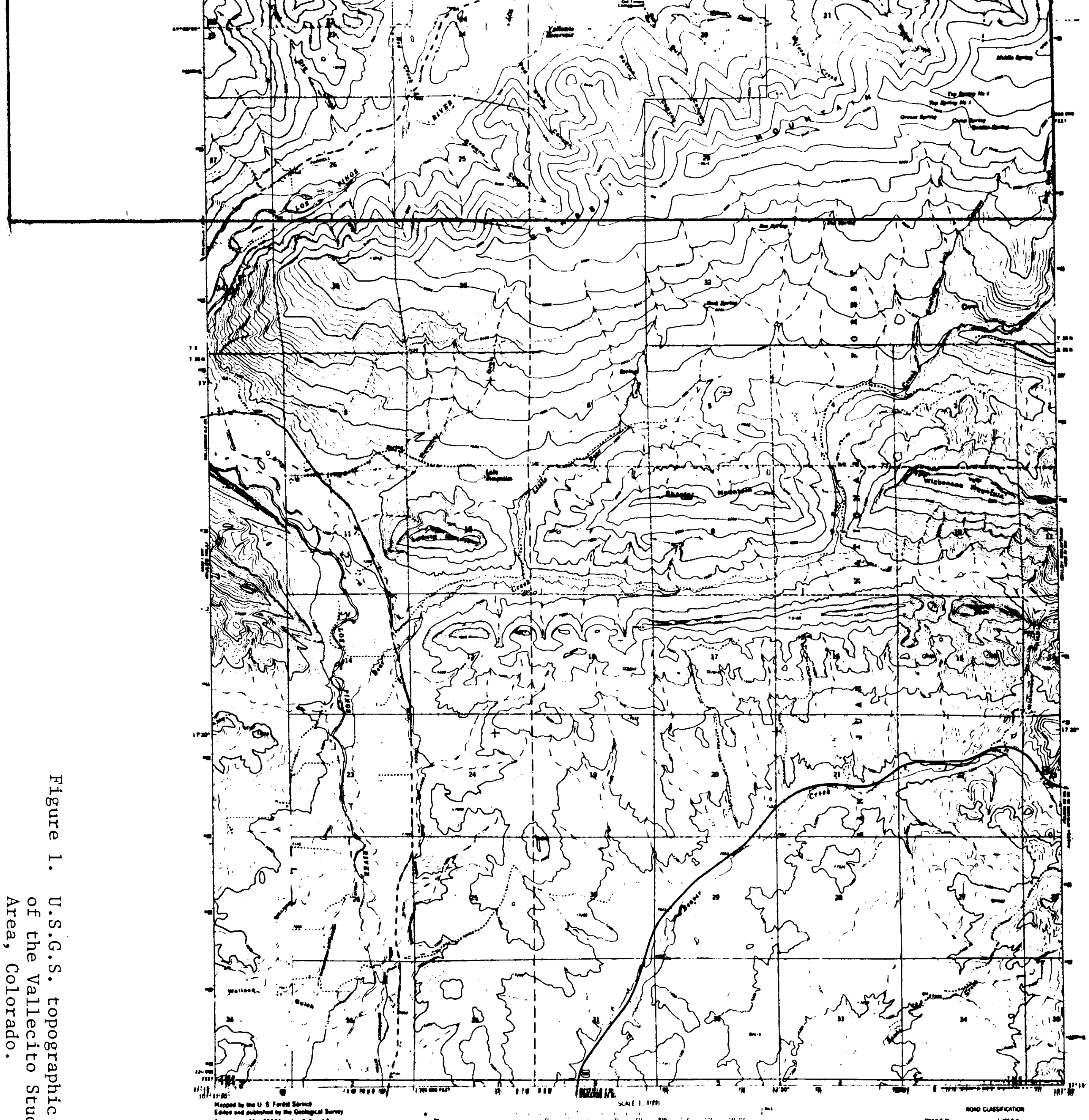


Figure 1. U.S.G.S. topographic
of the Vallecito Stu
Area, Colorado.

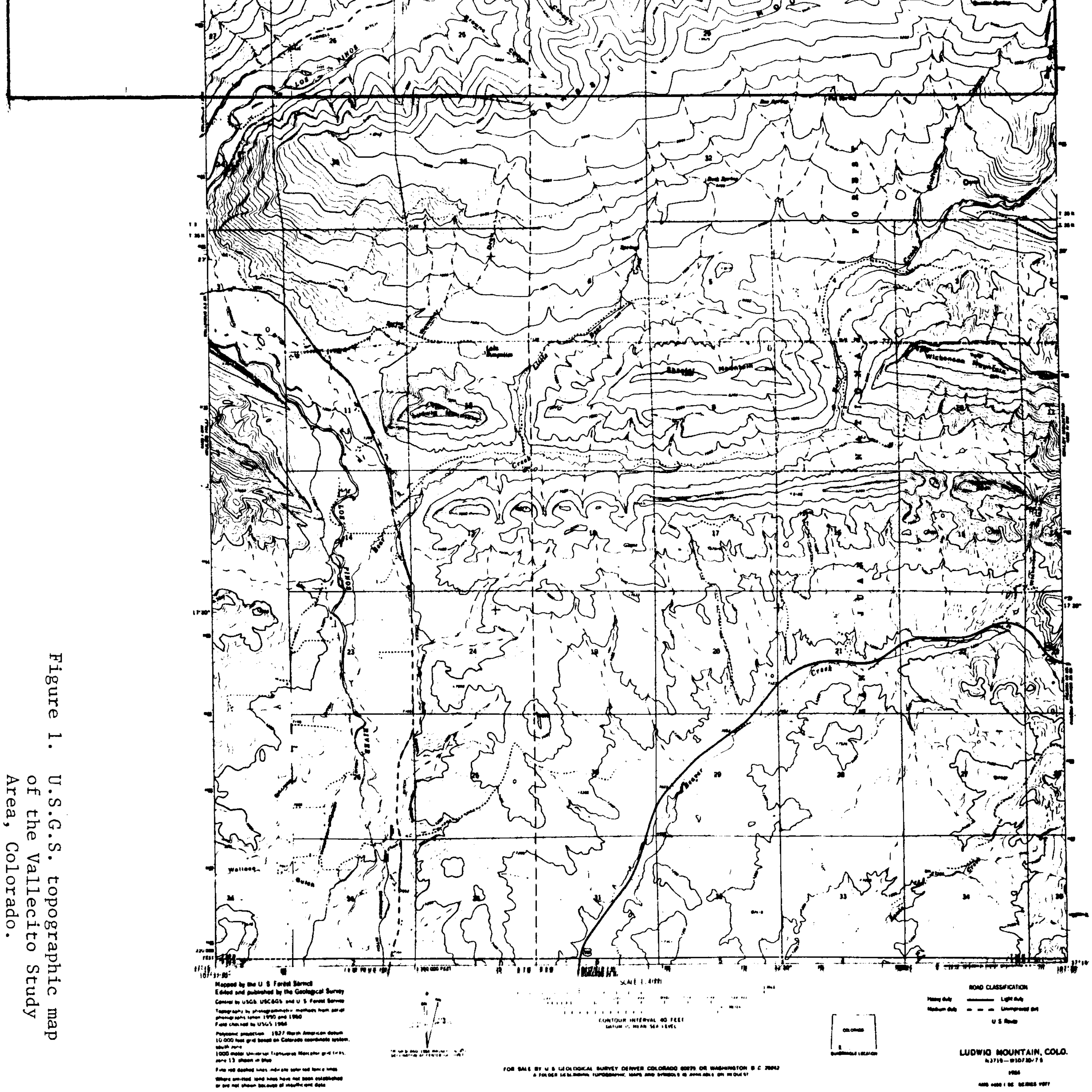
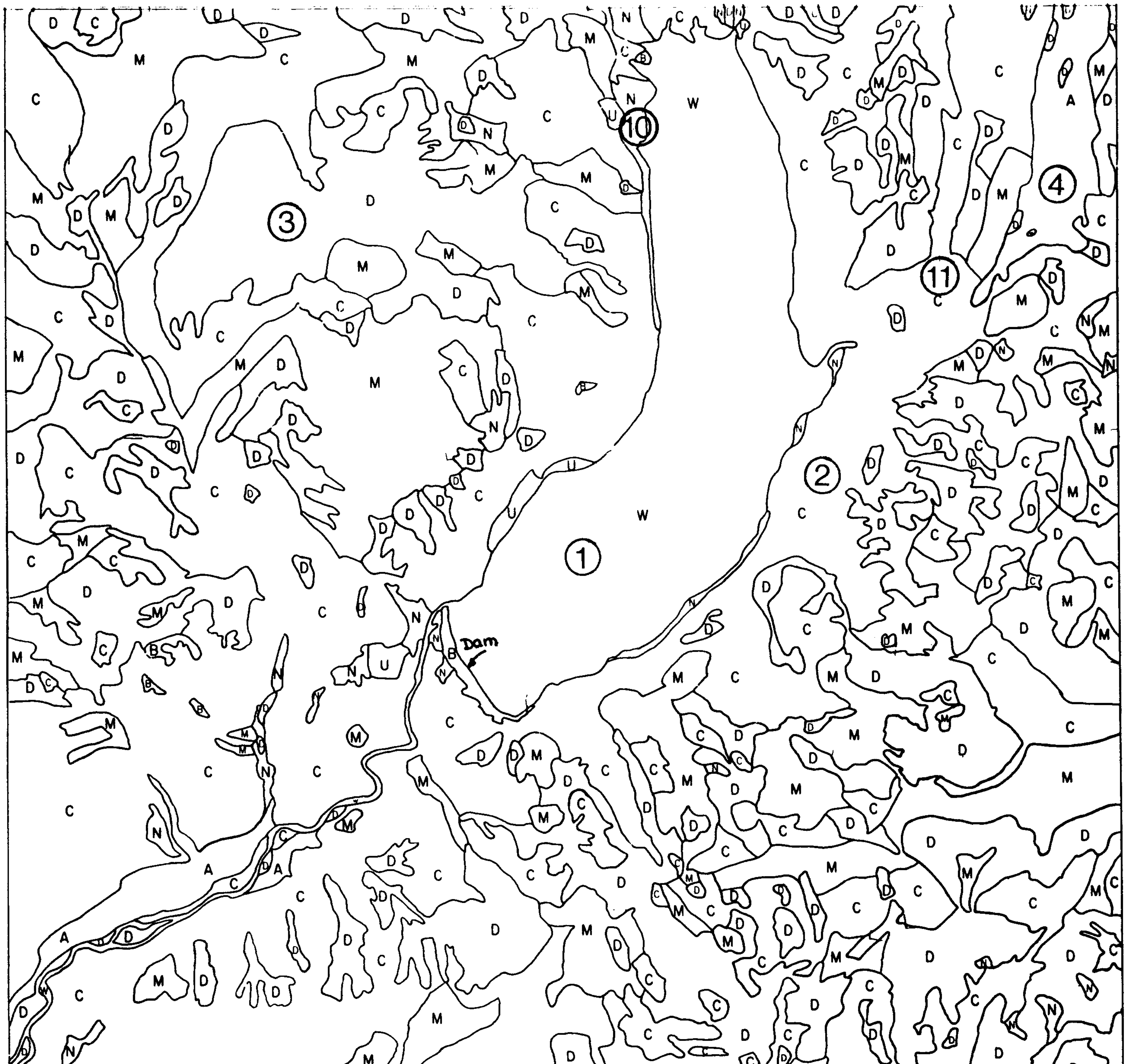

Shearer Mountain
Wickenson Mountain
Lake Pimpkin
Creek
Beaver
Wallace Gulch
Los Pinos
RIVER
CONTOUR INTERVAL 40 FEET
Mapped by the U.S. Forest Service
Edited and published by the Geological Survey
ROAD CLASSIFICATION
LUDWIG MOUNTAIN, COLO.
FOR SALE BY U.S. GEOLOGICAL SURVEY DENVER COLORADO 80225 OR WASHINGTON D.C. 20242

Figure 1. U.S.G.S. topographic map of the Vallecito Study Area, Colorado.


1
2
3
4
10
11
Dam
W
A
B
C
D
M
N
U

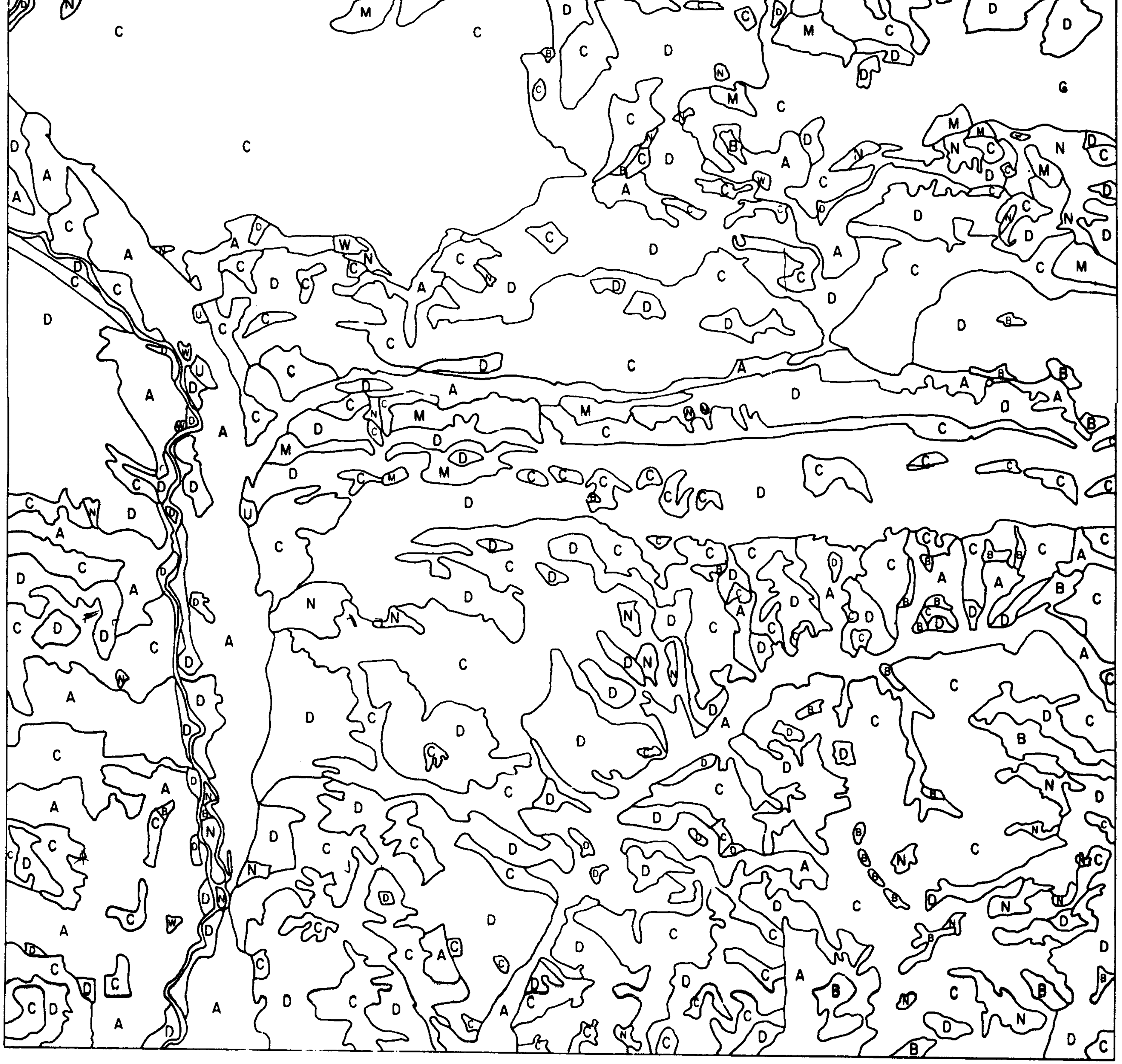


Figure 2. Level I cover type m
the Vallecito Study
produced through pho
interpretation.

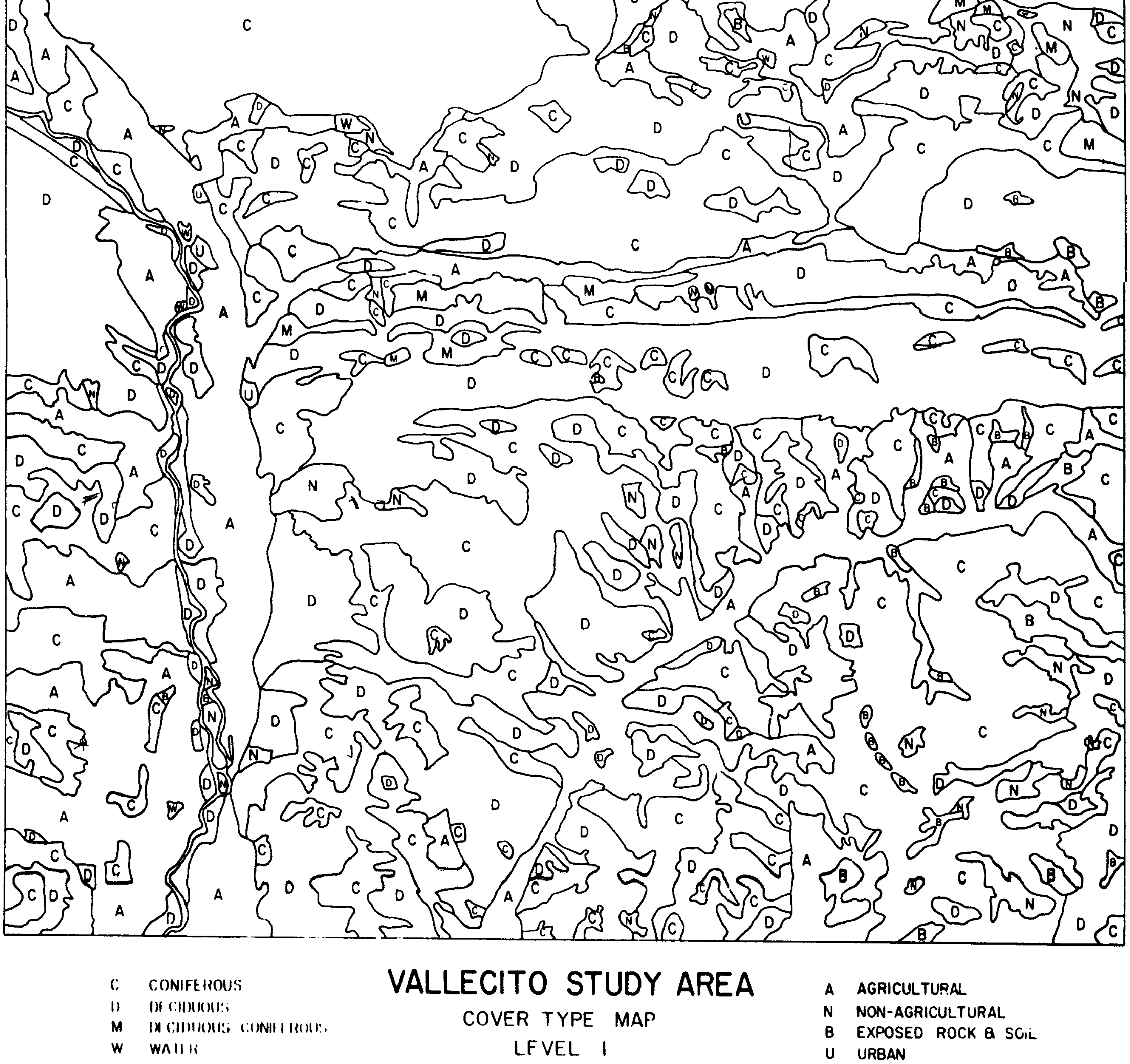


Figure 2. Level I cover type map of the Vallecito Study Area, produced through photo-interpretation.

present, e.g., deciduous forest, coniferous forest, agriculture, urban, water, etc. A color IR aerial photograph of the reservoir area is enclosed as a slide.

To enhance your familiarity with the scene, sketch the major geographic features of the area shown on the screen, providing general labels for features you can identify.

c) Are there any features in the image data which are transitory or seasonal and therefore may not appear on maps or on photographs taken at a different time? Examples may include floods, clouds and cloud shadows, seasonal vegetation changes, and so on. List here as many as you can identify.

Step 5 - Running Cluster

Now you are ready to run the Cluster processor. With the truecolor image displayed, type on the COMTAL keyboard:

```
EXecute Code
```

You will then be asked a series of questions that will control the way Cluster is to be run. For the clustering, you will use all three images you entered into COMTAL memory: Landsat Band 4 (.5-.6 μm) in Image 2; Band 5 (.6-.7 μm) in Image 3; and Band 7 (.8-1.1 μm) in Image 4. Each of these images is referred to as a feature, a pattern recognition term used to refer to the individual elements that make up a multivariate data element.

Enter the following responses to the prompts, followed by a space:

Prompt	Response
How many channels of input data	3
What image for results image?	1
What image plane contains feature 1?	2
What image plane contains feature 2?	3
What image plane contains feature 3?	4

The next prompt asks you to select the cluster area. You do this by moving the target with the trackball and hitting function switches 1 and 2 to identify the upper left and lower right corners respectively.

The first area you will cluster includes the Vallecito Reservoir, the surrounding forested area, some snow and an agricultural area. The area is outlined in Figure 1.

Move the target to the upper left corner of the area and hit fs 1; then move it to the lower right corner and hit fs 2. A rectangle will appear on the screen outlining that area. If you want to reposition either of the two corners, move the target and hit either fs 1 or fs 2, as appropriate. Hit the space bar to accept for clustering the area enclosed by the resulting rectangle.

The next prompt asks you to state the number of cluster centers you will input, in other words, how many classes should the data be divided into? There is no "right" answer to this question, but you can begin to answer this question by looking at the image and the reference data. Correlate the representation of this area on the display with its appearance on the topographic map and the cover type map, Figures 1 and 2.

Approximately how many different major classes of materials do you find in the scene? Count the identifiable major cover types; if any of the classes appear in areas exhibiting significantly different spectral responses, count them as additional classes. The snow-covered area would be an example of this. You should be able to identify between five and nine potential classes. List the ones you find.

1.	4.	7.
2.	5.	8.
3.	6.	9.

While you are looking at the classes, try to identify at least one fairly homogeneous sample of each of the classes. These samples will be used later for identifying cluster centers.

There are several ways to draw up the list, but for the sake of this exercise, we'll use these six classes:

1. water	4. agriculture
2. coniferous forest	5. snowedge
3. deciduous forest	6. snow

The location of a representative sample for each of these classes is shown on Figure 2.

In response, now, to the prompt, type 6 and then specify the six initial cluster centers by placing the target on a representative pixel, then hitting space. (Remember, the upper left corner of the target is the pixel you are pointing to.) Provide a representative pixel for each of the clusters by moving the target to the pixel and hitting space.

The last prompt asks you to set a maximum number of iterations. Set your limit at 99, the maximum allowed; in that way the processor will not stop prematurely but in all likelihood will run to 100% convergence, unless you interrupt it.

At this time the processor will begin to run. A message on the screen shows the number of the iteration currently underway and the number of points that had changed class assignment during the previous iteration. During the first iteration the number is, of course, zero, but during the second iteration the number that appears is the total number of points being processed.

Make note of that number here: ______________

If you wish, you may stop Cluster in either of two ways: hitting fs 1 will stop the processor when the current iteration is complete; ESC will stop it immediately. Both actions return you to the program where the prompt occurs "How many cluster centers?"

The ideal in clustering, of course, is to reach convergence, but, to save processing time, 99% convergence is quite adequate for most analysis of multispectral data. If you made note above of the total number of points being processed, figure out what 1% of that amount would be. When the number of points that changed class membership falls below that number, hit fs 1 to complete the current iteration and halt processing. Most likely this will occur in less than ten iterations. Alternatively, you can let the processor run to convergence. Don't be dismayed if the number of changes increases on a few iterations; the trend will continue toward convergence.

In either case, when the clustering is complete, you will be prompted: How many cluster centers? If you are not satisfied with the results, you may run Cluster again by typing in the desired number of clusters and responding to the subsequent prompts as before. If you are satisfied and want to proceed to the next step, simply hit the space bar.

Step 6 - Interpreting the results

When the clustering is complete, a classification image (or cluster map) has been created in Image 1, which you left clear for that purpose. Since there are six classes in that cluster map, the data values in Image 1 range from 1 through 6, with 0 for all parts of the image outside the map. For the six cluster classes to be distinguishable from each other on the screen, you may use an integer function to create Function Memory 1 with the best attributes:

```
          Add Annotation characters
          Set INteger function 1
          INteger function 1 = Integer (X * Constant 255 / Constant 6 #)
          Function memory 1 = Integer (integer function 1 #) #
```

The constant 6 was entered in the third line to correspond to the six classes used in the clustering. (If a different number of cluster classes had been requested, the constant should have corresponded to that number.)

To see what this step accomplished, hit ck J to display Function Memory 1. (Subtract the bottom line -- SUbtract Bottom line -- to allow you to see the whole screen.) The first seven values of the image (shown across the horizontal axis) will be displayed using the full brightness range of the display (shown along the vertical axis). You should be able to count the values from 0 through 6; see Figure 3 for further explanation.

Another way to enhance the classification image is to histogram all the values in Image 1 and equalize the histogram to create a new function memory; however, the large area of the image outside the clustered areas contains all zeros and would yield a less-than-ideal image function memory.

Add the annotation characters; hit ck G to see the enhanced cluster map.

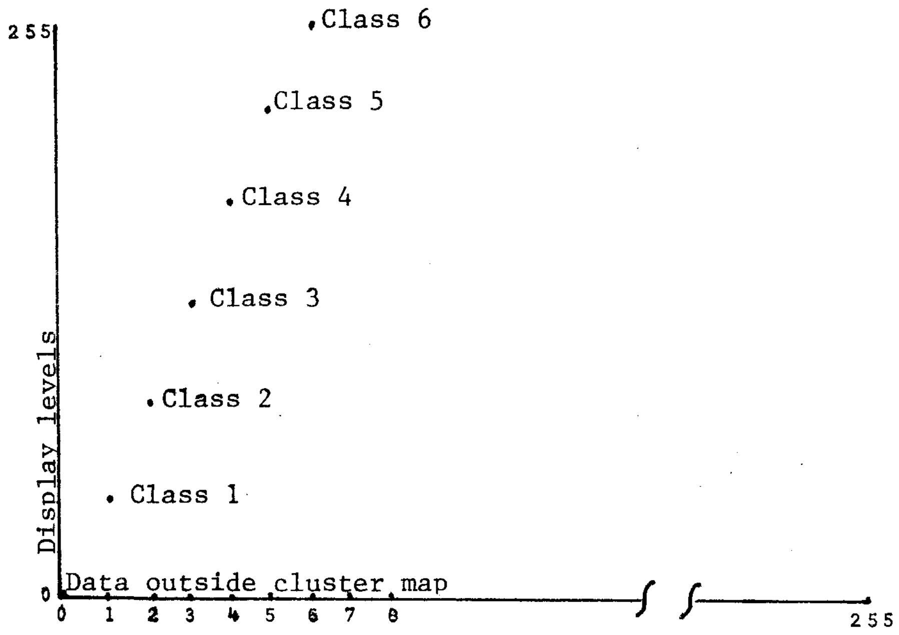


Figure 3. A portion of Function Memory 1, calculated to enhance a 6-class classification image. (Horizontal axis is exaggerated for clarity.)

Once the Cluster results image is enhanced and displayed, you are ready to begin the interpretation of the classes. Each of the six cluster classes will need to be identified with a descriptive label that relates it to cover type. To do this more easily, zoom the cluster map (Image 1) by a factor of four:

```
ROam Image; fs 1
```

Use the track ball to move the image on the screen to the upper left corner, then strike the Escape Key.

```
Display Zoom Image by a factor of 4
```

Now, with the cluster map displayed, put the COMTAL in the Dump Image mode by typing DUmp Image. The values that are shown make it easy to read the class assignment of each pixel in the map. Use the Level 1 Cover Type Map (Figure 2) and the aerial photograph (slide format) to help label each of the clusters in the chart below. It may help to compare the results image with the truecolor image. If you enlarge the truecolor image (Image 8) by the same factor and roam it so that the features have the same screen location as in the cluster results image, you can alternate viewing the two using ck N and ck G.

Another aid in viewing the cluster image is to display it with pseudocolor. Roll the three pseudocolor memories so that they are distributed along the vertical axis in the 0 to 6 range. Use ck 0 to display the pseudocolor memory, roll the pseudocolor memories to appropriate locations, then type:

```
ck D
Add Function memory 1
```

With the pseudocolor display of the results image, it is easier to locate individual occurrences of a given class. You may also use DUmp Image here with the pseudocolor image to assist in class definition, or you may use it with the truecolor image displayed. In the latter case, it allows you to read the relative spectral response in each channel and make judgments about cover materials based on this information.

Cluster Number	Cover type of input pixel	Class labels you assigned
1	Water	
2	Coniferous	
3	Deciduous	
4	Agriculture	
5	Snow Edge	
6	Snow	

Depending on where precisely you located the area to be clustered and which pixels you used to initialize the cluster centers, you should have found that the new class labels were the same or nearly the same as the original labels.

Are there any areas where the results are surprising, where something on the ground appears to be misclassified? If so, can you explain why it was classified as it was? Three specific areas that may have caused some problems require additional study:

1. How did the dam get classified? Most likely it came out as snow or snow edge. Recall that you input no cluster center to represent the dam and the highly reflective bare rock along side it. Can you explain why it was put into the same class as the snow?

2. On the west edge of the reservoir near the top there is a triangular area (#10) identified on the Cover Type Map as "non-agricultural." Which class was it assigned to? Look at the topographic map and the aerial photograph to see if you can explain why it was classified as it was.

3. According to the Cover Type Map, an area to the east of the reservoir (#11) contains coniferous forest. How was the area classified by Cluster? Look at the area on the aerial photograph and see if you can understand why it was so classified.

In comparing the image on the screen with the maps and photograph, did you notice that the geometric proportions of large recognizable features are not the same? For example, Vallecito Reservoir is more elongated on the maps than on the screen. This geometric distortion is caused by the fact that each pixel represents not a square but a rectangular area on the ground, as shown in Figure 4. This same pixel is displayed on the monitor by means of a screen area having equal length and width. The result is the vertical foreshortening of features on the display screen.

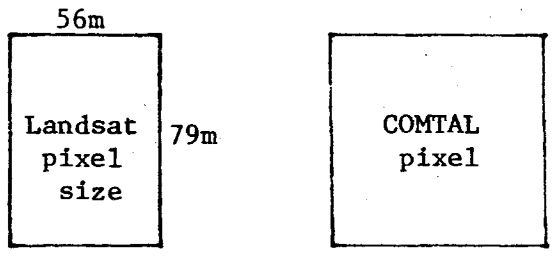


Figure 4. Area of the earth represented by each pixel of Landsat data.

<u>Step 7 - Run Cluster a second time</u>

To complete this exercise, run Cluster again using another portion of the scene.

You will first need to restore the images to their original size and position by zooming both Image 1 and Image 8 by a factor of 1 and then repositioning them with Roam Image and <u>fs 5</u>. (With the present version of the program, the area to be clustered and the initial cluster centers cannot be properly defined if the image is zoomed and/or roamed.)

Choose an area you would like to work with and select a specific rectangle to cluster. You may want to choose an area to the south of the area we just used since the maps that are available cover that area. In general this area is much brighter than the previous area; this change is especially evident when you contrast the appearance of the same cover types. For example the deciduous forests (#21) are quite a bit lighter in tone.

To begin Cluster again, display the truecolor image of the area and type:

```
EXecute Code
```

You may respond to the prompts as you did before, using up to three channels of data for the input data and placing the Cluster results in Image 1. (This will destroy the Cluster results currently there.)

Estimate the number of clusters needed for this area by looking at the image along with the reference data, as you did before. Up to 16 clusters may be requested. Again, be prepared to specify a cluster center for each cluster, and set the number of iterations at 99. Watch for the number that tells you how many pixels are in the area being clustered and hit <u>fs 1</u> when the number of changed pixels falls below 1% of that total. (Note: Ignore the number of changed pixels displayed during the first iteration.)

If you vary the number of clusters, make the area larger or smaller, or use two channels of input data instead of three, you will gain some understanding of relative differences in processing times with different parameters.

<u>Step 8 - Store Cluster results</u>

Now that you have completed the Cluster operation, you need to store the results for future use or for further processing. This can be done through the COMTAL Utilities Program which is implemented on the PDP-11/34 at Purdue/LARS.

The processor for doing this is DATAFR, a subroutine that sends data from the COMTAL to the host computer. Log on the PDP and enter the following commands:

RUN COMTAL	(to call the COMTAL Utilities Program)
DATA	(to request the routine for transferring data)
DATAFR	(for subroutine that sends data from the COMTAL to the host)

You will be prompted to provide a file name for the new image and to state the COMTAL image plane number where the cluster results are located.

```
Enter your image file name:
```

The format expected is filename.filetype where up to nine characters may be specified for the filename and up to three for the filetype. The next prompt will be:

```
Enter image number 1, 2, 3:
```

Since the Cluster results are in Image 1, respond by typing 1. The sign VIS@VIS1 indicates that the transfer is taking place. You may verify that the file is saved by listing all the files on your portion of the disk; first hit <u>ESC</u> and then type END twice to exit the COMTAL Utilities Program. To see a list of the files, type the following on the PDP:

```
PIP DB1:/LI
```

The list that appears should contain the name you just entered. If you wish to delete the file, type:

```
PIP DB1:FILENAME.FILETYPE;*/DE
```

This ends the hands-on portion of this exercise. If the concept of clustering was new to you, you may benefit from reviewing some of the user documentation that you read earlier, prior to this hands-on work.

Step 9 - Review portions of the user documentation

Now that you have completed the hands-on activity, take a few minutes to review portions of the user documentation to reinforce your understanding of the clustering process and the significance of its output results. The following sections are suggested:

1. "General Description of the Clustering Process," approximately two pages in the User's Guide to Cluster.

2. "Low-level Information Extraction," approximately two pages in Introduction to the Georeferenced Information System.

When you have completed this reading, do the self-check questions that follow.

Self-Check Questions

The questions below are based on some of the objectives stated at the beginning of this hands-on exercise. Please test your understanding of the material by writing your answers in the space provided. You may then check your answers against the sample answers on the following page.

1. Describe the role of clustering in the classification of multispectral data.

2. Using Figure CLU-2, explain the following terms and state how each is obtained by the processor: initial cluster centers; migration of cluster centers; iteration; class means; L distance; convergence.

3. Describe briefly the steps you followed in running Cluster.

Answers to self-check questions

1. The clustering algorithm finds natural clusters or groupings within the data, based on their values in two or more channels. Statistical parameters of the resulting spectral classes, such as the means and covariances of the classes, are then calculated and passed to the classifier to characterize the training classes used in the classification.

2. Initial cluster centers are the multi-channel values of points which have been submitted by the user to start the clustering process. Migration of cluster centers is the movement, in feature space, of the cluster centers as the mean of each class is re-calculated to represent the new clusters of pixels. An iteration in Cluster is the process of assigning each point to the nearest cluster center and then re-calculating the mean of each cluster. The class mean is the mean value of all the points assigned to a single cluster class. L Distance is a statistical distance measure that is the sum of the distances between two points measured for each channel (or feature). Convergence is the condition that occurs when newly calculated cluster centers are identical with previously calculated ones, indicating that no pixels were assigned to different classes during that iteration.

3. Your answer should contain at least the following: sent data to the COMTAL image memory; enhanced the images; sent the Cluster code to the COMTAL; responded to prompts to provide information on ths features to be used in processing, the desired location of the results image, the area to be clustered, the number of clusters, the locations of initial cluster centers, and the number of iterations; and stored the results.

SYSTEM DOCUMENTATION FOR THE CLUSTER PROGRAM

This documentation consists of three parts:

1. A brief description of the flow of the Cluster program.
2. A list of all subroutines, their functions, and input/output arguments.
3. A flowchart depicting subroutine relationships.

The reader will find it helpful to become familiar with the algorithm implemented in Cluster by reading the user documentation for the program (see "User's Guide to Cluster on the COMTAL Vison One).

This program was written to execute independently of the COMTAL operating system because documentation of the operating system is not currently adequate to support integration of new functions by user programmers. As a result, execution of Cluster "locks out" the operating system until the Cluster code relinquishes control. It is anticipated that this situation will be altered in the future so that the user can employ the COMTAL facilities for displaying images while the Cluster processor is in control.

The cluster program requires one refresh memory channel per input image and two refresh memory channels for output, one of which is used to read back the output from the preceding iteration to determine if a change of cluster assignment has occurred. As a result, the maximum number of input images which can be accommodated by the current system is 14 (there is a total of 16 refresh memory channels available).

1. Program Flow

The main routine of this program (CLUST) contains data definitions that are used globally throughout the program, constant declarations, and the program driver. The driver consists of three basic parts:

1) Getting user information required, and clearing the result image;
2) Performing actual calculations;
3) Restoring the machine to the state it was in upon entry to Cluster.

1) Getting user information. This is an area where interfacing with the COMTAL operating system would be most useful. Most of the prompts to the user use the same pattern, i.e., get address of message text, write message on monitor, store answer, return. Error-checking is performed in these routines. If an error is encountered, a message is displayed and "wait-for-space-bar" is entered to ensure that the user notices the message. (Space-bar has been adopted as the terminator for all keyboard input.)

Inputs specified by user:

1) Number of input data channels
2) Input and output image plane assignments
3) Area to be clustered
4) Number of cluster centers
5) Locations of pixels defining initial cluster centers
6) Maximum number of iterations.

Return from the computation routine is to the point at which the number of clusters is specified by the user. If 0 cluster centers is specified, control is returned to the COMTAL operating system.

More detail on these routines can be found in the comments in the program listings.

2) The clustering calculations. All the code for the actual clustering calculations resides in one routine (COMPUT). The logic involved is as follows:

A) Calculate size of data structure to be used to store new cluster center values. Initialize iteration count to 0.

B) Set convergence indicator to "converged." Increment iteration counter. Clear data structure to be used for new cluster centers.

C) Display iteration count message. Initialize all image status registers to 0, for left - right, up - down data path.

D) From the refresh memory, load the data for all input channels for the next pixel into temporary storage array.

E) Determine cluster center nearest to the pixel by summing absolute values of differences between cluster center and pixel data in each channel (L_1 distance); if this is less than the distances previously calculated for other cluster centers, update the cluster assignment.

F) Store final cluster assignment in result image.

G) Add the pixel values for each channel component-wise to the nearest-cluster total for this iteration. This will be used at the end of the iteration to calculate the updated cluster centers. Also check to see if the new cluster number is the same as the previous assignment; this checks convergence (convergence = no change for an entire iteration). If a change has occurred, reset convergence indicator to "not converged."

H) Check to see if the ESCAPE key was hit; if it was, then return (go to N).

I) Check to see if all pixels in the specified area have been processed. If no, then go to D; else J.

J) Calculate new cluster centers (cluster means) -- double precision arithmetic; use right shifts to discard unneeded precision.

K) If clustering has converged, jump to N; else go to L.

L) If maximum number of iterations has been reached, go to N.

M) Check to see if Function Key 1 was hit; if so, go to N. Otherwise go to C.

N) Restore registers and return to request new user inputs.

3) Restore entry status. Restoring the entry status consists only of clearing the image memory definitions and restoring the registers.

2. Subroutines

All routines used in the Cluster program are listed below, with a brief description. On the following page the function and input/output arguments of each are described.

CLUST	- Driver for the program
CLRBUF	- Clears the display buffer
CLRPLN	- Clears images, graphics selected
COMPUT	- Does the actual clustering
DISMES	- Displays a message on screen
DRWBND	- Draws the bounds of a rectangle
GRASUB	- Subtracts the graphic from the screen
GTBNDS	- Gets the cluster area boundaries
GTCLUC	- Gets the cluster center values
GTIMAB	- Determines the limits in the refresh memory of area to be clustered
GTIMAS	- Gets the image plane assignments for I/O
GTNCHS	- Gets the number of channels to cluster
GTNCLU	- Gets the number of cluster centers
GTNITS	- Gets the maximum iterations desired
RDNUMD	- Prints a message and reads the integer number
SETVAR	- Sets up the variables required
STIMAB	- Sets up the channel registers
TARGET	- Turns the target on and off
WTKYBD	- Waits for a keyboard character
WTPBUS	- Waits until buffer or P-Bus is not busy

Note: The term "all registers" refers here to registers 0 (R0) through 5 (R5); register 6 is referred to as SP.

CLUST - This routine is the driver for the program. Global data structures and constant definitions are made here. All registers are saved on entry and restored before exit.

CLRBUF - This routine clears out the print display buffer used to display a message on the monitor. One input is passed on top of the stack. This argument tells how many blank words to put in the display. A negative number indicates a start at the beginning of the display; a positive number means to continue from where display buffer pointer is. All registers are saved on entry and restored before return.

CLRPLN - This routine is used to clear out an image or graphic plane. One input argument is passed in R5. This argument contains a code passed to register 17 of the (REFC) card telling it where to direct output. For an image, pass 4377 ored with (the image number * 400) Note: image number is started at 0. Register 5 is used as working storage and is saved on entry and restored before exit. R5 is not changed in this routine, and so all registers have same value on exit as they had on entry.

COMPUT - This routine computes the cluster results using the L1 distance measure. All data is passed using global variables that define where the cluster input image data resides. The routine saves all registers on entry and restores them before exiting.

DISMES - This routine is used to display a one-time message on the color monitor. The top three lines, if displayed, are blank. Three arguments are passed to the stack and should be pushed in the following order:

1) the starting address of the message,
2) the number of bytes in the message,
3) if <u>non-zero</u>, wait for matching keyboard strike.

All registers are the same on exit as on entry.

DRWBND - This routine is used to draw a rectangle around the area defined by the pairs (IMAULX, IMAULY) and (IMALRX, IMALRY). Rectangle is drawn in bit 0 of the result image plane. No arguments are passed in or out. All registers are the same on exit as they were on entry.

GRASUB - This routine subtracts the graphic that was added to the screen earlier. Registers 1 and 3 are destroyed. No input or output arguments are used here.

GTBNDS - This routine gets the boundary area (screen coordinates) of the data set to be clustered. Function keys and the trackball are needed by the user. No input or output arguments are given. All registers are destroyed.

GTCLUC - This routine gets the initial cluster centers as indicated by the user with the trackball. No input or output arguments are supplied. All registers are destroyed.

GTIMAB - This routine takes the cluster area screen boundary and calculates the data bounds in the image data base. No input or output arguments are supplied. Register values are destroyed.

GTIMAS - This routine gets the numbers of the images to be used for inputs and results. Data base registers are set up here. No input or output arguments are supplied. All registers are destroyed.

GTNCHS - This routine gets the total number of images to be used for input. No input or output arguments are supplied. No registers are used, so all registers are returned the same as on entry.

GTNCLU - This routine gets the number of cluster centers the user desires. No input or output arguments are supplied. No register values are changed.

GTNITS - This routine gets the maximum number of iterations the user desires. It is "maximum" because the process may converge before reaching this number. No input or output arguments are specified. No register values are changed.

RDNUMD - This routine prints a message and waits to read in an integer number followed by a space. All registers are saved on entry and restored before exiting. Input arguments, in the order they are pushed on SP (Register 6), are:

1) Maximum number of digits to be read in,
2) Address of the question to be asked,
3) Number of bytes (characters) to write on the monitor.

Output arguments in the same stack position as above:

1) Number of digits actually read,
2) The number read in as the answer.

Valid input characters are 0 through 9 and "-" (minus). If an error occurs, an error message is displayed on the monitor and wait-for-spacebar is entered.

SETVAR - This routine acts as a driver for the initialization routines. No input or output arguments are supplied. Registers 3 and 5 are destroyed; others are not used locally but are destroyed in called routines.

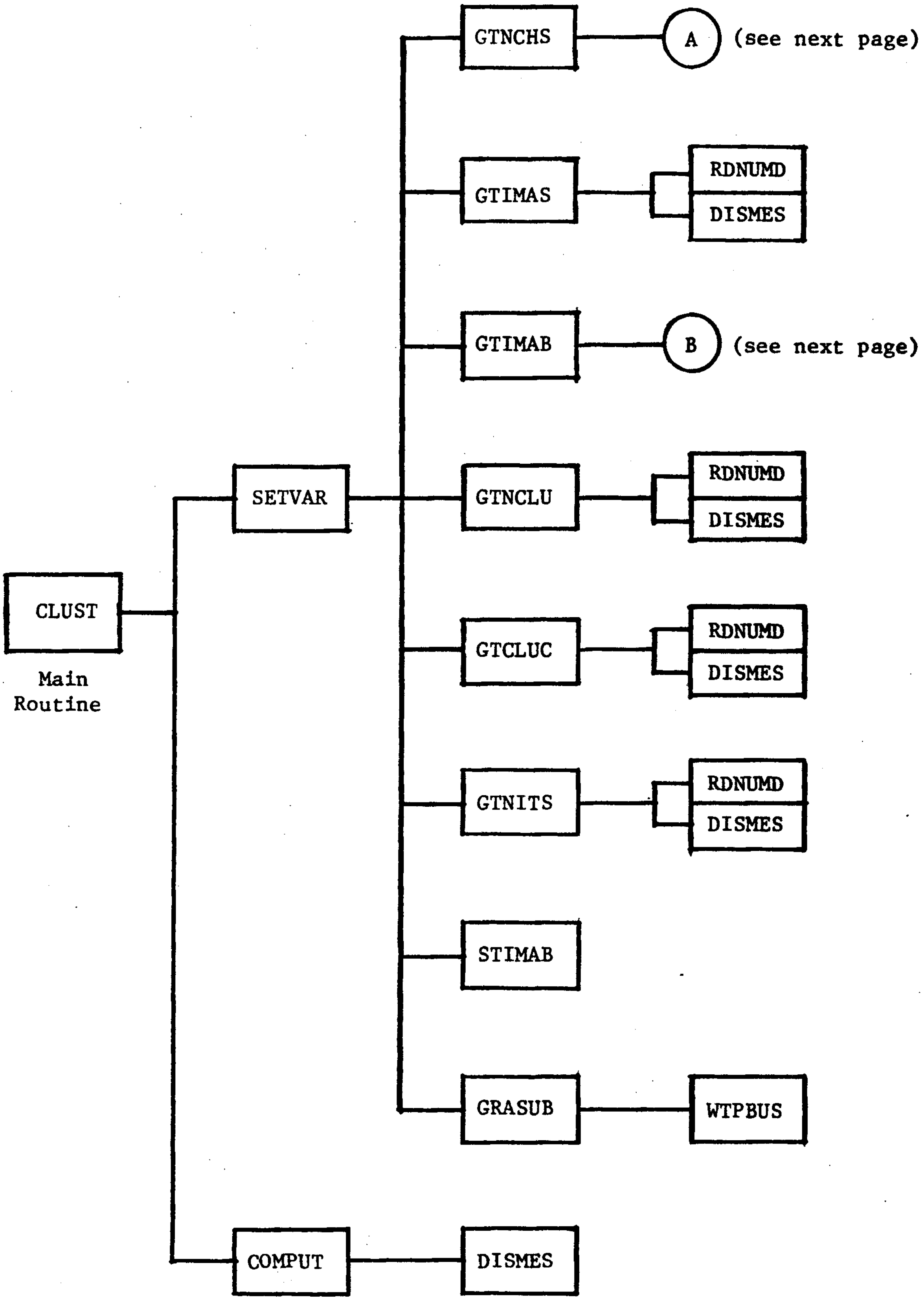
STIMAB - This routine sets up all the channel registers that are needed to define the input channels from the image data base area previously defined. No input or output arguments are supplied. Registers 0, 1, and 2 are destroyed.

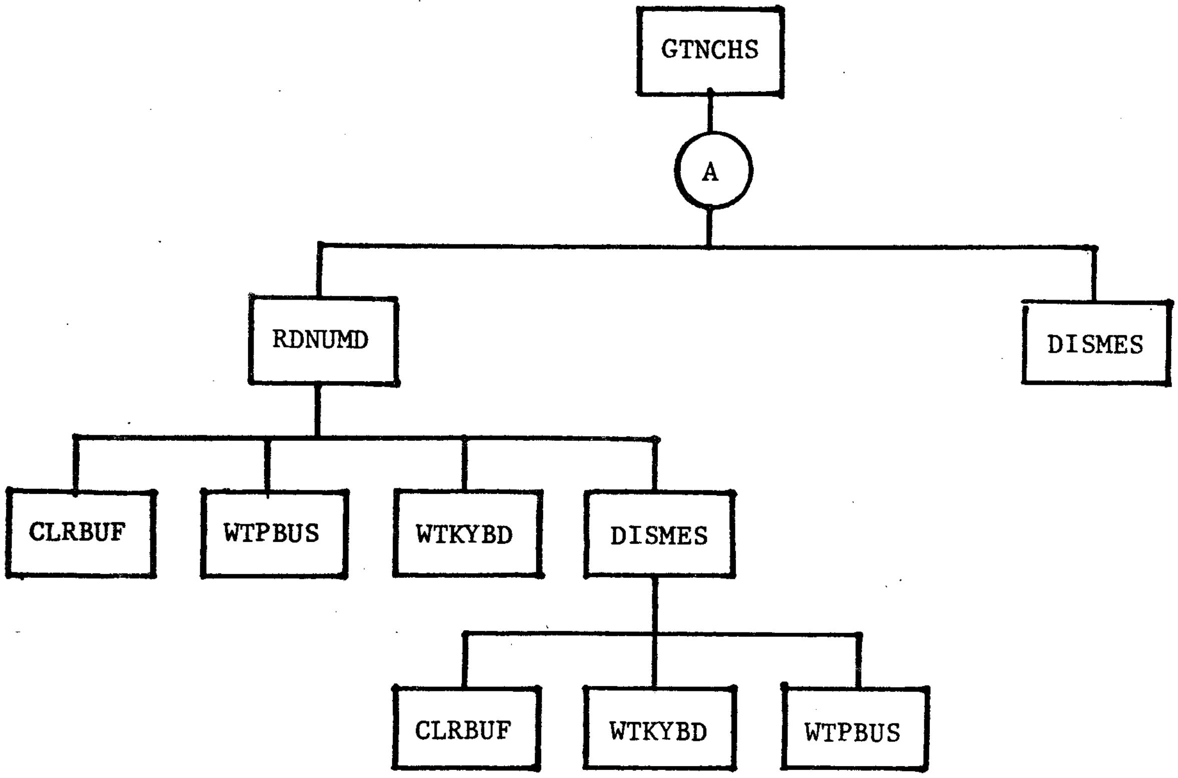
TARGET - This routine adds and subtracts the target from the display. All registers contain the same values on exit as they did on entry. R0 contains the only argument (input): a 0 turns off the target; anything else turns the target on.

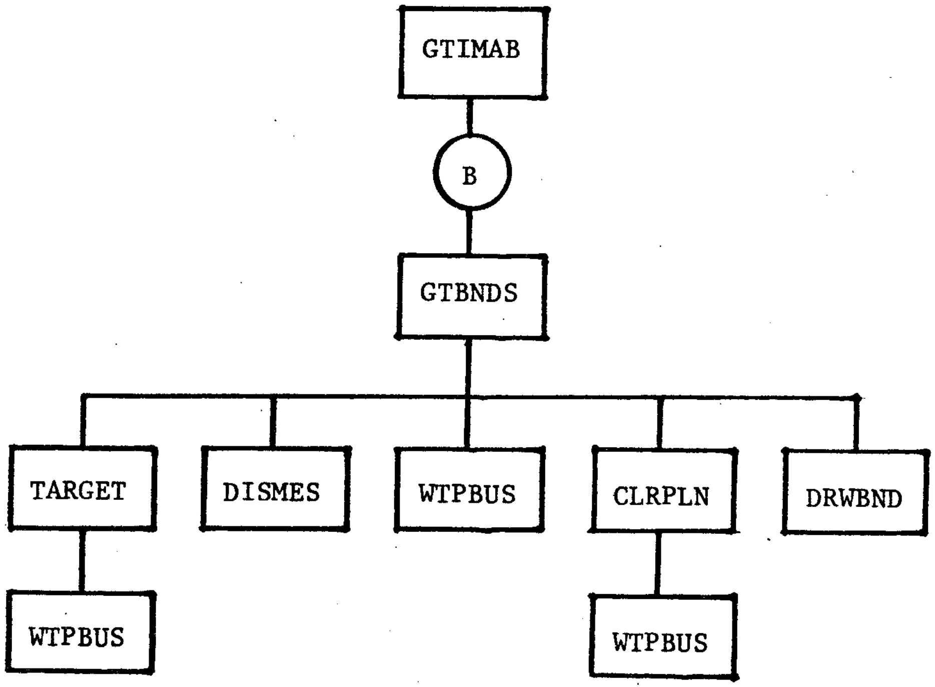
WTKYBD - This routine is used to wait for a keyboard hit (any character). No arguments are specified. No registers are used or destroyed.

WTPBUS - This routine is used to wait for a signal that the P-BUS transfer previously requested has been completed. No arguments are specified. No registers are used or destroyed.

3. Subroutine Flowchart




GTNCHS
A
RDNUMD
DISMES
CLRBUF
WTPBUS
WTKYBD
DISMES
CLRBUF
WTKYBD
WTPBUS


GTIMAB
B
GTBNDS
TARGET
DISMES
WTPBUS
CLRPLN
DRWBND
WTPBUS
WTPBUS

TEST PROCEDURES FOR CLUSTER

A. Initialization

1. Reset the Vision One/20: [SHIFT RS] R

2. Load test patterns as Images 2, 3 and 4
   (See Table 1.)
   ASsign Truecolor 8 red 1 green 2 blue 3
   INItialize Function memory 8
   Clear Image 1
   Display Image 8; Add Function memory 8 (ck N)

3. Define function memory for output image
   Set INteger function 1
   INteger function 1 = INteger (X * Constant 135 / Constant 8 + Constant 120 #)
   Function memory 1 = INteger (integer function 1 #) #

4. Load Cluster program code into the Vision One/20
   Note: Do not attempt to load the program code until the test pattern generation is complete.
   a. On the Purdue/LARS system:
      Log into the PDP-11/34
      RUN COMTAL
      Type CLUSTR when the main menu of the COMTAL Utilities Program is displayed.
   b. On other systems:
      Set CODe size 16
      Load Cluster code using local procedures.

Table 1. Test Patterns for Cluster

Three test patterns are used for testing the soundness of the Cluster software. It is convenient to generate these patterns using the Vision One and to store them elsewhere for downloading from the host computer; alternatively, they may be regenerated on the machine each time they are needed, although the time involved is not inconsequential.

The following commands generate the required patterns.

```
Set INteger function 2
INteger function 2 = INteger (X / Constant 32 * Constant 16 #)
Image 2 = INteger (integer function 2 #) #
Set INteger function 3
INteger function 3 = INteger (Y / Constant 32 * Constant 16 #)
Image 3 = INteger (integer function 3 #) #
Set INteger function 4
INteger function 4 = INteger (X + Y / Constant 32 * Constant 8 #)
Image 4 = INteger (integer function 4 #) #
```

NOTE:

1. Image 2 is vertical bars, Image 3 is horizontal bars, Image 4 is diagonal bars.
2. # denotes "type an additional space following the space which normally follows a typed character string"
3. A set of test patterns is stored on the distribution tape supplied to COMTAL Corp. by Purdue/LARS. CLUSTERX.DAT is vertical bars, CLUSTERY.DAT is horizontal bars, CLUSTERXY.DAT is diagonal bars.

B. <u>Run Cluster on Test Patterns</u>

1. EXecute Code

No. of channels of input data:	2
Results image plane:	1
Features 1,2:	2,3
Cluster area:	select any 8 squares (2x4) in test pattern
No. of cluster centers:	2
Cluster center locations:	extreme upper left and lower right corners of area specified.
Iterations (max):	2

RESULT: When program halts, asking "How many cluster centers?" hit space bar to exit. Display Image 1 with Function Memory (<u>ck G</u>). You should see two distinguishable squares.

2. Display Image 8; Add Function Memory 8 (<u>ck N</u>)
EXecute Code

No. of channels of input data:	2
Results image plane:	1
Features 1,2:	2,3
Cluster area:	Use the same 8 squares as above (it is only necessary to hit the space bar).
No. of cluster centers:	8
Location of cluster centers:	one in each of the 8 visible squares.
Iterations (max):	99

RESULT: The Cluster processor begins running at this point. Hit <u>ESC</u> immediately. The program should halt immediately, returning to ask for the number of cluster centers.

No. of cluster centers:	8
Location of cluster centers:	as above, one in each of the 8 visible squares.
Iterations (max):	99

RESULT: This time, hit function switch 1 (<u>fs 1</u>) immediately. The Cluster program should halt after completing the first iteration, WITHOUT proceeding to the second iteration.

No. of cluster centers:	8
Location of cluster centers:	one in each of the 8 visible squares.
Iterations (max):	99

RESULT: Program should halt at end of second iteration. Hit space bar to exit. Display Image 1 with Function Memory 1 (<u>ck G</u>). Should see 8 distinguishable squares.

3. Image 1 = Image 4
   INItialize Function memory 1
   Display Image 8; Add Function memory 8 (ck N)
   INteger function 1 = INteger (X * Constant 135 / Constant 16 + Constant 120 #)
   Function Memory 4 = INteger (integer function 1 #) #
   EXecute Code

No. of channels of input data:	3
Results image plane:	4
Features 1,2,3:	1,2,3
Cluster area:	8 squares (16 triangles)
No. of cluster centers:	16
Location of cluster centers:	One in each of the 16 visible triangles.
Iterations (max):	99

   RESULTS: Program should halt at completion of iteration 2. Hit space bar to exit. Display Image 4; Add Function memory 4. Should see 16 triangles displayed as different gray levels. Use DUmp Image to verify 16 distinct gray levels number 1 through 16 (background is level 0).

4. System reset: type Shift RS R
   Reload code as in step A.4 above.
   ASsign Trucolor 8 red 1 green 2 blue 3
   Display Image 8; Add Function memory 8 (ck N)
   Set INteger function 1
   INteger function 1 = INteger (X * Constant 135 / Constant 4 + Constant 120 #)
   Function memory 1 = INteger (integer function 1 #) #

   EXecute Code

No. of channels of input data:	2
Results image plane:	1
Features 1,2:	2,3
Cluster area:	Hit space bar to accept entire image area.
No. of cluster centers:	4
Location of cluster centers:	One in each extreme corner of the image area.
Iterations (max):	99

RESULT: Hit function switch 1 (fs 1) immediately, allow Cluster to finish full image area. When program halts, hit space bar to exit. Display image 1; Add Function memory 1 (ck G). The four quadrants of the screen should appear as distinct gray levels.

C. Run Cluster on Landsat Data

Unassign Image 8
ASsign Truecolor 8 red 4 green 3 blue 2
Display Image 8; Add Function memory 8 (ck N)
Load the Vallecito test data:

```
VALLECITO.CH1 into Image 2
VALLECITO.CH2 into Image 3
VALLECITO.CH4 into Image 4
```

On the LARS system, this may be done using the DATA function from the main menu of the COMTAL Utilities Program. On other systems, use local procedures.

```
Function memory 8 = Histogram of image
Equalize Function memory 8
INteger function 1 = INteger (X * Constant 135 / Constant 8 + Constant
    120 #)
Function memory 1 = INteger (integer function 1 #) #
EXecute Code
```

No. of channels of input data:	3
Results image plane:	1
Features 1,2,3:	2,3,4
Cluster area:	Pick any interesting area of about 100 pixels x 100 pixels.
No. of cluster centers:	8
Location of cluster centers:	Attempt to select 8 pixels of different color.
Iterations (max):	99

RESULT: The Cluster program will probably run to convergence in between 10 to 25 iterations. After the second iteration, the "change" indicator should show a decreasing trend but may occasionally increase for one or more iterations. If desired, the program may be stopped (hit ESC or fs 1) after the change counter reduces to less than 10 percent of the count during iteration 2. Then hit the space bar to exit the program, and view the clustering results (ck G). Flipping between the data (ck N) and the results (ck G) you should see a pronounced correspondence between spatial features in each.