INTERACTIVE CHOROPLETH MAPPING
USING HIGH RESOLUTION COLOR GRAPHICS

by

Julie B. Spielman

A RESEARCH PAPER

submitted to

THE DEPARTMENT OF GEOGRAPHY

in partial fulfillment of the
requirements for the
degree of

MASTER OF SCIENCE

July 1986

Directed by
Dr. A. Jon Kimerling

Table of Contents

Illustrations

Figure

Tables

Interactive Choropleth Mapping

Using High Resolution Color Graphics

ABSTRACT: A program, Color Choropleth Mapping System (CCMS), has been developed to provide high quality color choropleth maps at the Geography Department at Oregon State University. The program operates interactively and is menu-driven. When developing the software for mapping programs, programmers need to be aware of cartographic theory, as well as the relationship between maps, computers, and users. If color is utilized, the programmer should be familiar with color theory, perception and specification. By understanding more than the basics of computer science and programming, programmers will be able to develop software to produce high quality maps that will convey the information intended, and provide decision makers with a valuable and powerful problem-solving and planning tool.

INTRODUCTION

Mark Monmonier states in his book Computer-Assisted Cartography: Principles and Prospects (1982) that there are three general approaches to making computer-assisted cartography operational. First is the software package or "canned program" purchased by an individual or group, such as a university, to do one or more mapping procedures. The software package usually exists in a computer program library where a wide number of users can access it. Second is the turnkey system that includes both software and hardware and comes ready to run. It is designed to be used exclusively for a specific mapping task by a private firm or governmental agency. An example of this is the Decision Information Display System (DIDS) designed to provide federal administrators and legislators with high quality color choropleth maps. The third approach is the small multipurpose system consisting of a small mini-computer, CRT unit, pen plotter, and digitizer. The system is set up to be used by a variety of users for a wide range of tasks.

The mapping program is the backbone of all three systems and will be the focus of this paper. Mapping programs can be separated into two broad categories - single purpose and multipurpose programs (Lai, 1985). The single purpose program contains one mapping procedure such as choropleth mapping, contouring, or map projections. Examples of the former include DIDS, STATMAP, and MULTIMAP I. The multipurpose software package contains several mapping applications. DISSPLA, a multipurpose software package, contains several cartographic routines that produce contour maps and map projections along with a variety of graphs and charts (Carter, 1984).

Both categories of mapping programs are most often written for a certain class of display device such as a cathode-ray tube (CRT) unit or line printer (Monmonier, 1982). SYMAP is a well-known software package developed at Harvard Laboratory of Computer Graphics which produces choropleth, isopleth and trend surface maps on a line printer (Carter, 1984).

Mapping software packages can be designed, programmed and debugged by and for an individual user, a group of users, or an institution, and the process of developing one requires careful consideration. Software packages can be written to stimulate trial-and-error thought or they can be designed to facilitate the decision making process. A wide variety of cartographic techniques and methods can be incorporated in mapping software giving "map-makers access to the most current cartographic wisdom" (Monmonier, 1982).

SOFTWARE DEVELOPMENT

There are five primary components of a cartographic system: digitizing, storage, interactive display, edit, and drafting (Boyle, 1975). Mapping software would be of little use without mappable data, therefore the digitizing and storage of spatial data are of prime importance. It is the interactive display and edit facilities, however, that are the main concern when developing mapping software. These two components are the main interface between a digital system and the cartographer, and enable the cartographer to use his skills to guide the mapping process.

When developing a computer program, it is important to make the program as "user friendly" as possible. It may involve more programming, but ease in the use of a package reduces user frustration, and promotes the correct outcome of the programming task (in this case mapping). Computer programs can be written using methods which allow the computer and user to interact. In such a system, the computer rapidly responds to the program user's instructions, and the user can then react to the computer's response. The interactive display portion of a mapping program lets the user judge the maps appearance step by step, and change or alter (edit) portions of the map as needed.

When developing a mapping program it is not sufficient to understand just the basics of computer science and programming. Cartographic theory, as well as the relationship between maps and computers, and humans and computers, need to be considered. Computer generated maps are viewed differently than traditional paper maps. Map design and the process of designing maps has changed with computer use. For instance, the classification in choropleth mapping is not longer a necessity, but an option available to the user due to the classless choropleth map presented by Waldo Tobler (1973). Tobler proposed that cross-line patterns with proportional graytones were more appropriate for choropleth maps than a small number of graytones assigned to discrete categories (Monmonier, 1982).

The use of color on maps has been affected. Research on colors displayed on CRTs, conducted by Steike and Little (1984), concluded that the attribute of brightness is more important than hue when map-readers made judgments of the visual importance of colors. Computer technology is changing rapidly and it is important that programmers developing mapping software be aware of these changes and their effects on maps and cartographic techniques.

Map Design

The development of mapping software can facilitate the map design process. Designing maps is a decision making process and "decisions are more likely to be good if they are rational rather than intuitive" (Robinson, 1975).

Menus

Menus are one technique that a software developer can incorporate into a mapping program to promote good map design. Menu-driven programs can provide the user with rational choices while facilitating the usage of the program. A program user needs certain skills to give commands to a computer in an orderly manner. Menus reduce user frustration by making it a simple task to give computer instructions (Lai, 1985). Menus also limit the choices of a user, preventing the inexperienced user from making wrong choices for a certain task. Menus can be arranged in a hierarchical order and display cartographic tasks in sequential order, both of which will assist the user in constructing a map.

Perception

During the map design process, the cartographer is faced with the problem of creating the best solution (design) to convey the message of the map. One of the keys to the utility of thematic maps is that spatial phenomena are organized within a spatial framework, and organized material is perceived, understood, and retained better than unorganized material. The cartographer enhances basic image organization by selecting certain graphic techniques and design dimensions that further organize the image (Dobson, 1984).

Careful attention must be given to the communication aspects of a map. The perception of cartographic features must be considered in the design process. Michael Dobson states that the "key to understanding design requirements of cartographic displays lies in appreciating the visual processing needs of the map reader" (Dobson, 1983). Certain variables such as size, shape, color, pattern, direction, and location affect how the map is perceived. To promote communication, cartographic programmers need to be aware of the perceptual consequences these variables have when placed on a map. The perception of computer generated cartographic variables is especially important because there is usually no human review process of map displays before they are presented to the map reader.

New research needs to be undertaken on the interface between maps, map task performance, and the capabilities of digital display systems (Dobson, 1984). The cartographer-programmer needs to consider the nature of the technology (e.g. line printer, CRT, pen-plotter) under which a map is produced. In the case of the CRT, Human Factor specialists have been studing the perception of real-time computer displays, but so far have offered few insights relating to the complexities and presentation requirements of maps (Swezey and Davis, 1983).

Cartographic literature on theoretical mapping covers the perception of maps, but only in traditional format. A gap exists in display design guidelines between the cartographic requirements for representing spatial data, and the human factors related to efficient map use. The solution is obviously an interdisciplinary approach in which cartographers "determine the appropriate form for mapped elements" and human factor specialists "determine the constraints that may influence the visual processing" of cartogrpahic data (Dobson, 1983). As guidelines are created for computer-generated maps, it will be important that mapping software provides a reliable link between the map displayed and the display technology.

Interactive map design

As stated, mapping software can be developed to take advantage of the user's ability to interact with a computer. Interactive mapping takes advantage of the computer's ability to rapidly perform sequential processes, and gives the cartographer control over the processes not easily performed in a sequential manner (Dudycha, 1981). For instance, plotting points and drawing lines are easily performed by a computer because of the mathematical nature of the task and the repetition inherent to the task. However, the placement of text on a map can vary between maps or within the same map. This procedure can be assisted by the cartographer who can instruct the computer as to where to place the text (Rase, 1975). In recent years, research has been undertaken to automate the labeling of maps. One system still under development at ESL (Environmental Systems Laboratory) is ACES, which currently can solve moderately complex map labeling tasks involving point, line, and areal features (Pfefferkom et. al., 1985).

Interactive map design allows the user to find the best alternative for a specific map. The user can display a map on a CRT in several different formats and the data can be mapped using different methods to find the best solution. Different colors and symbols can be previewed by the map designer (Monmonier, 1985). An interactive mapping program allows the user to participate in an interactive design loop, inserting human creativity into the decison-making process.

Interactive display maps can be more valuable and powerful as a problem-solving and planning tool because they are able to show updated and changing information in a dynamic form (Anderson and Shapiro, 1980). Interactive display maps can show the results of complex computations rapidly and efficiently. In an interactive system the user can selectively choose what needs to be displayed. Often, these maps are created for private use to be used for analytical purposes. Maps produced on paper are usually designed for a large audience. The cartographer designing these maps must deal with displaying all the information necessary for public view, and therefore must balance clutter with the omission of needed information (Anderson and Shapiro, 1980).

A wide variety of needs and constraints can be incorporated in mapping software, providing map makers with the most current cartographic techniques and methodologies.

Currently, many professionals (businessmen, engineers, social scientists, and planners), who once only had access to printed maps, are generating new maps tailored to their specific needs. Such people rarely have the cartographic expertise to create maps, and therefore it becomes imperative that mapping programs be able to guide the user in designing and producing maps that will convey the information intended. Such programs might be written to mimic the ideal response to various design problems. 'Expert' systems based on Artificial Intelligence (AI) techniques are being developed in the field of geography to accomplish tasks such as this. Expert systems offer 'intelligent advice' or make an 'intelligent decision' about a processing function (Robinson and Jackson, 1985). Proper map design is necessary in digital cartography and must extend to the mapping software (Monmonier, 1982). The design of computer-generated maps could benefit considerably from the development of such systems.

The Use of Color

Color on a map can enhance and clarify the portrayal of information. Color allows greater detail, adds visual interest and increases the design possibilities (Heyn, 1985). There are three basic purposes color can serve on a map. First, it can represent the distribution of mapped variables. Second, color can be used to disassociate the foreground (thematic data) from the background (geographical space). And third, color can be used to annotate (the non-target) part of the display (e.g. grids and lettering). In these three purposes, color serves as an organizer and can affect the hierarchy of the image elements. In order to select colors effectively for a computer mapping system, it is necessary for the program developer to be familiar with color theory, perception and specification. It is also important to understand color production technology so that the desired colors can be produced.

Color Theory

White light is produced when a light source emits the full range of visible wavelengths. When white light passes through a prism, it is broken into the wavelengths forming the color spectrum. The various wavelengths create the different hues (e.g. blue, red, yellow) of the spectrum. Six of the hues are referred to as the primary colors because other hues can be created from their combinations.

The six primaries are broken into two sets. The first, consisting of red, green, and blue, are referred to as the additive primaries, so named because all other colors can be created on a "white" surface by adding or mixing light of these three colors in varying intensities (Robinson et. al., 1984). Colors on paper are created by applying pigments to a surface. The second set of primary colors, subtractive primaries, consist of yellow, cyan, and magenta. When these colors are applied to a surface, while light illuminating the surface is absorbed or subtracted by the colors, and what is reflected is the color seen. Most colors can be created by a mixture of these colors. Cartographers use the method of mixing subtractive primaries in map production (Robinson et. al., 1984).

There are three perceptual dimensions of color. The first is hue which, as discussed, is the dimension relating to the wavelength. The second characteristic is value, which describes the lightness or darkness of a pure color or hue. On a printed map, value indicates the amount of reflectance perceived to be given off by a surface. The term parallel to value used to describe the luminous intensity of a CRT surface is brightness. Saturation, the third characteristic, is a measure of the purity of a hue. Chroma is the perceptual term used to describe the purity of a color (Heyn, 1984).

Color Perception

As noted, color is a perceptual as well as a physical phenomenon. Different color systems have been developed that take into consideration how a color is perceived. The Munsell System is an example of a system in which colors are described using the three perceptual attributes - hue, value and chroma. Each characteristic is divided into a sequence of steps which are perceived to be equal (Robinson et. al., 1984).

Color production on a CRT

There are several ways color can be identified on a CRT. The simplest form is by a name or number. On the Apple II, color is specified by the assignment of a number. The numbers range from 0 to 15; 0 representing black, 7 representing light blue, and 15 representing white for example (Poole, 1981).

Some systems allow the user to produce the color. The RGB (Red, Blue, Green) color model is a common system used. The physical representation of the RGB model is usually a cube (Figure 1). The intesity for each color varies along one axis of the cube. The faces of the cube represent a combination of two of the three colors (Sibert, 1980).

In the RGB system the three primary colors are transmitted on three different channels. Through mixing the colors, a wide gamut of colors can be created. The intensity of the hue is calibrated electronically by assigning values, usually ranging from 0 to 255, 0 being no color, and 255 being the greatest intensity of the hue. The primary colors are transmitted by the individual color channel, and by specifying more than one channel, the primary colors are mixed and produce new colors (Figure 2), (Traeger, 1982). Each RGB system is dependent upon the color display device used for output (Robertson, 1986).

Alternative models exist to the RGB model. Most of these are variations on the HSL (hue, saturation, lightness) model. In an HSL system, each color is considered a pure hue modified by a saturation and a value (i.e. lightness), (Berk et. al., 1982). Algorithms have been published which transform the HSL and other models to RGB (Sibert, 1980).

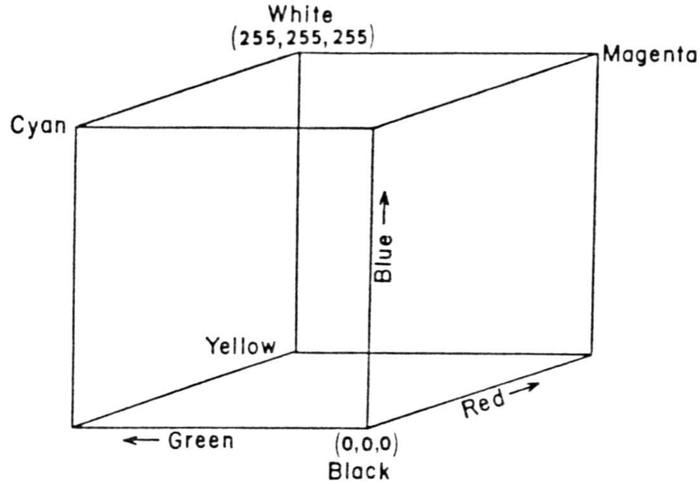

Figure 1. RGB color cube

HUE INTENSITY SCALE

255	255	255
255	233	233
243	178	178
230	111	111
217	0	0
168	0	0

COLOR MIXING SEQUENCE

RED
GREEN
BLUE

COLOR MIXING TABLE

Red	Cyan	Yellow	Green	Blue	Magenta
255	0	255	0	0	255
0	255	255	255	0	0
0	255	0	0	255	255

Figure 2. Notation for RGB system

Phenomena and Color

System developers should be aware of the effects phenomena and associated data characteristics have in selecting colors. Four such characteristics are the scaling system, spatial dimension of the variables, data collection units, and the color connotations of the data (White, 1979).

An important distinction to make in using color is whether qualitative or quantitative differences exist between mapped variables (Cuff, 1975). In mapping nominally scaled phenomena (no ranking intended) the colors chosen should carry little or no magnitude implications. Quantitative changes are usually indicated by value and/or chroma differentiation. It is important for the colors or chromas to be distinctive for each class and be arranged in an obvious progression. The colors should be sufficiently distinct so that the reader can easily identify an area's class by referring to the legend.

The physical size of mapping units affects color choice. When displaying relatively large areas, the color contrast between neighboring polygons can change the color perception (White, 1979). Certain colors also generate an emotional response, called the affective value of color, which must be taken into consideration. "Warm" reds and "cool" blues carry with them inherent connotations. For certain phenomena these two variables are difficult to incorporate into a mapping program, but an interactive system would permit the user to take advantage of this.

Mapping with color

Conceivably, hundreds of thousands of different colors could be produced using an RGB monitor. But, to facilitate the program-user it is important for the mapping software to limit the colors available, and to present them in some orderly and functional fashion.

Little research has been undertaken on the perception of colors on CRTs, but algorithms have been written to create color palettes with intensity and value varying along the x and y axes. Once a color palette is established, it can be located at either the top or bottom of the image frame, and using a pointing technique, the color can be assigned to areas and symbols within the body of the map (Traeger, 1982). The color palette, if created with adequate differences within it, can help the user to determine color progressions. Some mapping programs allow the user to choose a hue, and will generate the chroma progression automatically (Carter, 1984).

COMPUTER TECHNOLOGY

The CRT

Display technology has prompted an increase in the sophistication in which computerized information can be accessed (Infante, 1975). The development of the CRT in the 1970s allowed maps to be displayed and edited at a terminal (MacDonald and Crain, 1985). When output is displayed on a CRT, additional cartographic and visual process factors must be considered

(Dobson 1983). The methods of handling color, the amount of information displayed, and whether a system can be interactive in nature are all factors in software development due to variations among CRTs.

Resolution

Important characteristics of a CRT include resolution, color control, and internal memory structure (White, 1979). Resolution is probably the most important factor when choosing a display device for a certain application (Infante, 1975). The resolution affects the complexity of the display. It currently varies from high resolution (1024 X 1024) to low values (40 X 50) where the pixel size may be a large fraction of an inch. Low resolution display devices restrict the annotation capabilities, therefore, maps must be relatively simple and small in area (MacDonald and Crain, 1985). Because of the variety between different types of CRTs, mapping systems must be developed for the specific display device.

Colors

The colors that can be displayed on a CRT are referred to as the displayable colors. The number of colors available can range from 4 to 16 million or more. The displayable colors may be fixed or variable. Presently the CRT is the only technology available with a full gamut of colors (Infante, 1975). Resolution of a screen affects the color perceived. The effect of mosaic fusion exists when small regular grid cells of color are merged by the eye into an additive color mixture. The fusion effect may significantly alter the perception of data variation (White, 1979) and therefore should be compensated for in the software.

Refresh and storage tubes

The resolution of the system, and the number of colors and shades that can be created are related to the memory dedicated to refreshing the image (Carter, 1979). It is the refresh nature of CRTs that permit images to be changed and modified in real time. An image will disappear from a screen unless it is "refreshed" by a pass of an electron beam (at least 30 times a second). On storage tubes, the display image is stored on the screen for several minutes or longer. To change or modify an image on a storage tube CRT, the image must be erased and completely recreated by the program. This is not well suited to highly interactive applications (Dudycha, 1981).

Computer Processing

The speed of computer processing is important for the interaction between the user and mapping system to take place. Before 1970, software development did not require great speed in operations. Reliability was the most important criterion. Before the 1970s though, the CRT was not in common use. Interactive displays, and editing of cartographic data showed that the ease and flexibility of handling cartographic data was not enough. Speed became essential for true interaction to take place (Boyle, 1975).

CHALLENGES AND CONCERNS

The "electronic" map has changed many aspects of cartography. A little over a decade ago, maps were only conveniently available on paper. Today geographic data are stored, edited, distributed, and displayed electronically with the use of computers (Monmonier, 1985).

Computer cartography has forced cartographers to re-evaluate some of the basic cartographic techniques. Fundamental cartographic problems such as scale and generalization need to be re-examined, since scale changes greatly affect map quality (Taylor, 1973). Before the advent of computer-assisted cartography, the effective use of a map projection was a central problem in cartography. With the aid of computers, it has become a simple task to select from a wide range of projections or to create a combination of projection, scale, and projection orientation best suited for a particular cartographic problem (Dudycha, 1981).

Line printer, plotter, and CRT maps present new challenges. As stated, perception of CRT-displayed maps is a largely unexamined field. The visual processing of real-time cartographic displays is extremely complex and research on graphic techniques that will increase task performance (e.g. rapid information retrieval, target identification, and temporal changes across displays) for cartographic displays needs to be developed (Dobson, 1984).

Human Factor guidelines governing the man-computer interface are greatly needed. To develop an effective system it is important to understand the cognitive areas of this interface. Such areas include input devices (joystick, graphics tablet), interactive dialog (menu selection or query language), and design decisions relating to dynamic displays (zoom, scrolling) (Swezey and Davis, 1983). The lack of knowledge in these areas adversely affects the development of computer-assisted cartographic systems.

Little analysis has been done on the types of display designs that could promote useful and timely decision making. In 1978, the Decision Information Display System (DIDS) was implemented to provide decision makers with statistical data in a form that could be easily understood. The system, however, had some major shortcomings that should be noted by others trying to accomplish a similar task.

One of the drawbacks to DIDS was that the color raster display technology did not support an inexpensive or easy method to produce hardcopy products (Cowen, 1984). Presently, hardcopies of images on color CRTs can only be generated by photographing the screen to obtain slides or glossy prints, both of which are expensive and useless for providing reports with maps. COM units (computer output on microfilm units) or plotter and printer-plotters with the proper software can record images from a CRT. COM plots can be enlarged providing press plates for offset printing, but COM units are expensive ($120,000-$300,000) and only practical for reproducing large quantities. Ink jet plotters can provide rapid hardcopy color maps, but the problem of reproducing color accurately from the CRT screen still exists (Monmonier, 1982 and Orr, 1980). Research needs to be undertaken in the area of reproducing color faithfully and effectively from a CRT display.

CCMS - AN EXAMPLE MAPPING PROGRAM

Choropleth mapping is one of the simplest cartographic techniques used to represent a variety of themes from land use patterns to population characteristics. A program, Color Choropleth Mapping System (CCMS), has been developed to provide high quality color choropleth maps for a Hitachi CRT controlled by a Raster Technologies Model One/25 display driver.

The program was developed on a Gould S.E.L. 32/67 computer at the Oregon State University Geography Department. The program is written in FORTRAN 77 and uses the Raster Technologies graphic firmware.

Many concerns of software development discussed in this paper were incoporated into this mapping program. CCMS operates interactively and is menu-driven. The Main menu displays the possible options the program offers. Subsequent menus display instructions. Figure 3 iluustrates the menus displayed by CCMS. The different options and instructions have been written to minimize confusion, making the program "user friendly".

The program offers options which allow the user to create choropleth maps step by step. The user can specify almost every aspect of choropleth mapping from the size of the text letters and legend boxes, to a specific chroma progression used in classification. Or, if the user is unfamiliar with the design aspects of choropleth mapping, an assisted classification option is available. The user must simply provide the observations or statistical data used in classification.

Program Characteristics

Format

CCMS consists of 30 subroutines, each with a specific function. Six of the subroutines create the menus that appear at the bottom of the screen. The main portion of the program determines which option has been chosen by the user from the Main menu. These eight options, each a subroutine, call upon additional subroutines to carry out the tasks specified. Figure 4 illustrates this hierarchy. An explanation of each subroutine can be seen in Table 1.

Hitachi CRT and Raster Technologies Display Driver

The Hitachi CRT is a high resolution (512 X 512) color monitor. The refresh nature of the CRT (30 times a second) permits the image to be changed and or modified in real time. The colors are specified using the RGB model. The numerical range is from 0 to 255, providing 255^3 or 16,581,375 combinations

MAIN MENU	TEXT	POLYGON DRAW	COLOR FILL	LEGEND	ASSISTED CLASSIFICATION	BORDER	CLEAR & RESTART	EXIT
TEXT MENU	LETTER SIZE	ENTER TEXT	GIVE TEXT LOCATION AND WRITE TO SCREEN	EXIT				
COLOR MENU	SEE COLORS	CHOOSE COLOR	FILL AREA	EXIT				
LEGEND MENU	NUMBER OF CLASSES	BOX WIDTH, HEIGHT AND SPACING	POSITION ON SCREEN	ENTER TEXT	EXIT			
POLYGON MENU	SCALE	MOVE	EXIT					
CLEAR MENU	CLEAR PORTION OF SCREEN	CLEAR ENTIRE SCREEN						

Figure 3. Menus displayed by CCMS

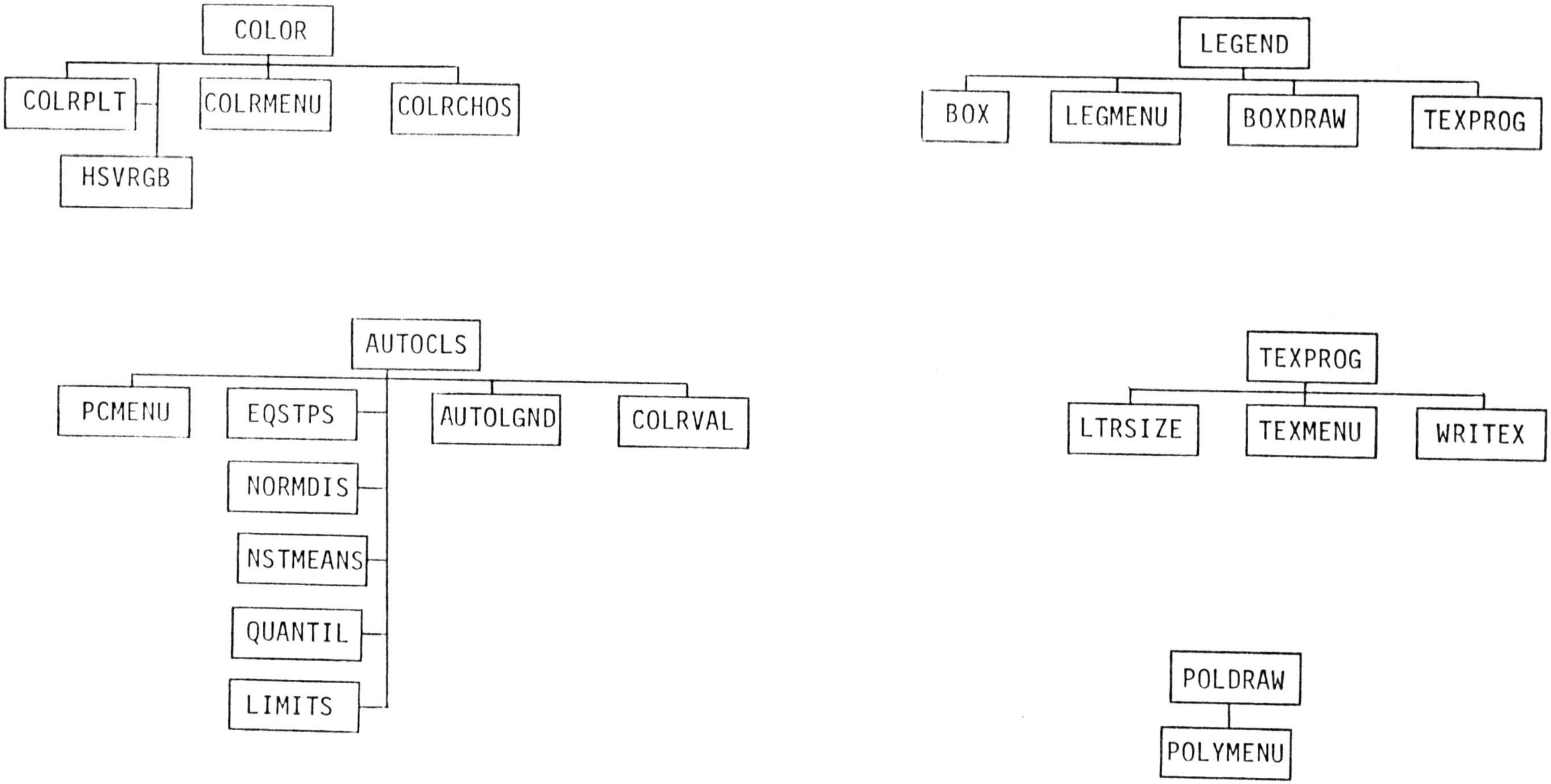

Figure 4. Subroutine hierarchy for program CCMS

Table 1. CCMS Subroutine Explanation

Subroutine Name	Function
TITLE	Writes program title to screen.
MMENU	Writes main menu to screen.
TEXPROG	Reads text entered by user.
LTRSIZE	Displays various letter sizes for the user to choose from and reads the size chosen.
WRITEX	Writes text to screen.
POLDRAW	Plots polygons on screen. Reads polygon data file and scale entered by user. Changes scale and position of polygons at user's request.
POLYMENU	Writes polygon menu to screen.
COLOR	Fills areas (polygons) with color specified by the user from a color palette.
COLRCHOS	Displays possible hues and reads the hue choice made by the user.
COLRPLT	Displays color palettes.
HSVRGB	Converts from H, S, V to R, G, B for creating color palettes.
LEGEND	Reads in the number of classes (boxes) for legend generation.
LEGMENU	Displays legend menu on screen.
BOX	Reads dimensions of legend box (height, width), spacing between boxes, and the direction the boxes are to be plotted (horizontal or vertical) specified by the user.

Table 1 continued.

BOXDRAW	Draws legend boxes on screen. Allows the user to move boxes around screen.
AUTOCLS	Reads statistical data file and polygon centroid data file. Reads the number of classes, color choice, and classification method specified by the user.
AUTOLGND	Automatically writes legend boxes on screen.
COLRVAL	Determines color progression for chosen hue.
BORDER	Reads lower left corner and right upper corner of border area and draws map border.
CLRPTN	Clears a portion of screen or the entire screen dependent upon the user's specification.
CENMAP	Calculates the center of the map body and moves the map to the screen center.
CLRMENU	Clears a window at the bottom of the screen where the menus are displayed.
EQSTPS, NORMDIS, NSTMEANS, QUANTIL, LIMITS - see text.	

of the three additive primary colors.

The Raster Technologies graphics firmware provides the programmer with very powerful graphic commands. The commands are specified in a manner similar to how subroutines are called (Parameters for the graphic commands are placed in parentheses after the command name).

Color Specification

As stated, the Raster Technologies display driver uses the RGB system to specify colors on the screen. Another model previously described is the HSL or HSV (Hue, Saturation, Value) model. In CCMS, a subroutine is called which transforms HSV to RGB to create the color palettes used in classification. The hue co-ordinate is determined by the RGB model (Figure 2). The equations used for the transformation for creating the chroma progression are

```
V=90.48*(10.*a)**.45/255.
S=1.-90.48*(10.*(1.-s))**.45/255.
```

where

s and a are incremented between 0 and 1 by .1 to determine the steps in the chroma progression (Kimerling, 1985).

Six color palettes of the additive and subtractive primary colors (Red, Green, Blue, Yellow, Cyan and Magenta) are created in the program (Figures 5 and 6). Users may see the color palettes by specifying the "See Colors" instruction in the color menu (Figure 3) and then choose the hue they wish to use. The user can pick from the color palette a chroma progression to fill and classify the areas (polygons).

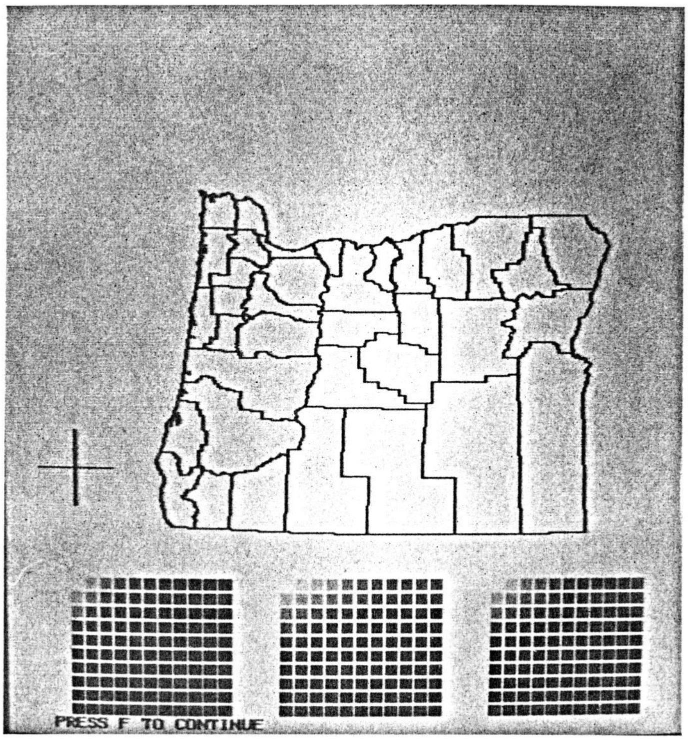

Figure 5. Color reproduction of additive primary color palettes produced by CCMS.

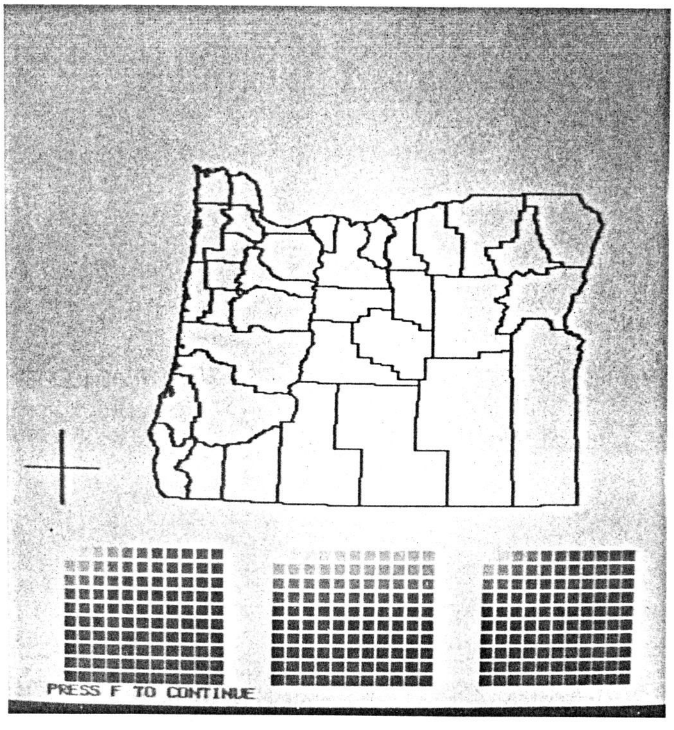

Figure 6. Color reproduction of subtractive primary color palettes produced by CCMS.

In the Assisted Classification option of the program, the color progressions have been predetermined to provide visually contrasting steps within the progression. Research presently being undertaken at O.S.U. on color perception on CRTs will provide a better method of determining color progressions on the Raster Technologies display driven CRT.

Menus

Upon developing CCMS, it was decided that the simplest and easiest way for a program user to give instructions to a program was by pointing to options and instructions displayed in a menu. CCMS utilizes six menus (Figure 3). The Main menu or first menu displayed consists of eight options. Each option assists the user in creating some aspect of the map, except for the last two.

The Text option, as it implies, assists the user in writing text to the screen. When this option is requested, the Text menu appears on the screen displaying four instructions. The "Letter Size" instruction allows the user to choose the letter size from seven different sizes, each demonstrated with an example (Figure 7). The "Enter Text" instruction directs the user's attention to the computer terminal where the user is instructed to type in a text string. The "Give Text Location And Write To Screen" prompts the user to move the cursor to the desired position for writing the text on the screen. The crosshair position will determine the bottom lefthand corner of the first letter in the text string. The "Exit" instruction, as in all exit instructions found in other menus (except for the Main menu), leaves the current option and returns to the previous menu.

The Polygon Draw option plots the polygons on the screen. The user must enter the polygon data file and scale factor. Currently, the scale is entered on the basis of pixels (e.g. 250). The Polygon menu is displayed after the polygons are plotted permitting the user to change the scale or move the polygons to a new position on the screen (Figure 8). (Note: If the screen has been cleared, the program will not request a new polygon data file, but will display the polygon menu requesting a scale or position.)

The Color Fill option is used to fill the polygons and legend boxes with color chosen by the user. In the Color menu, the user can request to see the possible color palettes (Figures 5 and 6), and then choose the hue or color desired. The "Fill Area" instruction will display instructions on how the user can fill areas (e.g. polygons or legend boxes) with colors selected from the color palette (Figure 9).

The Legend option assists the user in creating the legend boxes. When specifying the "Enter Number of Classes" instruction, the user enters the number of classes, or boxes, enters the box dimensions (width and height) in pixels (e.g. 15 X 15) and gives the spacing between the boxes, also in pixels (e.g. 10), and the direction used for plotting the boxes, either vertically or horizontally. The user can then request to position the string of boxes on the screen by using the crosshair to specify the

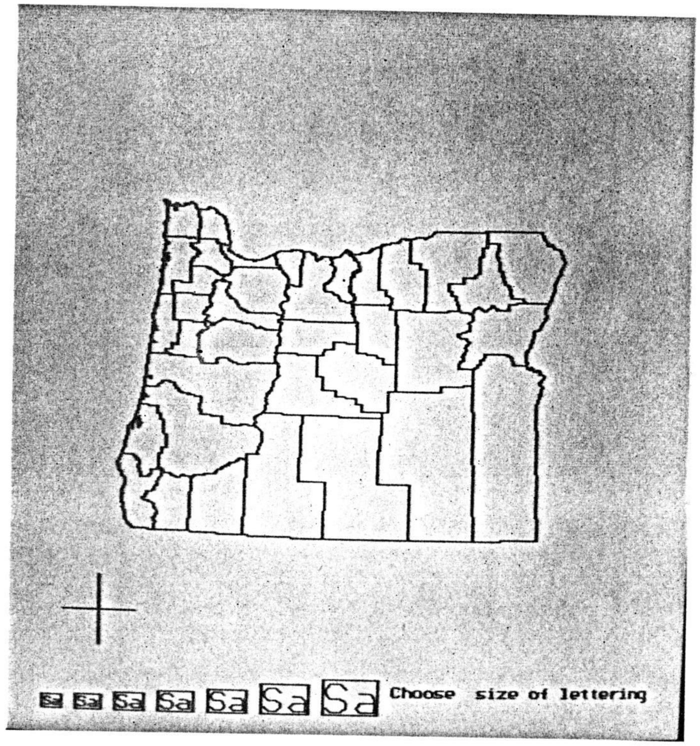

Figure 7. Color reproduction of seven optional letter
sizes for text size of CCMS.

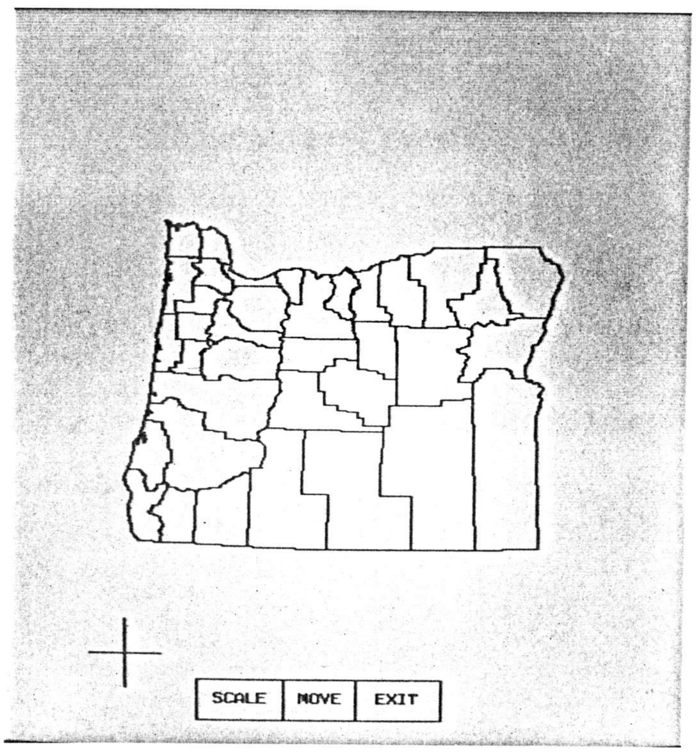

Figure 8. Color reproduction of plotted polygons and the
Polygon menu produced by CCMS.

lower lefthand corner (Figure 10). The Legend menu permits the user to enter the Text option to write the legend text to the screen.

An Assisted Classification option is provided in CCMS which allows the user to choose among five different classification methods to determine class intervals. The five classification methods are:

(1) Equal Steps - divides the statistical data into n number of equal distant steps.

(2) Normal Distribution - class intervals are determined by calculating the mean and standard deviation, and one standard deviation is then added to the mean and subtracted from the mean to determine the class limits.

(3) Nested Means - the mean of the data is calculated dividing the data into two parts. Means are then calculated for the two parts which create the class limits along with the original mean.

(4) Quantiles - ranks the data from lowest to highest and then divides the ranked data into n number of intervals.

(5) Class Limits Entered Manually - lower and upper limits for each class are entered by the user.

The user is queried for the number of classes and for a hue from the computer terminal. The program determines the color progression and classifies each polygon with the appropriate chroma according to the classification method specified (Figure 11).

The legend is automatically written to the screen using this option. The legend is placed vertically on the screen on

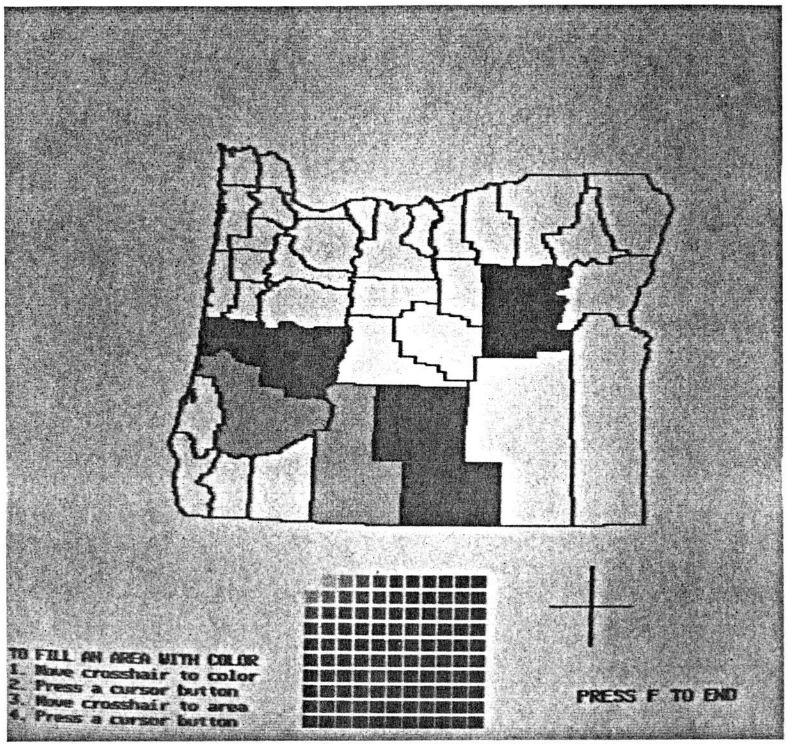

Figure 9. Color reproduction of manual classification using the "Fill Area" instruction of CCMS.

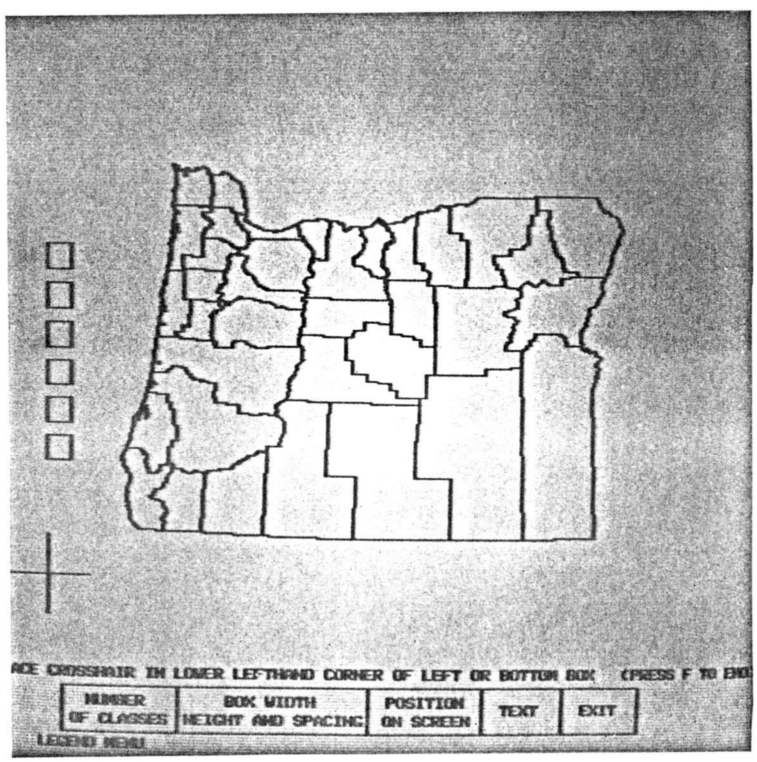

Figure 10. Color reproduction of manual legend generation of CCMS.

the side of the map body with the largest margin (Figure 11). If both margins are too narrow to accommodate a legend, the legend is written horizontally underneath the map body (Figure 12).

The Border option instructs the user in creating a map border. Borders can be created for other aspects of the map (i.e. the legend or map body), but it is important to note that the program uses the map border to determine the map center when centering the map on the screen. So if more than one border is drawn, the last border created should be the border surrounding the entire map.

The Clear and Restart and Exit options perform screen support. The Clear and Restart option displays the Clear menu. The user can clear, or erase a portion of the screen by defining a rectangle surrounding the area, or the entire screen can be cleared, allowing the user to create a new map, and restart without exiting the program. The Exit option exits the program. Upon exiting, the use is queried if the map should be centered on the screen, or left alone.

These options and instructions have just been described in the order in which they occur on the menus. This is not necessarily the order in which they should be specified. An improvement could be made to CCMS by reorganizing the options and instructions displayed in the menus into the logical order in which they should be requested.

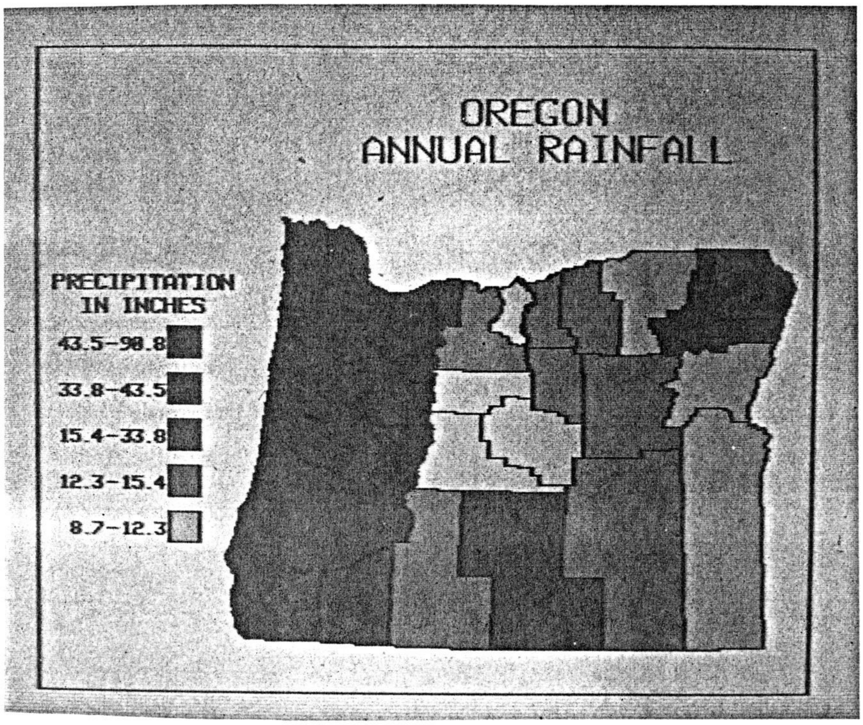

Figure 11. Color reproduction of final map (with vertical legend) using the Assisted Classification option of CCMS.

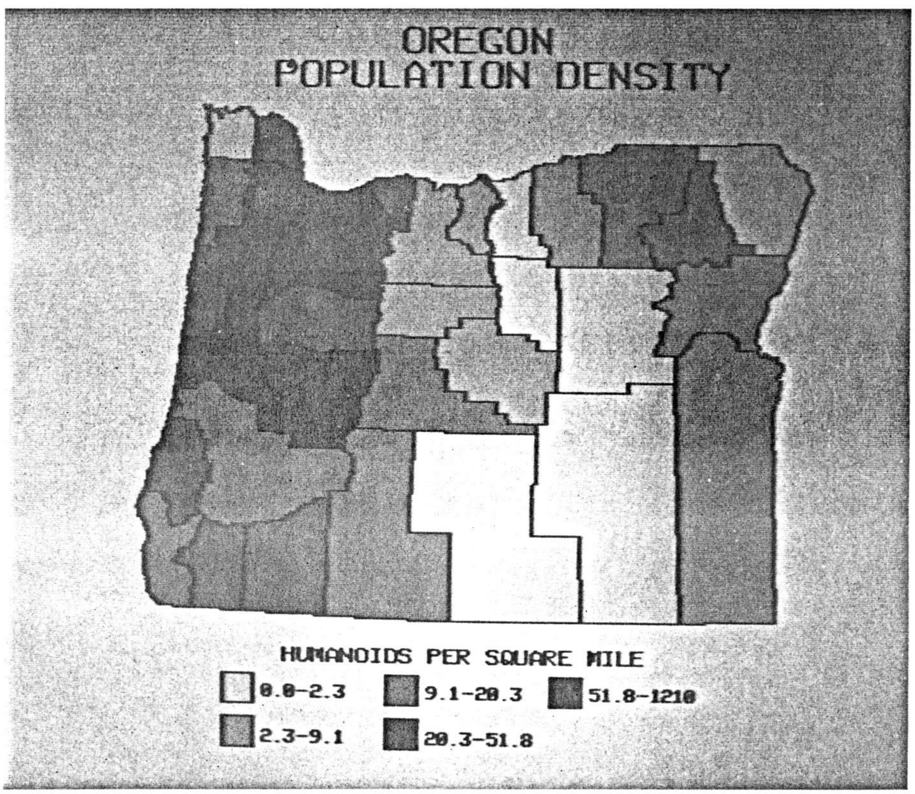

Figure 12. Color reproduction of final map (with horizontal legend) using the Assisted Classification option of CCMS.

Program Usage

A generalized flowchart of the program is shown in Figure 13. To run CCMS, a digitizer tablet and cursor need to be connected to the system. Figure 14 shows all the system components of CCMS.

Menus

As stated, the program is menu-driven. The first menu to appear is the main menu displaying the options the program provides. To request an option the user moves the cursor on the digitizer table, which in turn moves a crosshair on the CRT screen. When the crosshair falls on the menu box containing the desired option, the user then presses a button on the cursor. The next menu to appear on the screen will contain the instructions for the option. Sometimes the user's attention will be directed to the computer terminal to enter data files or answer a question. All the menus in the program can be seen in Figure 3.

Data Input

The first option the user should specify is to draw (plot) the polygons on the graphics screen. The program will request an input file containing the topological data describing the polygons. A sample of this input file is shown in Table 2, and consists of four types of records. The first record type contains the line number and the number of points describing a line. The second record type contains the numbers labeling the polygons to the right and left of the line. The third record

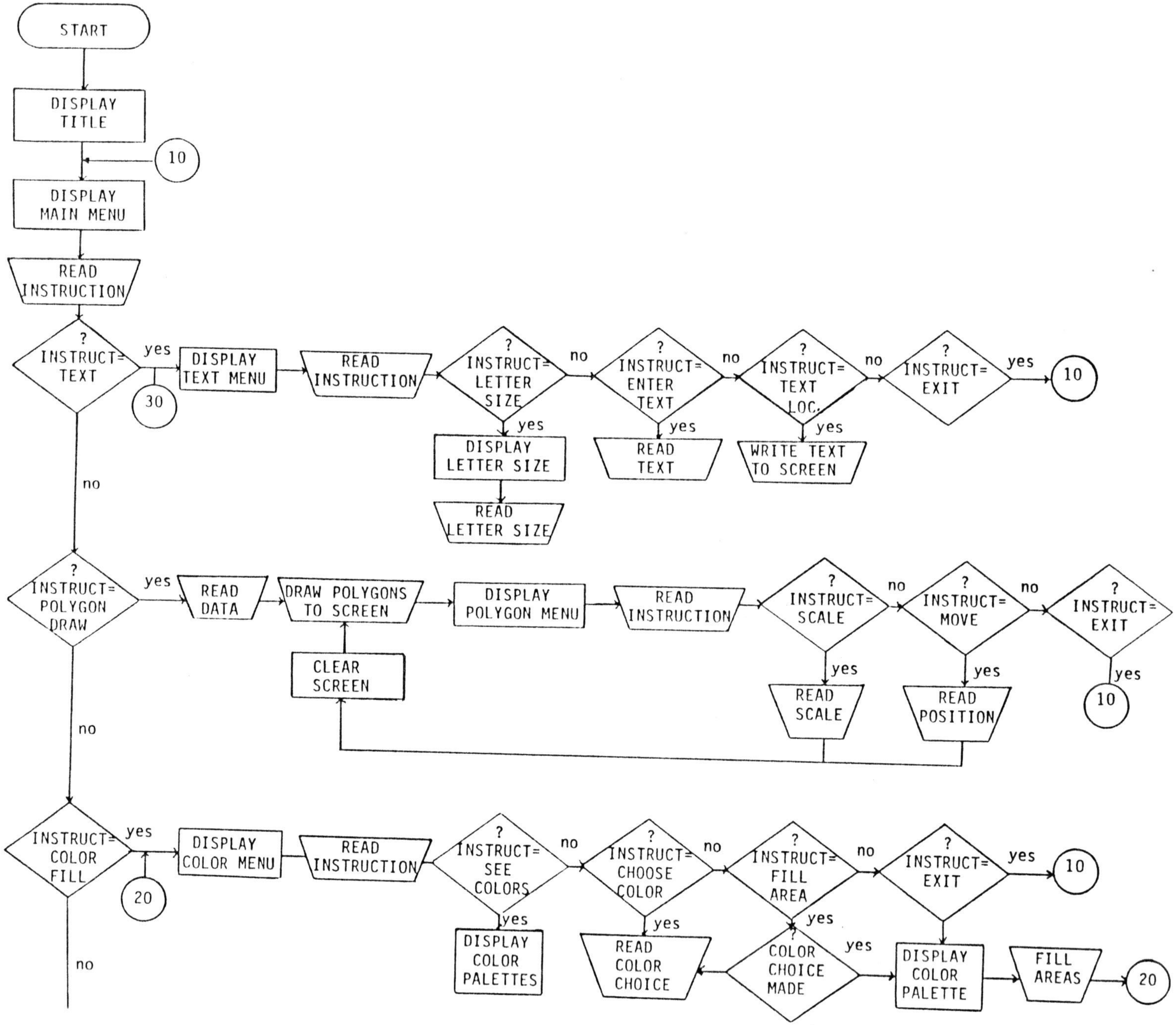

Figure 13. Flowchart of program CCMS

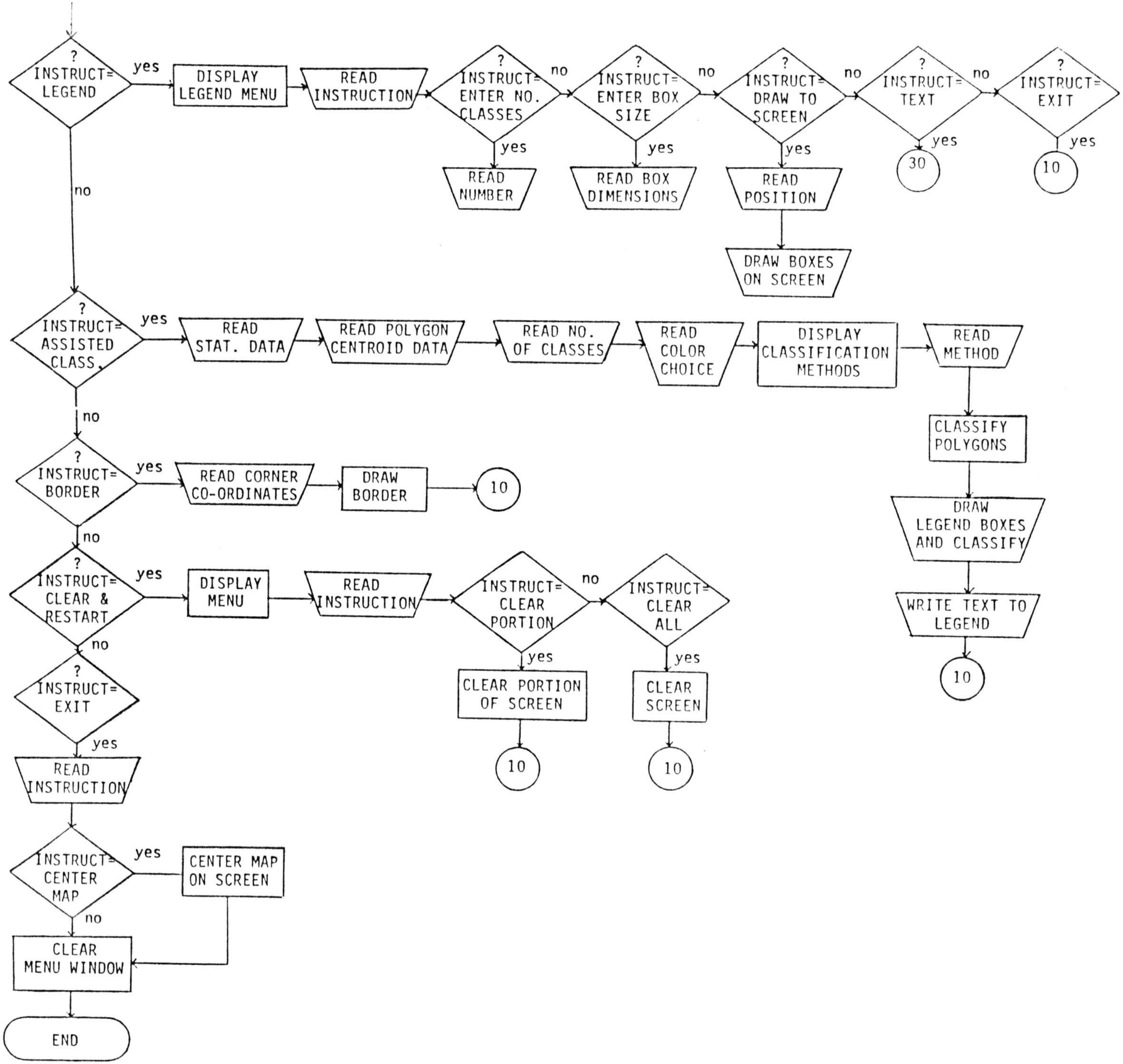

Figure 13 continued.

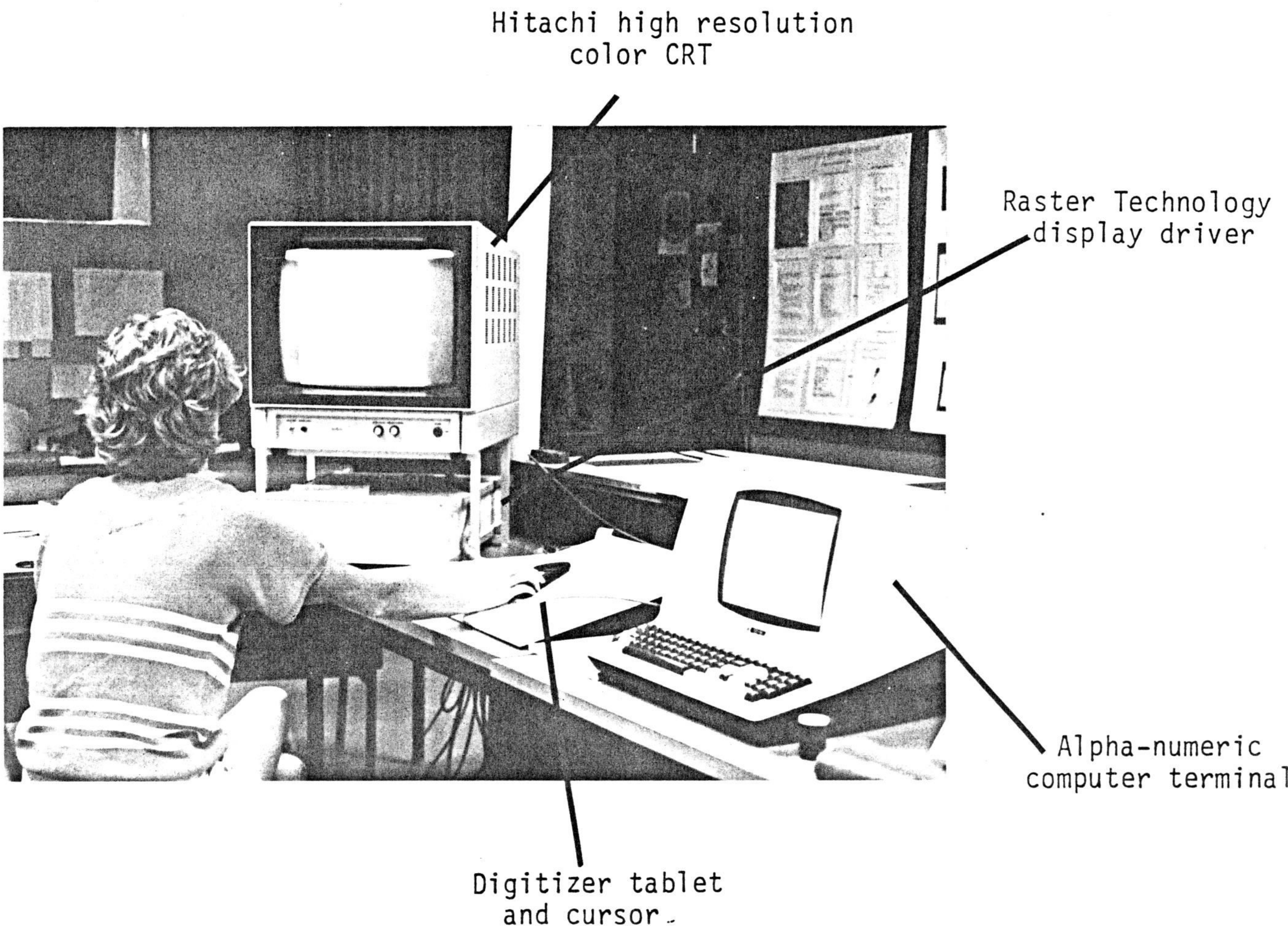

Figure 14. System components of CCMS

type contains the maximum and minimum x and y co-ordinate values for the line. The fourth record type contains a x and y co-ordinate of the line.

Other input files are requested come from the Assisted Classification option of the program. In the Assisted Classification option, the user needs to have two input files. The first file requested is the statistical or observation data file. An example is given in Table 3. Each record contains the data for one area or polygon. The second input file requested contains the centroid of each polygon (Table 4). The statistical data file and centroid data file must be arranged in the same order (e.g. the statistical data for polygon 12 and the centroid for polygon 12 are both the first records in each data file). All other data are entered interactively either from the keyboard or using the cursor buttons.

Output

Presently the only form of output is the map displayed on the screen (Figures 11 and 12). Hardcopies may be acquired by photographing the screen, or using the Rembrandt digital camera.

CONCLUSIONS

The challenge to software developers in years to come will be to incorporate an intelligent set of commands that will enable users to quickly produce maps that will not only please decision makers, but will not offend cartographer's sensibilities. To accomplish this it will be important to have efficient, well-

Table 2. Example of input file for polygon data points

2	2		
23	13		
26984	26721	15670	5869
26984	5869		
26721	15670		
3	2		
23	12		
26721	26707	16723	15670
26721	15670		
26707	16723		
104	5		
0	27		
18525	17683	23796	23350
17683	23350		
18082	23450		
18183	23450		
18375	23650		
18525	23796		

Table 3. Example of input file for observation data

11.25
39.7
47.72
82.07
59.93
56.0
10.49
83.35

Table 4. Example of input file for polygon centroid data points

28309	18913
8954	18202
12983	21283
8322	25233
10297	25075
6347	11961
19224	16227

documented, sound mapping software to provide a reliable link between the user, the data software, and the display hardware.

Monmonier commented that computers have given "aesthetically insensitive, geographically ignorant people the opportunity to create cartographic monstrosities with unprecedented ease" (Monmonier, 1984). To combat this, mapping software will need to become sophisticated enough to detect and prevent cartographic blunders. This is why it is important for software developers to know more than cartographic basics and computer programming. Color theory and production, map design processes, computer and computer peripheral technology, and the interface of humans and computers needs to be understood and taken into consideration when creating computer-assisted cartographic systems.

References

Anderson, Robert H. and Shapiro, Norma Z. 1980. Design considerations for computer-based interactive map display systems. In Computer graphics hardware Vol. 9. ed. P.A. Moore, pp 9-35. Cambridge: Harvard Library of Computer Graphics 1980 Mapping Collection.

Berk, T., Brownston, L. and Kaufman, A. 1982. A new color-naming system for graphics languages. IEEE Computer Graphics and Applications 5:37-44

Boyle, A. R. 1975. Small automated cartographic systems. Proceedings Auto-Carto II 1:298-302.

Carter, J. R. 1984. Computer Mapping: Progress in the 80s. Washington D.C.: Assoc. of American Geographers

Cowen, David J. 1984. Rethinking DIDS: The next generation of interactive color mapping systems. Proceedings, Auto-Carto Six 1:89-92.

Cuff, David J. 1975. Conflicting goals in choosing colors for quantitative maps. Proceedings Auto-Carto IV 1:286-288.

Dobson, M. W. 1983. High resolution microcomputer based color system for examing the human factors aspects of cartographic displays in real-time user environment. Proceedings, Auto-Carto Six 1:352-360.

______. 1984. Effective color display for map task performance in a computer environment. Proceedings, International Symposium on Spatial Data. 2:332-347.

Dudycha, D. J. 1981. Impact of computer cartography. Canadian Cartographer 18:116-147.

Heyn, B. N. 1984. An evaluation and comparison of choropleth map color schemes. Unpublished paper. Univ. of South Carolina

Infante, Carlo, 1979. Display tecnologies in computer aided applications. Proceedings, Auto-Carto 4 1:33-39.

Kimerling, A. J. 1985. A perceptual based color model for computer-assisted map design. Paper presented at the ACSM Fall Technical Meeting, Indianapolis, Indiana.

Lai, Poh-Chin, 1985. Mis-application of automated mapping: An assessment. Proceedings, Auto-Carto 7 1:322-325.

MacDonald, C.L. and Crain, I.K. 1985. Applied computer graphics in a geographic information system: Problems and successes. IEEE CG&A 1:34-39.

Monmonier, M.S. 1982. Computer-Assisted cartography principles and prospects. Englewood Cliffs, NJ: Prentice-Hall, Inc.

______. 1985. Technical Transistion in Cartography. Madison: Univ. of Wisconsin Press.

Orr, J. N. 1980. Matching user requirements with hardware capabilities: Graphic hard-copy output devices. In Computer Graphics Hardware Vol. 9., ed. P.A. Moore, pp. 68-74. Cambridge: Library of Computer Graphics, 1980 Mapping Collection.

Pfefferkorn, C., Burr, D., Harrison, D., Heckman, D. 1985. ACES: A cartographic expert system. Proceedings, Auto-Carto 7 1:399-407.

Poole, Lon, 1981. Apple II user's guide. Berkeley: McGraw-hill Inc.

Rase, W. D. 1975. Computer-assisted thematic mapping with a dedicated microcomputer system. Proceddings, Auto-Carto 4 1:322-329.

Robertson, P. K. and O'Callaghan, J. F. 1986. The generation of color sequence for univariate and bivariate mapping. IEEE Computer Graphics and Applications. 2:24-32.

Robinson, A. H. 1975. Map design. Proceedings, Auto-Carto II 1:9-14.

Robinson, A., Sale, R., Morrison, J., and Muehrcke, P. 1984. Elements of cartography. 5th ed. New York: Wiley.

Robinson, G., and Jackson, M. 1985. Expert systems in map design. Proceddings, Auto-Carto 7 1:430-439.

Sibert, J. L. 1980. Continuous-color choropleth maps. Geo-processing 1:207-216.

Swezey, R. W. and Davis, E. G. 1983. A case study of human factors guidelines in computer graphics. IEEE Computer Graphics & Applications 11:21-29.

Taylor, D. R. F. 1973. The Canadian cartographer and the computer: Present trends and future challenges. Cartographica, Monograph No. 9. ed. A. L. Le Blanc. pp. 1-9.

Tobler, W. R. 1973. Choropleth maps without class intervals? Geographical Analysis 5:262-265.

Traeger, M. 1982. A color system for computer generated cartography. Proceedings, Auto-Carto V 2:687-691.

White, Denis, 1979. Interactive color mapping. Proceedings, Auto-Carto IV 1:272-277.

```
          PROGRAM CCMS
C**********************************************************************
C  COLOR CHOROPLETH MAPPING SYSTEM (CCMS) IS WRITTEN IN
C  FORTAN 77 ON A GOULD S.E.L 32/67 MINI COMPUTER FOR THE
C  RASTER TECHNOLOGIES HIGH RESOLUTION CRT. THE PROGRAM
C  UTILIZES THE RASTER TECHNOLOGIES GRAPHIC FIRMWARE.
C
C  TO RUN CCMS, A DIGITIZER TABLET AND CURSOR NEED TO BE
C  CONNECTED TO THE RASTER TECHNOLOGIES CRT.
C  CCMS IS RUN INTERACTIVELY.  MENU INSTRUCTIONS APPEAR AT
C  THE BOTTOM OF THE SCREEN. USING A DIGITIZER TABLET AND CURSOR
C  THE USER ENTERS THE DESIRED COMMAND BY PLACING A CROSSHAIR
C  OVER THE INSTRUCTION AND PRESSING A CURSOR BUTTON. CCMS THEN
C  WILL DISPLAY THE NEXT MENU CONTAINING THE NEXT SET OF
C  INSTRUCTIONS AND SO ON AND SO FORTH. THE NAME OF EACH MENU
C  DISPLAYED APPEARS IN THE LOWER LEFTHAND CORNER.
C**********************************************************************
          INTEGER*2 IX,IY,IY1,IBUT,IPOSY,CC,IPOSX,PX,PY,BCNTRX,BCNTRY
          INTEGER*2 ACFLAG,HFLG,UBX,UBY,IX1
          INTEGER*4 MX,MIX,MY,MIY,PRX,PRY
          CHARACTER*15 STR1,STR2,STR3,STR4
C
          DATA STR1/'CLEAR PORTION'/
          DATA STR2/'OF SCREEN'/
          DATA STR3/'CLEAR ENTIRE'/
          DATA STR4/'SCREEN'/
          CC=0
          BCNTRX=-250
          BCNTRY=-250
C
C ENTER GRAPHICS MODE AND CLEAR SCREEN
          CALL RTINIT
          CALL ENTGRA
          CALL VAL8(255)
          CALL FLOOD
          CALL VAL8(0)
          CALL PRMFIL(0)
          CALL WARM
C
C WRITE TITLE OF PROGRAM ON SCREEN
          CALL TITLE
C
C WRITE MENU
          CALL MMENU
C ACTIVATE CROSSHAIR
          CALL MACDEF(5)
          CALL CMOVE(5,2)
          CALL MACEND
          CALL XHAIR(0,1)
          CALL BUTTBL(0,5)
          CALL FLUSH
C
C READ COMMAND AND CALL APPROPRIATE SUBROUTINE
 11       CALL READBU(1,1,IBUT,IX,IY)
          CALL CLRMENU
C
          IF (IX .LT. -205) THEN
```

```
         CALL TEXPROG
         CALL MMENU
         ELSE IF (IX .LT. -140) THEN
           CALL POLDRAW(IPOSY,IPOSX,MAXX,MINX,MAXY,MINY,IPROARX,IPROARY,
     *UBX,UBY)
           PX=IPOSX
           PY=IPOSY
           MX=MAXX
           MIX=MINX
           MY=MAXY
           MIY=MINY
           PRX=IPROARX
           PRY=IPROARY
           CALL MMENU
         ELSE IF (IX .LT. -90) THEN
           CALL COLOR(IPOSY,IY1,CC)
           CALL MMENU
         ELSE IF (IX .LT. -40) THEN
           ACFLAG=0
           CALL LEGEND(IPOSY,IY1,IX1)
           CALL MMENU
         ELSE IF (IX .LT.  75) THEN
           ACFLAG=1
           CALL AUTOCLS(PY,PX,MX,MIX,MY,MIY,PRX,PRY,HFLG)
           CALL MMENU
         ELSE IF (IX .LT. 130) THEN
           CALL BORDER(BCNTRX,BCNTRY)
           CALL MMENU
         ELSE IF (IX .LT. 190) THEN
           CALL CLRMENU
C WRITE CLEAR MENU
           CALL MOVABS(-100,-236)
           CALL RECTAN(0,-206)
           CALL RECTAN(100,-206)
C
           CALL MOVABS(-95,-218)
           CALL TEXT1(STR1)
           CALL MOVABS(-80,-230)
           CALL TEXT1(STR2)
C
           CALL MOVABS(10,-218)
           CALL TEXT1(STR3)
           CALL MOVABS(20,-230)
           CALL TEXT1(STR4)
C DETERMINE COMMAND OF CLEAR ROUTINE
           CALL READBU(1,1,IBUT,IX,IY)
           IF (IX .LT. 0) THEN
             CALL CLRPRTN
             CALL MMENU
           ELSE
             CALL VAL8(255)
             CALL FLOOD
             CALL VAL8(0)
             CALL MMENU
           END IF
         ELSE
           CALL CENMAP(BCNTRX,BCNTRY,PX,PY,UBX,UBY,IX1,IY1,HFLG,ACFLAG)
```

```
            GO TO 99
          END IF
C READ NEXT COMMAND
         GO TO 11
C
C CLEAR MENU, TURN CROSSHAIR OFF AND END PROGRAM
 99      CALL CLRMENU
         CALL XHAIR(0,0)
         CALL QUIT
         END
C
C
         SUBROUTINE TITLE
C ********************************************************************
C       SUBROUTINE TO WRITE TITLE ON SCREEN
C ********************************************************************
         CHARACTER*1 TSTR5,TSTR6,TSTR7,TSTR8,R
         CHARACTER*5 TSTR1
         CHARACTER*6 TSTR4
         CHARACTER*7 TSTR3
         CHARACTER*10 TSTR2
C
         DATA TSTR1/'JULIE'/
         DATA TSTR2/'AUTOMATED'/
         DATA TSTR3/'MAPPING'/
         DATA TSTR4/'SYSTEM'/
         DATA TSTR5/'J'/
         DATA TSTR8/'A'/
         DATA TSTR6/'M'/
         DATA TSTR7/'S'/
C
         CALL VALUE(255,125,125)
         CALL TEXTC(70,0)
         CALL MOVABS(-150,150)
         CALL TEXT1(TSTR1)
C
         CALL MOVABS(-100,50)
         CALL TEXT1(TSTR2)
C
         CALL MOVABS(-50,-50)
         CALL TEXT1(TSTR3)
C
         CALL MOVABS(0,-150)
         CALL TEXT1(TSTR4)
C
C BLANK OUT FIRST CHARACTER OF EACH WORD
         CALL VALB(255)
         CALL MOVABS(-150,150)
         CALL TEXT1(TSTR5)
         CALL MOVABS(-100,50)
         CALL TEXT1(TSTR8)
         CALL MOVABS(-50,-50)
         CALL TEXT1(TSTR6)
         CALL MOVABS(0,-150)
         CALL TEXT1(TSTR7)
C
C MAKE FIRST  CHARACTER OF EACH WORD LARGER
         CALL VALUE(255,30,30)
```

```
      CALL TEXTC(110,0)
      CALL MOVABS(-160,140)
      CALL TEXT1(TSTR5)
      CALL MOVABS(-110,40)
      CALL TEXT1(TSTR8)
      CALL MOVABS(-60,-60)
      CALL TEXT1(TSTR6)
      CALL MOVABS(-10,-160)
      CALL TEXT1(TSTR7)
      CALL EMPTYB
C
      WRITE('UT',*)'ENTER A RETURN TO START PROGRAM'
      READ('UT',5) R
  5   FORMAT(A1)
C
      CALL VAL8(255)
      CALL FLOOD
      CALL VAL8(0)
      RETURN
      END
C
C
      SUBROUTINE MMENU
C *******************************************************************
C             SUBROUTINE DRAWS MAIN MENU
C *******************************************************************
      CHARACTER*4 MSTR1,MSTR3,MSTR5,MSTR10,MSTR12
      CHARACTER*5 MSTR4
      CHARACTER*6 MSTR6,MSTR14
      CHARACTER*7 MSTR2,MSTR8,MSTR9
      CHARACTER*8*MSTR11
      CHARACTER*14 MSTR7,MSTR15
C
      DATA MSTR1/'TEXT'/
      DATA MSTR2/'POLYGON'/
      DATA MSTR3/'DRAW'/
      DATA MSTR4/'COLOR'/
      DATA MSTR5/'FILL'/
      DATA MSTR6/'LEGEND'/
      DATA MSTR7/'CLASSIFICATION'/
      DATA MSTR8/'CLEAR &'/
      DATA MSTR9/'RESTART'/
      DATA MSTR10/'EXIT'/
      DATA MSTR11/'ASSISTED'/
      DATA MSTR12/'MAP'/
      DATA MSTR14/'BORDER'/
      DATA MSTR15/'MAIN MENU'/
C
      CALL MOVABS(-240,-236)
      CALL RECTAN(-205,-206)
      CALL RECTAN(-140,-206)
      CALL RECTAN(-90,-206)
      CALL RECTAN(-40,-206)
      CALL RECTAN( 75,-206)
      CALL RECTAN(130,-206)
      CALL RECTAN(190,-206)
      CALL RECTAN(240,-206)
```

```
      CALL TEXTC(18,0)
      CALL MOVABS(-235,-223)
      CALL TEXT1(MSTR1)
C
      CALL MOVABS(-197,-218)
      CALL TEXT1(MSTR2)
      CALL MOVABS(-186,-230)
      CALL TEXT1(MSTR3)
C
      CALL MOVABS(-135,-218)
      CALL TEXT1(MSTR4)
      CALL MOVABS(-130,-230)
      CALL TEXT1(MSTR5)
C
      CALL MOVABS(-85,-223)
      CALL TEXT1(MSTR6)
C
      CALL MOVABS(-5,-218)
      CALL TEXT1(MSTR11)
      CALL MOVABS(-25,-230)
      CALL TEXT1(MSTR7)
C
      CALL MOVABS(95,-218)
      CALL TEXT1(MSTR12)
      CALL MOVABS(85,-230)
      CALL TEXT1(MSTR14)
C
      CALL MOVABS(138,-218)
      CALL TEXT1(MSTR8)
      CALL MOVABS(138,-230)
      CALL TEXT1(MSTR9)
C
      CALL MOVABS(199,-223)
      CALL TEXT1(MSTR10)
C
      CALL MOVABS(-220,-246)
      CALL TEXT1(MSTR15)
C
      RETURN
      END
C
C
      SUBROUTINE TEXPROG
C ****************************************************
C     SUBROUTINE TO ENTER TEXT ON THE SCREEN
C ****************************************************
      INTEGER*2 IX,IY,IBUT
      INTEGER*2 ITXSZ
      LOGICAL*1 FLAG1,FLAG2
      CHARACTER*25 TSTR1
      CHARACTER*80 TEXSTR
C
      DATA TSTR1/'ENTER TEXT AT TERMINAL'/
      FLAG1=.FALSE.
      FLAG2=.FALSE.
C
```

```
C CREATE RECTANGLES
        CALL MOVABS (-170,-236)
        CALL RECTAN (-110,-206)
        CALL RECTAN (-55,-206)
        CALL RECTAN (95,-206)
        CALL RECTAN (155,-206)
C
        CALL TEXTC (20,0)
        CALL MOVABS (-159,-218)
        CALL TEXT1 (STR4)
        CALL MOVABS (-155,-230)
        CALL TEXT1 (STR5)
C
        CALL MOVABS (-101,-218)
        CALL TEXT1 (STR6)
        CALL MOVABS (-97,-230)
        CALL TEXT1 (STR7)
C
        CALL MOVABS (-45,-218)
        CALL TEXT1 (STR1)
        CALL MOVABS (-50,-230)
        CALL TEXT1 (STR2)
C
        CALL MOVABS (110,-223)
        CALL TEXT1(STR9)
C
        CALL MOVABS(-220,-246)
        CALL TEXT1(STR10)
C
        RETURN
        END
C
C
        SUBROUTINE LTRSIZE(ISZ)
C ****************************************************************
C      SUBROUTINE TO DETERMINE LETTER SIZE OF TEXT
C ****************************************************************
        INTEGER*2 IBUT,IX,IY,ISZ
        CHARACTER*2 STR1
        CHARACTER*26 STR2
C
        DATA STR1 /'Sa'/
        DATA STR2 /'Choose  size of lettering'/
C
C DRAW LETTER SIZE EXAMPLES
        CALL MOVABS (-200,-236)
        CALL TEXTC (16,0)
        CALL TEXT1 (STR1)
        CALL MOVREL (-2,-2)
        CALL RECREL (14,10)
C
        CALL MOVABS (-176,-236)
        CALL TEXTC (20,0)
        CALL TEXT1 (STR1)
        CALL MOVREL (-2,-2)
        CALL RECREL (18,11)
C
        CALL MOVABS (-148,-236)
        CALL TEXTC (25,0)
        CALL TEXT1 (STR1)
```

```
          CALL MOVREL (-2,-2)
          CALL RECREL (20,13)
C
          CALL MOVABS (-118,-236)
          CALL TEXTC (30,0)
          CALL TEXT1 (STR1)
          CALL MOVREL (-2,-2)
          CALL RECREL (24,15)
C
          CALL MOVABS (-84,-236)
          CALL TEXTC (35,0)
          CALL TEXT1 (STR1)
          CALL MOVREL (-2,-2)
          CALL RECREL (26,17)
C
          CALL MOVABS (-48,-236)
          CALL TEXTC (45,0)
          CALL TEXT1 (STR1)
          CALL MOVREL (-2,-2)
          CALL RECREL (32,22)
C
          CALL MOVABS (-6,-236)
          CALL TEXTC (55,0)
          CALL TEXT1 (STR1)
          CALL MOVREL (-2,-2)
          CALL RECREL (38,26)
C ASK FOR SIZE OF LETTER
          CALL MOVABS (40,-224)
          CALL TEXTC(20,0)
          CALL TEXT1(STR2)
          CALL EMPTYB
C
C DETERMINE LETTER SIZE
          CALL READBU (1,1,IBUT,IX,IY)
            IF (IX .GT.-8) THEN
              ISZ=55
            ELSE IF (IX .GT. -50) THEN
              ISZ=45
            ELSE IF (IX .GT. -86) THEN
              ISZ=35
            ELSE IF (IX .GT. -120) THEN
              ISZ=30
            ELSE IF (IX .GT. -150) THEN
              ISZ=25
            ELSE IF (IX .GT. -178) THEN
              ISZ=20
            ELSE
              ISZ=16
            END IF
C
          CALL CLRMENU
          RETURN
          END
C
C
```

```
      SUBROUTINE WRITEX(ITXSZ,TEXSTR)
C ***********************************************************************
C  SUBROUTINE TO LOCATE AND WRITE TEXT ON SCREEN
C ***********************************************************************
      INTEGER*2 IX,IY,IX1,IY1,IBUT,ITXSZ
      CHARACTER*20 WSTR1,WSTR2
      CHARACTER*60 TEXSTR
C
      DATA WSTR1/'PRESS F TO END'/
      DATA WSTR2/'PRESS A TO DELETE'/
      IX1=-255
      IY1=-255
C
      CALL MOVABS(0,-200)
      CALL TEXTC(20,0)
      CALL TEXT1(WSTR1)
      CALL MOVABS(0,-190)
      CALL TEXT1(WSTR2)
C
 201  CALL READBU(1,1,IBUT,IX,IY)
        IF (IBUT .EQ. 10) GO TO 215
        IF (IBUT .EQ. 15) THEN
 203      CALL VAL8(255)
          CALL MOVABS(0,-200)
          CALL TEXTC(20,0)
          CALL TEXT1(WSTR1)
          CALL MOVABS(0,-190)
          CALL TEXT1(WSTR2)
          CALL VAL8(0)
          RETURN
        END IF
 215  CALL VAL8(255)
      CALL MOVABS(IX1,IY1)
      CALL TEXTC(ITXSZ,0)
      CALL TEXT1(TEXSTR)
      IF (IBUT .EQ. 10) GO TO 203
C
      CALL MOVABS (IX,IY)
      CALL VAL8(0)
      CALL TEXT1(TEXSTR)
      CALL EMPTYB
      IX1=IX
      IY1=IY
      GO TO 201
      END
C
C
     SUBROUTINE POLDRAW(IPOSY,IPOSX,MAXX,MINX,MAXY,MINY,IPROARX,
    *IPROARY,UBX,UBY)
C ***********************************************************************
C          SUBROUTINE TO DRAW POLYGONS ON SCREEN
C ***********************************************************************
      INTEGER*4 MAXX,MINX,MAXY,MINY,MAXXL,MINXL,MAXYL,MINYL
      INTEGER*2 LNUM, NUMPTS, ISTOP, ISTART, RPOLY, LPOLY
      INTEGER*2 IX,IY,IPROPX,IPROPY,IPOSX,IPOSY,ISZ,LBX,LBY,UBX,UBY
      INTEGER*4 PTS(3025,2), INDX(256,2)
      REAL*4 RPROPX,RPROPY
      CHARACTER*8 DATAFIL
      CHARACTER*45 PSTR2
```

```
          CHARACTER*65 PSTR4
C
          DATA PSTR2 /'ENTER DATA FILE AND SCALE AT TERMINAL'/
          DATA PSTR4  /'MOVE CROSSHAIR TO THE DESIRED BOTTOM LEFTHAND COR
     *NER OF MAP'/
C
C CHECK IF SUBROUTINE HAS BEEN CALLED, AND IF TRUE SKIP TO POLYMENU
          IF (IFLAG .EQ. 1) GO TO 205
C
          IFLAG=0
          MAXX=0
          MAXY=0
          MINX=100000
          MINY=100000
          ISTART=1
C
          CALL MOVABS(-200,-200)
          CALL TEXT1(PSTR2)
          CALL EMPTYB
C
C  READ IN POLYGON DATA
        WRITE('UT',*)'ENTER POLYGON DATA FILE: '
        READ('UT',100) DATAFIL
  100   FORMAT(A8)
        OPEN(UNIT=10,FILE=DATAFIL,BLOCKED=.TRUE.,ERR=900,IOSTAT=ISTAT,
     *STATUS='OLD')
          IF (ISTAT .EQ. 10) GOTO 900
C
          CALL VAL8(255)
          CALL MOVABS(-200,-200)
          CALL TEXT1(PSTR2)
          CALL VAL8(0)
C  READ IN DATA
        DO 1 I=1,500
          READ(10,101,END=201,ERR=902) LNUM,NUMPTS
          READ(10,101,END=901,ERR=902) RPOLY, LPOLY
  101     FORMAT(2(1X,I6))
          READ(10,102,END=901,ERR=902) MAXXL,MINXL,MAXYL,MINYL
  102     FORMAT(4(1X,I6))
C  DETERMINE MINIMUM AND MAXIMUM POINTS
          MAXX=MAXO(MAXX,MAXXL)
          MAXY=MAXO(MAXY,MAXYL)
          MINX=MINO(MINX,MINXL)
          MINY=MINO(MINY,MINYL)
C
C  STORE BEGINNING AND ENDING POINTERS FOR EACH LINE
          ISTOP=ISTART + (NUMPTS-1)
          INDX(I,1)=ISTART
          INDX(I,2)=ISTOP
C
C  READ IN POLYGON POINTS
           DO 2 J=ISTART,ISTOP
             READ(10,101,END=901,ERR=902) PTS(J,1),PTS(J,2)
  2        CONTINUE
           ISTART=ISTOP+1
  1     CONTINUE
C
```

```
C  DETERIMINE SCALE FACTOR FOR DATA
 201      WRITE('UT',*)'ENTER THE MAP SIZE IN PIXELS: '
          READ('UT',103) ISZ
 103      FORMAT(I3)
          MXDIS=MAXX-MINX+1
          MYDIS=MAXY-MINY+1
          RPROPX=FLOAT(MXDIS)/FLOAT(MYDIS)
          RPROPY=FLOAT(MYDIS)/FLOAT(MXDIS)
          IF (RPROPX .GT. 1.0) RPROPX=1.0
          IF (RPROPY .GT. 1.0) RPROPY=1.0
C
C  SCALE THE PLOT AREA TO THE PROPORTION OF THE DATA
          IPROPX=INT(ISZ*RPROPX) + 1
          IPROPY=INT(ISZ*RPROPY) + 1
          IPROARX=MXDIS/IPROPX
          IPROARY=MYDIS/IPROPY
C  DETERMINE BOTTOM LEFTHAND CONER OF PLOT
          IPOSX= -ISZ/2 + (ISZ-IPROPX)/2
          IPOSY= -ISZ/2 + (ISZ-IPROPY)/2
C
C  CLEAR JUST THE POLYGON WINDOW
 202      IF (IFLAG .EQ. 1) THEN
            CALL VAL8(255)
            CALL WINDOW(LBX,LBY,UBX,UBY)
            CALL CLEAR
            CALL VAL8(0)
            CALL WINDOW(-256,-256,255,255)
          END IF
C
          IFLAG=1
          CALL XHAIR(0,0)
C  PLOT POLYGONS
          DO 4 I=1,LNUM
           ISTART=INDX(I,1)
           ISTOP=INDX(I,2)
C  MOVE TO BEGINNING OF LINE
           IX=(PTS(ISTART,1)-MINX)/IPROARX + IPOSX
           IY=(PTS(ISTART,2)-MINY)/IPROARY + IPOSY
           CALL MOVABS(IX,IY)
C DRAW LINE
           DO 5 K=ISTART+1, ISTOP
             IX=(PTS(K,1)-MINX)/IPROARX + IPOSX
             IY=(PTS(K,2)-MINY)/IPROARY + IPOSY
             CALL DRWABS(IX,IY)
  5       CONTINUE
  4     CONTINUE
        CALL EMPTYB
C
C DETERMINE POLYGON WINDOW
          LBX=IPOSX
          LBY=IPOSY
          UBX=(MAXX-MINX)/IPROARX+IPOSX
          UBY=(MAXY-MINY)/IPROARY+IPOSY
C
          CALL XHAIR(0,1)
 205      CALL POLYMENU
C
```

```
          CALL READBU(1,1,IBUT,IX,IY)
          IF (IX .LT. -40) THEN
C CHANGE SCALE
            GO TO 201
          ELSE IF (IX .LT. 10) THEN
C MOVE POLYGONS TO NEW POSITION
            CALL MOVABS(-220,-200)
            CALL TEXT1(PSTR4)
            CALL READBU (1,1,IBUT,IX,IY)
            IPOSX=IX
            IPOSY=IY
            CALL VALS(255)
            CALL MOVABS(-220,-200)
            CALL TEXT1(PSTR4)
            CALL VALS(0)
            GO TO 202
          ELSE
            CALL CLRMENU
          END IF
          GOTO 999
C
C WRITE ERROR STATEMENTS
 900      WRITE('UT',*) 'ERROR ENCOUNTED WHILE ATTEMPTING TO OPEN FILE'
          GOTO 999
 901      WRITE('UT',*) 'ERROR! END OF FILE REACHED'
          GOTO 999
 902      WRITE('UT',*) 'ERROR ENCOUNTERED WHILE READING FILE ',DATAFIL
C
 999   CLOSE(10)
       RETURN
       END
C
C
          SUBROUTINE POLYMENU
C ********************************************************************
C       SUBROUTINE TO DRAW POLYGON MENU ON SCREEN
C ********************************************************************
          CHARACTER*4 PCTR2,PCTR3
          CHARACTER*5 PCTR1
C
          DATA PCTR1 /'SCALE'/
          DATA PCTR2 /'MOVE'/
          DATA PCTR3 /'EXIT'/
C
          CALL MOVABS(-100,-236)
          CALL RECTAN(-40,-206)
          CALL RECTAN(10,-206)
          CALL RECTAN(70,-206)
C
          CALL TEXTC(20,0)
          CALL MOVABS(-38,-223)
          CALL TEXT1(PCTR1)
C
          CALL MOVABS(-29,-223)
          CALL TEXT1(PCTR2)
          CALL MOVABS(25,-223)
          CALL TEXT1(PCTR3)
C
          RETURN
          END
```

```
      SUBROUTINE COLOR(IPOSY,IY1,CC)
C ******************************************************************
C                 SUBROUTINE TO COLOR MAP
C ******************************************************************
      INTEGER*2 IX,IY,IBUT,IPOSY,IY1,MINY,CC,FLAG3
      CHARACTER*20 STR1
      CHARACTER*40 STR2,STR3,STR4,STR5,STR6
C
      DATA STR1 /'PRESS F TO END'/
      DATA STR2/'TO FILL AN AREA WITH COLOR'/
      DATA STR3/'1. Move crosshair to color'/
      DATA STR4/'2. Press a cursor button'/
      DATA STR5/'3. Move crosshair to area '/
      DATA STR6/'4. Press a cursor button'/
      FLAG3=0
C
C DETERMINE MINIMUM Y POSITION ON MAP (IPOSY->POLYGON, IY1->LEGEND)
        IF (IPOSY .LT. IY1) THEN
          MINY=IPOSY
        ELSE
          MINY=IY1
        END IF
C
      CALL TEXTC(20,0)
      CALL COLRMENU
C READ INSTRUCTION
 44   CALL READBU(1,1,IBUT,IX,IY)
      IF (IX .LT. -35) THEN
C DRAW ALL COLOR PALETTES TO SCREEN
        CALL CLRMENU
        FLAG3=0
        CALL COLRPLT(MINY,FLAG3,CC,IYWDOW)
        CALL PRMFIL(0)
        CALL COLRMENU
      ELSE IF (IX .LT. 25) THEN
C READ IN COLOR CHOICE
        CALL CLRMENU
        CALL COLRCHOS(CC,FLAG3)
        CALL COLRMENU
      ELSE IF (IX .LT. 75) THEN
C FILL AREAS WITH COLOR FROM A COLOR PALETTE
        IF (CC .EQ. 0) THEN
          WRITE('UT',*) 'ERROR! NO COLOR HAS BEEN CHOSEN'
          GO TO 44
        END IF
        CALL CLRMENU
        CALL MOVABS(100,-223)
        CALL TEXT1(STR1)
        CALL COLRPLT(MINY,FLAG3,CC,IYWDOW)
C WRITE DIRECTIONS ON HOW TO FILL AN AREA WITH COLOR TO SCREEN
        CALL VAL8(0)
        CALL TEXTC(16,0)
        CALL MOVABS(-250,-200)
        CALL TEXT1(STR2)
        CALL MOVABS(-250,-210)
        CALL TEXT1(STR3)
        CALL MOVABS(-250,-220)
        CALL TEXT1(STR4)
        CALL MOVABS(-250,-230)
```

```
          CALL TEXT1(STR5)
          CALL MOVABS(-250,-240)
          CALL TEXT1(STR6)
          CALL TEXTC(20,0)
C
C DEFINE MACRO TO FILL AN AREA
          CALL MACDEF(10)
          CALL CMOVE(0,2)
          CALL AREA1
          CALL MACEND
C
C READ COLOR FROM PALETTE
 50       CALL READBU(1,1,IBUT,IX,IY)
          IF (IBUT .EQ. 15) GO TO 54
          CALL CMOVE(0,2)
          CALL RDPIXR(0)
C READ POSITION OF AREA TO BE FILLED
          CALL READBU(1,1,IBUT,IX,IY)
          IF (IBUT .EQ. 15) GO TO 54
          CALL MACRO(10)
          GO TO 50
C
C CLEAR COLOR PALETTE FROM SCREEN
 54       CALL WINDOW(-256,-256,250,IYWDOW)
          CALL VAL8(255)
          CALL CLEAR
          CALL VAL8(0)
          CALL WINDOW(-256,-256,255,255)
          CALL PRMFIL(0)
          CALL COLRMENU
        ELSE
          GO TO 901
        END IF
        GO TO 44
C
 901    CALL CLRMENU
        RETURN
        END
C
C
        SUBROUTINE COLRMENU
C *******************************************************************
C                 SUBROUTINE TO DRAW COLOR MENU
C *******************************************************************
        CHARACTER*4 CSTR4,CSTR5,CSTR6
        CHARACTER*6 CSTR2,CSTR3
        CHARACTER*10 CSTR1,CSTR7
C
        DATA CSTR1 /'SEE COLORS'/
        DATA CSTR2 /'CHOOSE'/
        DATA CSTR3 /'COLOR'/
        DATA CSTR4 /'FILL'/
        DATA CSTR5 /'AREA'/
        DATA CSTR6 /'EXIT'/
        DATA CSTR7/'COLOR MENU'/
C
        CALL MOVABS(-125,-236)
```

```
      CALL RECTAN(-35,-206)
      CALL RECTAN(25,-206)
      CALL RECTAN(75,-206)
      CALL RECTAN(125,-206)
C
      CALL MOVABS(-115,-223)
      CALL TEXT1(CSTR1)
C
      CALL MOVABS(-26,-218)
      CALL TEXT1(CSTR2)
      CALL MOVABS(-24,-230)
      CALL TEXT1(CSTR3)
C
      CALL MOVABS(35,-218)
      CALL TEXT1(CSTR4)
      CALL MOVABS(35,-230)
      CALL TEXT1(CSTR5)
C
      CALL MOVABS(83,-223)
      CALL TEXT1(CSTR6)
C
      CALL MOVABS(-220,-246)
      CALL TEXT1(CSTR7)
C
      RETURN
      END
C
C
      SUBROUTINE COLRCHOS(CC,FLAG3)
C ********************************************************************
C                SUBROUTINE TO CHOOSE COLOR
C ********************************************************************
      INTEGER*2 CC,FLAG3
      CHARACTER*7 CCSTR1,CCSTR2,CCSTR3,CCSTR4,CCSTR5,CCSTR6
C
      DATA CCSTR1 /'RED'/
      DATA CCSTR2 /'GREEN'/
      DATA CCSTR3 /'BLUE'/
      DATA CCSTR4 /'YELLOW'/
      DATA CCSTR5/'CYAN'/
      DATA CCSTR6/'MAGENTA'/
C
C WRITE COLORS AVAILABLE TO USER
      CALL MOVABS(-160,-236)
      CALL RECTAN(-125,-206)
      CALL RECTAN(-80,-206)
      CALL RECTAN(-30,-206)
      CALL RECTAN( 30,-206)
      CALL RECTAN(85,-206)
      CALL RECTAN(145,-206)
C
      CALL MOVABS(-153,-223)
      CALL TEXT1(CCSTR1)
C
      CALL MOVABS(-120,-223)
      CALL TEXT1(CCSTR2)
C
```

```
          CALL MOVABS(-67,-223)
          CALL TEXT1(CCSTR3)
C
          CALL MOVABS(-23,-223)
          CALL TEXT1(CCSTR4)
C
          CALL MOVABS(43,-223)
          CALL TEXT1(CCSTR5)
C
          CALL MOVABS( 90,-223)
          CALL TEXT1(CCSTR6)
C
C READ COLOR CHOSEN AND ASSIGN TO VARIABLE CC
          CALL READBU(1,1,IBUT,IX,IY)
          IF (IX .LT. -120)THEN
            CC=1
          ELSE IF (IX .LT. -90) THEN
            CC=2
          ELSE IF (IX .LT. -30)THEN
            CC=3
          ELSE IF (IX .LT. 30) THEN
            CC=4
          ELSE IF (IX .LT. 95) THEN
            CC=5
          ELSE
            CC=6
          END IF
C
          FLAG3=1
          CALL CLRMENU
          RETURN
          END
C
C
      SUBROUTINE COLRPLT(MINY,FLAG3,CC,IYWDOW)
C ******************************************************************
C     SUBROUTINE TO DRAW COLOR PALETTE ON SCREEN
C ******************************************************************
          INTEGER *2 X,Y,XO,YO,R,G,B,RD,M,IDIFF,CC,FLAG3,IYWDOW,MINY
          CHARACTER*20 CPSTR1
C
          DATA CPSTR1/'PRESS F TO CONTINUE'/
C
C  SCALE COLORS TO FIT ON SCREEN
          RD=7
          M=10
          IYWDOW=-121
          IF(MINY .LT. -122) THEN
            IDIFF=IABS(-130 - MINY)
            IDIFF= FLOAT(IDIFF)/10.0 + 0.5
            M=M-IDIFF
            RD=RD-IDIFF
            IYWDOW=-252 + 10*M + 2 + RD
          END IF
C
C CALCULATE COLOR PALETTE
      YO=-252
```

```
      K3=1
      YMC=0
C INITIALIZE HUE
      DO 34 J=1,2
         CALL XHAIR(0,0)
         DO 35 I=1,3
C
C DRAW COLOR PALETTE FOR CHOOSEN COLOR USED TO FILL AN AREA
         IF (FLAG3 .EQ. 1) THEN
           IF (CC .LT. 4) THEN
             I=CC
           ELSE
             YMC=.5
             I=CC-3
           END IF
           XO=-69
           GO TO 40
         END IF
C
        XO = -210 + 141*(I-1)
 40     H =((I+YMC)-1)*120
C CALCULATE VALUE
        DO 36 J=1,10
          W=FLOAT(J)/10
          Y = YO + J*M  + 2
C CALCULATE SATURATION
          DO 37 K2=0,10
            S=FLOAT(K2)/10.
            X = XO + K2*M  + 2
C CALCULATE R,G,B VALUES AND DRAW PALETTE
            CALL HSVRGB(H,S,W,R,G,B,K3)
            CALL VALUE(R,G,B)
            CALL MOVABS(X,Y)
            CALL PRMFIL(1)
            CALL RECREL(RD,RD)
 37       CONTINUE
 36     CONTINUE
         IF (FLAG3 .EQ. 1) THEN
           FLAG3=0
           CC=0
           CALL XHAIR(0,1)
           RETURN
         END IF
 35   CONTINUE
C
C SHOW COLORS UNTIL BUTTON IS PUSHED
        CALL MOVABS(-220,-250)
        CALL TEXT1(CPSTR1)
        CALL XHAIR(0,1)
        CALL READBU(1,1,IBUT,IX,IY)
        CALL WINDOW (-240,-255,240,IYWDOW)
        CALL VAL8(255)
        CALL CLEAR
        CALL VAL8(0)
        CALL WINDOW(-256,-256,255,255)
         YMC=.5
 34     CONTINUE
```

```
      RETURN
      END
C
C
      SUBROUTINE HSVRGB(H,S,A,R,G,B,K)
C ********************************************************************
C     SUBROUTINE TO CALCULATE COLOR PALETTE
C ********************************************************************
      REAL*4 H,H1
      INTEGER*2 R,G,B
C
      IF(K .EQ. 0) GO TO 7
      V=90.48*(10.*A)**.45/255.
 7    IF (S .NE. 0) GO TO 10
      R=V*255.
      G=R
      B=R
      RETURN
C
 10   IF (K .EQ. 0) GO TO 8
      S=1.-90.48*(10.*(1.-S))**.45 /255.
 8    IF (H .NE. 360.) GO TO 11
      H1=0.
      GO TO 12
 11   H1=H/60.
 12   I=INT(H1)
      F=H1-I
      P=V*(1.-S)*255.
      Q=V*(1.-S*F)*255.
      T=V*(1.-(S*(1.-F)))*255.
      V1=V*255.
C
C BRANCH TO HUE CHOSEN
      GO TO (1,2,3,4,5,6),I+1
C RED
 1    R=V1
      G=T
      B=P
      RETURN
C
C GREEN
 2    R=Q
      G=V1
      B=P
      RETURN
C
C BLUE
 3    R=P
      G=V1
      B=T
      RETURN
C
C YELLOW
 4    R=P
      G=Q
      B=V1
      RETURN
C
C CYAN
 5    R=T
      G=P
      B=V1
      RETURN
```

```
C
C MAGENTA
 6        R=V1
          G=P
          B=Q
          RETURN
          END
C
C
          SUBROUTINE LEGEND(IPOSY,IY1,IX1)
C *********************************************************************
C                SUBROUTINE TO CREATE LEGEND OF MAP
C *********************************************************************
          INTEGER*2 IX,IY,IBUT,IBH,IBW,IBSP,IDIR,NB
          CHARACTER*60 LSTR10,LSTR11
C
          DATA LSTR10/'Give number of classes by pressing cursor'/
          DATA LSTR11/'(f to end)'/
C
          CALL TEXTC(16,0)
          CALL LEGMENU
 205      CALL READBU(1,1,IBUT,IX,IY)
C
          IF (IX .LT. -115) THEN
            CALL MOVABS(-200,-200)
            CALL TEXT1(LSTR10)
            CALL MOVABS(70,-200)
            CALL TEXT1(LSTR11)
            CALL READBU(1,1,IBUT,IX,IY)
            NB=IBUT
 201        CALL READBU(1,1,IBUT,IX,IY)
            IF (IBUT .NE. 15) THEN
              NB=NB*10 + IBUT
              GO TO 201
            END IF
            CALL VAL8(255)
            CALL MOVABS(-200,-200)
            CALL TEXT1(LSTR10)
            CALL MOVABS(70,-200)
            CALL TEXT1(LSTR11)
            CALL VAL8(0)
C  CREATE LEGEND BOXES
          ELSE IF (IX .LT. 0 ) THEN
            CALL CLRMENU
            CALL BOX(IBH,IBW,IBSP,IDIR)
            CALL LEGMENU
C  POSITION BOXES ON SCREEN
          ELSE IF (IX .LT. 70) THEN
            CALL BOXDRAW(IBH,IBW,IBSP,IDIR,NB,IY1,IX1)
C TEXT LEGEND
          ELSE IF (IX .LT. 120) THEN
            CALL CLRMENU
            CALL TEXPROG
            CALL TEXTC(16,0)
            CALL LEGMENU
C EXIT AND CLEAR LEGEND MENU
          ELSE
            GO TO 999
```

```
      END IF
      GO TO 205
C
 999  CALL CLRMENU
      RETURN
      END
C
C
      SUBROUTINE LEGMENU
C ***********************************************************************
C          SUBROUTINE TO DRAW LEGEND MENU
C ***********************************************************************
      CHARACTER*4 LSTR8,LSTR9,LSTR6
      CHARACTER*5 LSTR7
      CHARACTER*6 LSTR1
      CHARACTER*8 LSTR5
      CHARACTER*9 LSTR3,LSTR10
      CHARACTER*10 LSTR2
      CHARACTER*11 LSTR11
      CHARACTER*18 LSTR4
C
      DATA LSTR1 /'NUMBER'/
      DATA LSTR2 /'OF CLASSES'/
      DATA LSTR3 /'BOX WIDTH'/
      DATA LSTR4 /'HEIGHT AND SPACING'/
      DATA LSTR5 /'POSITION'/
      DATA LSTR6 /'TEXT'/
      DATA LSTR9 /'EXIT'/
      DATA LSTR10/'ON SCREEN'/
      DATA LSTR11/'LEGEND MENU'/
C
      CALL MOVABS(-186,-236)
      CALL RECTAN (-115,-206)
      CALL RECTAN (  0,-206)
      CALL RECTAN (70,-206)
      CALL RECTAN (120,-206)
      CALL RECTAN (170,-206)
C
      CALL MOVABS (-170,-218)
      CALL TEXT1 (LSTR1)
      CALL MOVABS (-180,-230)
      CALL TEXT1 (LSTR2)
C
      CALL MOVABS ( -85,-218)
      CALL TEXT1 (LSTR3)
      CALL MOVABS (-110,-230)
      CALL TEXT1 (LSTR4)
C
      CALL MOVABS ( 12,-218)
      CALL TEXT1 (LSTR5)
      CALL MOVABS ( 10,-230)
      CALL TEXT1 (LSTR10)
C
      CALL MOVABS (82,-223)
      CALL TEXT1 (LSTR6)
C
      CALL MOVABS (133,-223)
```

```
          CALL TEXT1 (LSTR9)
C
          CALL MOVABS(-200,-246)
          CALL TEXT1(LSTR11)
C
          RETURN
          END
C
C
          SUBROUTINE BOX(IBW,IBH,IBSP,IDIR)
C ***********************************************************************
C      SUBROUTINE TO DETERMINE BOX SIZE (WIDTH, HEIGHT AND SPACING)
C ***********************************************************************
          INTEGER*2 IX,IY,IBUT,IBH,IBW,IBSP,IDIR,IRMDR,IDIG(4),ND
          INTEGER*2 IRETURN
          CHARACTER*3 LCHAR,LBCHAR(3)
          CHARACTER*40 LSTR11,LSTR12,LSTR13,LSTR14,LSTR15,LSTR16
          CHARACTER*60 LSTR17
C
          DATA LSTR11 /'(F TO END)'/
          DATA LSTR12 /'GIVE BOX WIDTH'/
          DATA LSTR13 /'GIVE BOX HEIGHT'/
          DATA LSTR14 /'GIVE SPACING BETWEEN BOXES'/
          DATA LSTR15 /'PRESS 1 FOR HORIZONTAL SPACING'/
          DATA LSTR16 /'PRESS 2 FOR VERTICAL SPACING'/
          DATA LSTR17 /'BOX EXAMPLE, (TO CONTINUE PRESS F)'/
C
C ENTER LEGNED BOX WIDTH
          CALL MOVABS (-200,-210)
          CALL TEXT1 (LSTR12)
          CALL MOVABS (-50,-210)
          CALL TEXT1(LSTR11)
C
          CALL READBU(1,1,IBUT,IX,IY)
          IBW=IBUT
 202      CALL READBU(1,1,IBUT,IX,IY)
            IF (IBUT .NE. 15) THEN
              IBW=IBW*10 + IBUT
              GO TO 202
            END IF
C CORRECT FOR ZERO CURSOR BUTTON NOT WORKING
          IF (IBW .EQ. 1 .OR. IBW .EQ. 2) IBW=IBW*10
C SET VARIABLES FOR INTEGER->CHARACTER CONVERSION AND WRITE TO SCREEN
          IRMDR=IBW
          IRETURN=1
          GOTO 206
 207      CALL MOVABS(100,-210)
          CALL TEXT1(LBCHAR(IRETURN))
C
C ENTER LEGEND BOX HEIGHT
          CALL MOVABS(-200,-220)
          CALL TEXT1(LSTR13)
          CALL MOVABS(-50,-220)
          CALL TEXT1(LSTR11)
C
          CALL READBU(1,1,IBUT,IX,IY)
          IBH=IBUT
```

```
      IBUT=0
 203  CALL READBU(1,1,IBUT,IX,IY)
        IF (IBUT .NE. 15) THEN
          IBH=IBH*10 + IBUT
          GO TO 203
        END IF
C CORRECT FOR ZERO CURSOR BUTTON NOT WORKING
      IF (IBH .EQ. 1 .OR. IBH .EQ. 2) IBH=IBH*10
C SET VARIABLES FOR INTEGER->CHARACTER CONVERSION AND WRITE TO SCREEN
      IRMDR=IBH
      IRETURN=2
      GOTO 206
 208  CALL MOVABS(100,-220)
      CALL TEXT1(LBCHAR(IRETURN))
C
C ENTER SPACING BETWEEN LEGEND BOXES
      CALL MOVABS(-200,-230)
      CALL TEXT1(LSTR14)
      CALL MOVABS( 10,-230)
      CALL TEXT1(LSTR11)
      CALL READBU(1,1,IBUT,IX,IY)
      IBSP=IBUT
      IBUT=0
 204  CALL READBU(1,1,IBUT,IX,IY)
        IF (IBUT .NE. 15) THEN
          IBSP=IBSP*10 + IBUT
          GO TO 204
        END IF
C CORRECT FOR ZERO BUTTON NOT WORKING
      IF (IBSP .EQ. 1 .OR. IBSP .EQ. 2)IBSP=IBSP*10
C SET VARIABLES FOR INTEGER->CHARACTER CONVERSION AND WRITE TO SCREEN
      IRMDR=IBSP
      IRETURN=3
      GOTO 206
 209  CALL MOVABS(100,-230)
      CALL TEXT1(LBCHAR(IRETURN))
C
C ENTER HORIZONTAL OR VERTICAL SPACING
 205  CALL CLRMENU
      CALL MOVABS(-200,-210)
      CALL TEXT1(LSTR15)
      CALL MOVABS(-200,-220)
      CALL TEXT1(LSTR16)
      CALL READBU(1,1,IBUT,IX,IY)
      IDIR=IBUT
      IF (IDIR .NE. 1 .AND. IDIR .NE. 2) GOTO 205
C
C DRAW BOXES TO SHOW USER WHAT THEY HAVE CHOOSEN
      CALL CLRMENU
      CALL MOVABS(0,-250)
      CALL RECREL(IBW,IBH)
        IF (IDIR.EQ.1) THEN
          IX=IBW+IBSP
          CALL MOVABS (IX,-250)
        ELSE
          IY=-250 + IBH + IBSP
          CALL MOVABS (0,IY)
```

```
            END IF
          CALL RECREL(IBW,IBH)
C
          CALL MOVABS(-240,-245)
          CALL TEXT1(LSTR17)
          CALL READBU(1,1,IBUT,IX,IY)
          GOTO 99
C
C CLEAR CHARACTER ARRAY
 206      DO 4 I=1,3
            LCHAR(I:I)=' '
 4        CONTINUE
C *****CALCULATE NUMBER OF DIGITS*****
          DO 1 I=0,10
            IF (10**I .GT. IRMDR) GOTO 101
   1      CONTINUE
C
C SEPARATE NUMBERS INTO DIGITS
 101      ND=I
          DO 2 I=ND,1,-1
            IDIG(I)= INT(IRMDR/10**(I-1))
            IRMDR=IRMDR-(IDIG(I)*10**(I-1))
 2        CONTINUE
C CONVERT DIGITS TO CHARACTERS
          N=1
          DO 3 I=ND,1,-1
            LCHAR(N:N)=CHAR(IDIG(I) + 48)
            N=N+1
 3        CONTINUE
          LBCHAR(IRETURN)=LCHAR
          GOTO (207,208,209) IRETURN
C
C  CLEAR WINDOW
 99       CALL CLRMENU
          RETURN
          END
C
C
        SUBROUTINE BOXDRAW(IBH,IBW,IBSP,IDIR,NB,IY1,IX1)
C *******************************************************************
C             SUBROUTINE TO DRAW LEGEND BOXES ON SCREEN
C *******************************************************************
          INTEGER*2 IX,IY,IBUT,IX1,IX2,IY1,IY2,IBH,IBW,IBSP,IDIR,NB
          CHARACTER*25 LSTR19
          CHARACTER*70 LSTR18
C
          DATA LSTR18 /'PLACE CROSSHAIR IN LOWER LEFTHAND CORNER OF LEFT
     *OR BOTTOM BOX'/
          DATA LSTR19 /'(PRESS F TO END)'/
C
           IX1=-255
           IY1=-255
C
          CALL MOVABS(-230,-200)
          CALL TEXTC(16,0)
          CALL TEXT1(LSTR18)
          CALL MOVABS (160,-200)
```

```
          CALL TEXT1(LSTR19)
C
 205      CALL READBU(1,1,IBUT,IX,IY)
C ERASE DIRECTIONS AND RETURN
            IF (IBUT.EQ.15) THEN
              CALL VAL8(255)
              CALL MOVABS(-230,-200)
              CALL TEXT1(LSTR18)
               CALL MOVABS(160,-200)
               CALL TEXT1(LSTR19)
              CALL VAL8(0)
              RETURN
            END IF
C ERASE OLD BOXES
          CALL VAL8(255)
          CALL MOVABS(IX1,IY1)
          CALL RECREL(IBW,IBH)
            IF (IDIR .EQ. 1) THEN
              DO 1 J=1,NB-1
                IX2=IX1 + IBW*J + IBSP*J
                CALL MOVABS(IX2,IY1)
                CALL RECREL(IBW,IBH)
  1           CONTINUE
            ELSE
              DO 2 J=1,NB-1
                IY2=IY1 + IBH*J + IBSP*J
                CALL MOVABS(IX1,IY2)
                CALL RECREL(IBW,IBH)
  2           CONTINUE
            END IF
C PLOT BOXES
          CALL VAL8(0)
          CALL MOVABS(IX,IY)
          CALL RECREL(IBW,IBH)
            IF (IDIR .EQ. 1) THEN
              DO 3 J=1,NB-1
              IX2=IX + IBW*J + IBSP*J
                CALL MOVABS(IX2,IY)
                CALL RECREL(IBW,IBH)
  3          CONTINUE
            ELSE
              DO 4 J=1,NB-1
                IY2=IY + IBH*J + IBSP*J
                CALL MOVABS(IX,IY2)
                CALL RECREL(IBW,IBH)
  4           CONTINUE
            END IF
          IX1=IX
          IY1=IY
          GO TO 205
          END
C
```

```
C
      SUBROUTINE AUTOCLS(PY,PX,MX,MIX,MY,MIY,PRX,PRY,HFLG)
C ***********************************************************************
C    SURBOUTINE TO ASSIST CLASSIFICATION
C ***********************************************************************
      INTEGER*2 IX,IY,IBUT,ACFLAG,LGND(2,15),IST,HFLG,DASHLEN,NDG(30)
      INTEGER*2 NB,CC,R,G,B,NP,IDIG(15),LTX,LTY,ICLASS(100),PX,PY,IC
      INTEGER*2 NDMAX,MNDG
      INTEGER*4 PCNTR(50,2),MX,MIX,MY,MIY,PRX,PRY
      REAL*4 RMDR,CB(20),STAT(100),STATMIN,STATMAX,DIV
      CHARACTER*1 CDIG(10),DASH
      CHARACTER*10 LGTXT(20),TEXT
      CHARACTER*60 ACSTR1,ACSTR2
      CHARACTER*8 STATFIL,PCNTRFIL
C
      DATA ACSTR1/'ASSISTED CLASSIFICATION, ENTER DATA AT TERMINAL'/
      DATA ACSTR2/'USE CURSOR TO ENTER CLASSIFICATION METHOD'/
      DATA DASH/'-'/
C
      HFLG=0
      DFLAG=0
      NP=0
      NDMAX=0
      STATMAX=-99999.
      STATMIN=100000.
C
C INITIALIZE ARRAYS
      DO 16 INT=1,15
          IDIG(INT)=0
 16   CONTINUE
C
      DO 17 INT=1,10
        CDIG(INT)=' '
 17   CONTINUE
C
C ENTER DATA FOR CLASSIFICATION
      CALL TEXTC(20,0)
      CALL MOVABS(-150,-223)
      CALL TEXT1(ACSTR1)
      CALL EMPTYB
C
C
      WRITE('UT',*) 'ENTER STATISTICAL DATA FILE: '
      READ('UT',100) STATFIL
 100  FORMAT(A8)
      OPEN(UNIT=11,FILE=STATFIL,BLOCKED=.TRUE.,ERR=991,STATUS='OLD',
     *IOSTAT=ISTAT)
      IF (ISTAT .EQ. 10) GOTO 991
C
      WRITE('UT',*) 'ENTER POLYGON CENTROID FILE: '
      READ('UT',100) PCNTRFIL
      OPEN(UNIT=12,FILE=PCNTRFIL,BLOCKED=.TRUE.,ERR=991,STATUS='OLD',
     *IOSTAT=ISTAT)
      IF (ISTAT .EQ. 10) GOTO 991
C
```

```
C   READ STATISTICAL DATA
          DO 1 J=1,100
            READ(11,110,END=901,ERR=992) STAT(J)
  110       FORMAT(F8.2)
            IF (STAT(J) .EQ. -9999.) GOTO 115
            STATMAX=AMAX1(STATMAX,STAT(J))
            STATMIN=AMIN1(STATMIN,STAT(J))
  115       NP=NP+1
    1     CONTINUE
          REWIND 11
C
C ENTER POLYGON CENTER POINTS
 901      DO 2 K=1,NP
            READ(12,120,END=999,ERR=992) PCNTR(K,1),PCNTR(K,2)
 120        FORMAT(2(1X,I6))
 2        CONTINUE
          REWIND 12
C
C SCALE POLYGON CENTER POINTS TO SCREEN
          IF (MX .EQ. 0 .AND. PX .EQ. 0) THEN
            WRITE('UT',*) 'ERROR! POLYGON DRAW MUST BE CALLED FIRST'
          GO TO 999
          END IF
C
          DO 3 J=1,NP
            PCNTR(J,1)=(PCNTR(J,1)-MIX)/PRX+PX
            PCNTR(J,2)=(PCNTR(J,2)-MIY)/PRY+PY
 3        CONTINUE
C
C ENTER NUMBER OF CLASSES
          WRITE('UT',*) 'ENTER NUMBER OF CLASSES FOR CLASSIFICATION: '
          READ('UT',130) NB
 130      FORMAT(I2)
C
C ENTER COLOR CHOICE
          WRITE('UT',*) 'RED=1,     GREEN=2, BLUE=3'
          WRITE('UT',*) 'YELLOW=4, CYAN=5,  MAGENTA=6'
          WRITE('UT',*) 'ENTER COLOR CHOICE: '
          READ('UT',140) CC
 140      FORMAT(I2)
C
          WRITE('UT',*) 'ENTER CLASSIFICATION METHOD AT SCREEN'
C
C SET CLASS BOUNDARIES WITH LOWEST AND HIGHEST STAT VALUES
          CB(1)=STATMIN
          CB(NB+1)=STATMAX
C
C PICK CLASSIFICATION TYPE AND CLASSIFY
          CALL CLRMENU
          CALL PCMENU
          CALL MOVABS(-200,-245)
          CALL TEXT1(ACSTR2)
```

```
C
          CALL READBU(1,1,IBUT,IX,IY)
          IF (IX .LT. -185) THEN
            CALL EQSTPS(STAT,STATMIN,STATMAX,NB,NP,ICLASS,CB)
          ELSE IF (IX .LT. -155) THEN
            CALL  NORMDIS(STAT,NP,ICLASS,CB)
          ELSE IF (IX .LT. 55) THEN
            CALL NSTMEAN(STAT,NP,ICLASS,NB,CB)
          ELSE IF (IX .LT. 125) THEN
            CALL QUANTIL(NB,STAT,NP,ICLASS,CB)
          ELSE
            CALL LIMITS(STAT,ICLASS,NP,NB,CB)
          END IF
C
C FILL POLYGONS WILL CLASS COLOR
          DO 4 K=1,NP
            IF (STAT(K) .EQ. -9999.)THEN
              R=0
              G=0
              B=0
              GOTO 155
            END IF
            IC=ICLASS(K)
            CALL COLRVAL(IC,CC,R,G,B)
 155        CALL VALUE(R,G,B)
            IX=PCNTR(K,1)
            IY=PCNTR(K,2)
            CALL MOVABS(IX,IY)
            CALL AREA1
 4        CONTINUE
          CALL EMPTYB
C
C DRAW LEGNED BOXES
          CALL CLRMENU
          CALL AUTOLGND(NB,PX,PY,MX,MIX,MY,MIY,PRX,PRY,LLX,LLY,LGND,HFLG)
C
C  FILL LEGEND BOXES WITH CLASS COLOR
          DO 12 I=1,NB
            IC=I
            CALL COLRVAL(IC,CC,R,G,B)
            CALL VALUE(R,G,B)
            IX=LGND(1,I)
            IY=LGND(2,I)
            CALL MOVABS(IX,IY)
            CALL AREA1
 12       CONTINUE
          CALL EMPTYB
C
C CONVERT CLASS BOUNDARIES TO TEXT AND WRITE TO LEGEND
          CALL VAL8(0)
          DO 5 K=1,NB+1
          IST=1
          TEXT='            '
```

```
C CALCULATE NUMBER OF DIGITS IN EACH NUMBER
            RMDR=ABS(CB(K))
            DO 6 J=0,10
              IF (10**J .GT. RMDR) GOTO 65
 6          CONTINUE
C
 65         ND=J
            IF(CB(K) .GT. -1 .AND. CB(K) .LT. 1) THEN
              ND=1
              J=1
            END IF
C SEPARATE NUMBERS INTO DIGITS
            DO 7 L=ND,1,-1
              IDIG(L)=INT(RMDR/10**(L-1))
              RMDR=RMDR-(IDIG(L)*10**(L-1))
 7          CONTINUE
            RMDR=RMDR*10
C
C DETERMINE FRACTIONAL PORTION OF NUMBER
            IF(RMDR .NE. 0) THEN
              ND=ND+2
              IST=IST+2
              CDIG(1)=CHAR(INT(RMDR)+48)
              CDIG(2)='.'
            END IF
C
C DETERMINE IF NUMBER IS NEGATIVE AND ADD NEGATIVE SIGN
            IF (CB(K) .LT. 0) THEN
              ND=ND+1
              CDIG(ND)=DASH
            END IF
C
C CONVERT TO CHARACTER
            DO 8 N=1,J
              CDIG(IST)=CHAR(IDIG(N) + 48)
              IST=IST+1
 8          CONTINUE
C
            DO 9 N=1,ND
              TEXT(N:N)=CDIG(ND-(N-1))
 9          CONTINUE
            LGTXT(K)=TEXT
            NDG(K)=ND
            IF (ND .GT. NDMAX) NDMAX=ND
 5        CONTINUE
          DASHLEN=NDMAX*7+5
C
C WRITE LEGEND TEXT TO SCREEN
          CALL TEXTC(16,0)
C DETERMINE HORIZONTAL OR VERTICAL LEGEND
          IF (HFLG .NE. 1) THEN
C CHECK WHICH MARGIN AND SET BEGINNING COORDINATES
            LTX=LLX+21
            LTY=LLY+4
```

```
C
          DO 11 J=1,NB
           IF (LLX .LT. PX) THEN
             LTX=LLX-(NDG(J)*6)-DASHLEN-2
           END IF
           CALL MOVABS(LTX,LTY)
           CALL TEXT1(LGTXT(J))
           CALL MOVABS(LLX-DASHLEN,LTY)
           CALL TEXT1(DASH)
           CALL MOVABS(LLX-DASHLEN+8,LTY)
           CALL TEXT1(LGTXT(J+1))
           LTY=LTY+24
 11       CONTINUE
C WRITE HORIZONTAL LEGEND
         ELSE
           LTX=LLX+21
           LTY=LLY+4
           NT=INT(FLOAT(NB)/2. + .5)
           N=1
           DO 15 J=1,NT
             DO 14 K=1,2
               CALL MOVABS(LTX,LTY)
               CALL TEXT1(LGTXT(N))
               CALL MOVABS(LTX+NDG(N)*6+1,LTY)
               CALL TEXT1(DASH)
               CALL MOVABS(LTX+NDG(N)*6+8,LTY)
               CALL TEXT1(LGTXT(N+1))
               LTY=LTY-24
               IF (N .EQ. NB) GOTO 150
               N=N+1
 14          CONTINUE
             LTX=LTX+85
             LTY=LTY+48
 15        CONTINUE
 150     END IF
         GOTO 999
C
 991     WRITE('UT',*) 'ERROR OCCURED WHILE OPENING A FILE'
         GOTO 999
 992     WRITE('UT',*) 'ERROR OCCURED WHILE READING A FILE'
C
 999     CLOSE (11)
         CLOSE (12)
         CALL CLRMENU
         RETURN
         END
C
C
         SUBROUTINE PCMENU
C ***********************************************************************
C    SUBROUTINE TO CREATE MENU OF CLASSIFICATION METHODS
C ***********************************************************************
         CHARACTER*5 PCSTR1,PCSTR2
         CHARACTER*11 PCSTR7
         CHARACTER*16 PCSTR5,PCSTR6
         CHARACTER*20 PCSTR3,PCSTR4,PCSTR8
```

```
      DATA PCSTR1/'EQUAL'/
      DATA PCSTR2/'STEPS'/
      DATA PCSTR3/'NORMAL DISTRIBUTION'/
      DATA PCSTR4/'(4 CLASSES ONLY)'/
      DATA PCSTR5/'NESTED MEANS'/
      DATA PCSTR6/'QUANTILES'/
      DATA PCSTR7/'ENTER CLASS'/
      DATA PCSTR8/'LIMITS SEPARATELY'/
C
      CALL TEXTC(16,0)
      CALL MOVABS(-240,-236)
      CALL RECTAN(-185,-206)
      CALL RECTAN(-55,-206)
      CALL RECTAN(55,-206)
      CALL RECTAN(125,-206)
      CALL RECTAN(235,-206)
C
      CALL MOVABS(-230,-218)
      CALL TEXT1(PCSTR1)
      CALL MOVABS(-230,-230)
      CALL TEXT1(PCSTR2)
C
      CALL MOVABS(-178,-218)
      CALL TEXT1(PCSTR3)
      CALL MOVABS(-173,-230)
      CALL TEXT1(PCSTR4)
C
      CALL MOVABS(-40,-218)
      CALL TEXT1(PCSTR5)
      CALL MOVABS(-50,-230)
      CALL TEXT1(PCSTR4)
C
      CALL MOVABS( 62,-223)
      CALL TEXT1(PCSTR6)
C
      CALL MOVABS(140,-218)
      CALL TEXT1(PCSTR7)
      CALL MOVABS(130,-230)
      CALL TEXT1(PCSTR8)
C
      CALL EMPTYB
      RETURN
      END
C
C
      SUBROUTINE EQSTPS(STAT,STATMIN,STATMAX,NB,NP,ICLASS,CB)
C *********************************************************************
C     SUBROUTINE TO CALCULATED EQUAL STEPPED INTERVALS FOR CLASSES
C *********************************************************************
      INTEGER*2 NB,NP,NUMSTP,EQSTP,ICLASS(100)
      REAL*4 STAT(100),STATMIN,STATMAX,CB(20)
```

```
C
C CALCULATE STEP
          EQSTP=INT(STATMAX-STATMIN)/NE
          DO 1 M=1,NP
C CLASSIFTY EACH POLYGON
            DO 2 J=1,NE
              NUMSTP=NE-J
              CB(NUMSTP+1)=STATMIN+NUMSTP*EQSTP
              IF (STAT(M) .GE. STATMIN+NUMSTP*EQSTP) THEN
                ICLASS(M)=NUMSTP+1
                GOTO 1
              END IF
 2          CONTINUE
 1        CONTINUE
          RETURN
          END
C
C
          SUBROUTINE NORMDIS(STAT,NP,ICLASS,CB)
C ******************************************************************
C SUBROUITINE TO CALCULATE CLASS INTERVALS USING NORMAL DISTRIBUTION
C ******************************************************************
          INTEGER*2 ICLASS(100),NP
          REAL*4 MEAN,STANDEV,SUMSTAT,SUMSQ,STAT(100),CB(20)
C
          SUMSQ=0
          SUMSTAT=0
C
C CALCULATE SUM SQUARED AND TOTAL SUM OF STAT DATA
          DO 1 N=1,NP
            IF (STAT(N) .EQ. -9999) GOTO 1
            SUMSQ=SUMSQ+STAT(N)**2
            SUMSTAT=SUMSTAT+STAT(N)
 1        CONTINUE
C
C CALCULATE MEAN AND STANDARD DEVIATION OF DATA
          MEAN=SUMSTAT/FLOAT(NP)
          CB(3)=MEAN
          STANDEV=SQRT((SUMSQ-SUMSTAT**2/FLOAT(NP))/FLOAT(NP-1))
          CLIMIT1=MEAN-STANDEV
          CB(2)=CLIMIT1
          CLIMIT2=MEAN+STANDEV
          CB(4)=CLIMIT2
C
C ASSIGN POLYGON TO APROPRIATE CLASS
          DO 2 J=1,NP
            IF (STAT(J) .LE. CLIMIT1) THEN
              ICLASS(J)=1
            ELSE IF(STAT(J) .LE. MEAN) THEN
              ICLASS(J)=2
            ELSE IF(STAT(J) .LE. CLIMIT2) THEN
              ICLASS(J)=3
            ELSE
              ICLASS(J)=4
            END IF
 2        CONTINUE
```

```
C
          RETURN
          END
C
C
          SUBROUTINE NSTMEAN(STAT, NP,ICLASS,NB,CB)
C ***********************************************************************
C  SUBROUTINE TO CALCULATED CLASS INTEVERALS USING NESTED MEANS
C ***********************************************************************
          INTEGER*2 NP,NB,ICLASS(100)
          REAL*4 STAT(100), CB(20), MEAN, SUMSTAT,MEAN1,MEAN2,SS1,SS2
C
C DETERMINE MEAN
          DO 1 N=1,NP
            IF (STAT(N) .EQ. -9999.) GOTO 1
            SUMSTAT=SUMSTAT + STAT(N)
 1        CONTINUE
          MEAN=SUMSTAT/NP
          CB(3)=MEAN
          K1=0
          K2=0
C DETERMINE NESTED MEANS
          DO 2 J=1,NP
            IF (STAT(J) .EQ. -9999.) GOTO 2
            IF (STAT(J) .LT. MEAN) THEN
              SS1=SS1+STAT(J)
              K1=K1+1
            ELSE
              SS2=SS2+STAT(J)
              K2=K2+1
            END IF
 2        CONTINUE
C
          MEAN1=SS1/K1
          MEAN2=SS2/K2
          CB(2)=MEAN1
          CB(4)=MEAN2
C ASSIGN POLYGON TO CLASS
          DO 3 N=1,NP
            IF (STAT(N) .LE. MEAN1) THEN
              ICLASS(N)=1
            ELSE IF (STAT(N) .LE. MEAN) THEN
              ICLASS(N)=2
            ELSE IF (STAT(N) .LE. MEAN2) THEN
              ICLASS(N)=3
            ELSE
              ICLASS(N)=4
            END IF
 3        CONTINUE
C
          RETURN
          END
C
```

```
      SUBROUTINE QUANTIL(NB,STAT,NP,ICLASS,CB)
C ***********************************************************************
C   SUBROUTINE TO CALCULATE CLASS LIMITS USING QUANTILES
C ***********************************************************************
      INTEGER*2 PART,NP,NB,IBGN,ISTP,POLYORD(100),ICLASS(100)
      INTEGER*2 ITEMP,IRANK(100),NM
      REAL*4 STAT(100),CB(20),TEMP,STAT1(100)
      NM=0
C
C INITIALIZE RANK ARRAY
      DO 9 K=1,NP
        IRANK(K)=K
        STAT1(K)=STAT(K)
C  COUNT NUMBER OF POLYGONS WITH MISSING DATA
        IF (STAT(K) .EQ. -9999.) NM=NM+1
  9   CONTINUE
C
C PLACE POLYGONS IN ASCENDING ORDER ACCORDING TO STAT DATA
      DO 1 I=1,NP
        J=I
        DO 2 I2=I+1,NP
          IF (STAT1(I2) .LT. STAT1(J)) J=I2
 2      CONTINUE
        TEMP=STAT1(I)
        STAT1(I)=STAT1(J)
        STAT1(J)=TEMP
        ITEMP=IRANK(I)
        IRANK(I)=IRANK(J)
        IRANK(J)=ITEMP
1     CONTINUE
C
C SET DIVISION OF NUMBER OF OBSERVATIONS
      PART=(NP-NM)/NB
      IBGN=1+NM
      ISTP=PART+NM
C ASSIGN POLYGONS TO CLASS
      DO 3 J=1,NB
        DO 4 K=IBGN,ISTP
          ICLASS(IRANK(K))=J
 4      CONTINUE
        CB(J+1) =  STAT(IRANK(ISTP))
        IBGN=ISTP+1
        ISTP=ISTP+PART
        IF (J .EQ. NB-1) ISTP=NP
 3    CONTINUE
C
      RETURN
      END
C
C
```

```
      SUBROUTINE LIMITS(STAT,ICLASS,NP,NB,CB)
C***********************************************************************
C     SUBROUTINE TO ENTER CLASS LIMITS SEPARATELY
C***********************************************************************
      INTEGER*2 ICLASS(100),NP,NB
      REAL*4 STAT(100),CB(20)
      CHARACTER*40 LSTR1
C
      DATA LSTR1/'ENTER CLASS LIMITS AT TERMINAL'/
C
      CALL CLRMENU
      CALL MOVABS(-200,-218)
      CALL TEXT1(LSTR1)
      CALL EMPTYB
C
C READ IN CLASS LIMITS
      K=1
      DO 1 I=1,NB
        WRITE('UT',*) 'ENTER CLASS LIMITS FOR CLASS',I,':'
        READ('UT',*)   CB(K),CB(K+1)
C110    FORMAT(F5.1,F5.1)
        K=K+2
   1  CONTINUE
C
C ASSIGN POLYGON TO CLASS
      DO 2 I=1,NP
        DO 3 I2=2,NB*2,2
          IF (STAT(I) .GE. CB(I2-1) .AND. STAT(I) .LE. CB(I2)) THEN
            ICLASS(I)=I2/2
            GOTO 2
          END IF
  3     CONTINUE
  2   CONTINUE
C
C RENUMBER CLASS BOUNDARIES FOR AUTO-LEGEND
      J=1
      DO 4 I=3,NB+2
        CB(I)=CB(I+J)
        J=J+1
   4  CONTINUE
C
      RETURN
      END
C
C
      SUBROUTINE COLRVAL(IC,CC,R,G,B)
C*********************************************************************
C      SUBROUTINE TO OBTAIN R, G, AND B VALUES FOR AUTOMATED COLOR FILL
C *********************************************************************
      INTEGER*2 BL(3,8),GRN(3,8),RD(3,8),YLW(3,8),IC,CC,R,G,B
      INTEGER*2 CYN(3,8),MGT(3,8)
C
C INITIALIZE COLOR ARRAYS
      DATA RD/255,230,230,255,202,202,243,178,178,243,141,141,230,111
     *,111,217,0,0,202,0,0,168,0,0/
      DATA GRN/243,255,243,217,255,217,186,255,186,152,230,152,0,217,
     *0,0,186,0,0,148,0,90,123,90/
```

```
          DATA BL/230,230,255,202,202,255,168,168,255,123,123,255,0,0,243
     *,0,0,217,0,0,186,0,0,148/
          DATA YLW/255,255,217,255,255,168,255,255,123,255,255, 0,243,243
     *,0,230,230,81,217,217,77,202,202,71/
          DATA CYN/217,255,255,178,243,243,152,230,230,77,217,217,117,202
     *,202,0,186,186,98,168,168,0,148,148/
          DATA MGT/255,230,255,255,186,255,243,141,243,230,81,230,217,77,
     *217,202,98,202,168,0,168,148,52,148/
C
C
          GOTO(10,20,30,40,50,60)CC
 10       R=RD(1,IC)
          G=RD(2,IC)
          B=RD(3,IC)
          GOTO 99
C
 20       R=GRN(1,IC)
          G=GRN(2,IC)
          B=GRN(3,IC)
          GOTO 99
C
 30       R=BL(1,IC)
          G=BL(2,IC)
          B=BL(3,IC)
          GOTO 99
C
 40       R=YLW(1,IC)
          G=YLW(2,IC)
          B=YLW(3,IC)
          GOTO 99
C
 50       R=CYN(1,IC)
          G=CYN(2,IC)
          B=CYN(3,IC)
          GOTO 99
C
 60       R=MGT(1,IC)
          G=MGT(2,IC)
          B=MGT(3,IC)
C
 99       RETURN
          END
C
C
          SUBROUTINE AUTOLGND(NB,PX,PY,MX,MIX,MY,MIY,PRX,PRY,LLX,LLY,
     *LGND,HFLAG)
C*********************************************************************
C     SUBROUTINE FOR AUTOMATED LEGEND GENERATOR
C *********************************************************************
          INTEGER*2 IBH,IBW,IBSP,IDIR,NB,PX,PY,MX,MIX,MY,MIY,IX,IY
          INTEGER*2 PRX,PRY,LLX,LLY,LGND(2,15),NMAXX,NMAXY,NYDIS
          INTEGER*2 NYCNTR,IY1,LMARG,RMARG,LGNDLEN,IX2,IY2,LLX1,LLY1
          INTEGER*2 NXCNTR,NROW,HFLAG
          REAL*4 ODD
C
          HFLAG=0
C  SET LEGEND BOX PARAMETERS
```

```
          IBH=16
          IBW=16
          IBSP=8
          IDIR=2
C
C
C DETERMINE SCALE OF MAP
          NMAXX=(MX-MIX)/PRX+PX
          NMAXY=(MY-MIY)/PRY+PY
C
C CALCULATE LEFT AND RIGHT MARGIN OF MAP
          LMARG= IABS(-255 - PX)
          RMARG= 255 - NMAXX
C
C VERTICAL OR HORIZONTAL LEGEND
          IF (LMARG .GT. 100 .OR. RMARG .GT. 100) THEN
            LGNDLEN= NB*IBH + (NB-1)*IBSP
            NYDIS = NMAXY - PY
            NYCNTR = NYDIS/2 + PY
            LLY = NYCNTR - LGNDLEN/2
C PLACE LEGEND ON SIDE WITH LARGEST LEGEND
          IF (LMARG .GT. RMARG) THEN
            LLX = PX - 30
          ELSE
            LLX = NMAXX + 20
          END IF
C
C DRAW VERTICAL LEGEND BOXES
          IX=LLX
          IY=LLY
          CALL MOVABS(IX,IY)
          CALL RECREL(IBW,IBH)
          LGND(1,1)=IX + IBW/2
          LGND(2,1)=IY + IBH/2
            IF (IDIR .EQ. 1) THEN
              DO 3 J=1,NB-1
              IX2=IX + IBW*J + IBSP*J
                CALL MOVABS(IX2,IY)
                CALL RECREL(IBW,IBH)
                LGND(1,J+1)=IX2 + IBW/2
                LGND(2,J+1)=IY + IBH/2
  3           CONTINUE
            ELSE
              DO 4 J=1,NB-1
                IY2=IY + IBH*J + IBSP*J
                CALL MOVABS(IX,IY2)
                CALL RECREL(IBW,IBH)
                LGND(1,J+1)=IX + IBW/2
                LGND(2,J+1)=IY2 + IBH/2
  4           CONTINUE
            END IF
          ELSE
            HFLAG=1
            NXCNTR=(NMAXX-PX)/2 + PX
            ODD=FLOAT(NB)/2.
            NROW=NB/2
            IF (NROW .NE. ODD) THEN
```

```
            NROW=NROW+1
          END IF
          LLX1=NXCNTR - 50 - (NROW*20 + (NROW-2)*20)
          LLY1=PY-45
          LLX=LLX1
          LLY=LLY1
          N=1
          DO 5 J=1,NROW
            DO 5 K=1,2
              CALL MOVABS(LLX,LLY)
              CALL RECREL(I3W,I3H)
C CALCULATE CENTER POINT FOR LEGEND BOX
              LGND(1,N)=LLX + I3W/2
              LGND(2,N)=LLY + I3H/2
              LLY=LLY-25
              IF(N .EQ. N3) GOTO 101
              N=N+1
  5         CONTINUE
C REPOSITION CO-ORDINATES FOR NEXT ROW
            LLX=LLX+55
            LLY=LLY+45
  5       CONTINUE
 101      LLX=LLX1
          LLY=LLY1
        END IF
C
        CALL EMPTYS
        RETURN
        END
C
C
        SUBROUTINE BORDER(BCNTRX,BCNTRY)
C ******************************************************************
C     SUBROUTINE TO DRAW BORDER AROUND MAP
C ******************************************************************
        CHARACTER*1 LPOS
        CHARACTER*60 BSTR1,BSTR2,BSTR3,BSTR4
        INTEGER*2 UBX,UBY,LBX,LBY,IX,IY,IBUT,BCNTRX,BCNTRY
C
        DATA LPOS/'.'/
        DATA BSTR1/'MOVE CROSSHAIR TO LOWER LEFTHAND CORNER OF BORDER'/
        DATA BSTR2/'AND PRESS CURSOR BUTTON'/
        DATA BSTR3/'MOVE CROSSHAIR TO UPPER RIGHTHAND CORNER OF BORDER'
     */
        DATA BSTR4/'PRESS E TO END, A TO START OVER'/
C
 101    CALL CLRMENU
        CALL MOVABS(-200,-210)
        CALL TEXT1(BSTR1)
        CALL MOVABS(-200,-220)
        CALL TEXT1(BSTR2)
C
C ENTER LOWER LEFT CORNER OF BORDER
        CALL READBU(1,1,IBUT,IX,IY)
        LBX=IX
        LBY=IY
        CALL MOVABS(LBX,LBY)
        IF (LBY .LT. -195) LBY=-195
        CALL TEXT1(LPOS)
C
        CALL MOVABS(-200,-230)
        CALL TEXT1(BSTR3)
```

```
          CALL TEXT1(CLSTR3)
C
          CALL READBU(1,1,IBUT,IX,IY)
          UBX=IX
          UBY=IY
C
C DRAW RECTANGLE AROUND AREA TO BE CLEARED
          CALL MOVABS(LBX,LBY)
          CALL RECTAN(UBX,UBY)
C
          CALL CLRMENU
          CALL MOVABS(-200,-210)
          CALL TEXT1(CLSTR4)
          CALL MOVABS(-200,-220)
          CALL TEXT1(CLSTR5)
          CALL MOVABS(-200,-230)
          CALL TEXT1(CLSTR6)
C
          CALL READBU(1,1,IBUT,IX,IY)
C
          IF (IBUT .EQ. 15 ) THEN
            CALL WINDOW(LBX,LBY,UBX,UBY)
            CALL VAL8(255)
            CALL CLEAR
            CALL VAL8(0)
            CALL WINDOW(-256,-256,255,255)
          ELSE
            CALL VAL8(255)
            CALL MOVABS(LBX,LBY)
            CALL RECTAN(UBX,UBY)
            CALL VAL8(0)
            IF (IBUT .EQ. 10) GOTO 101
          END IF
C
          CALL CLRMENU
          RETURN
          END
C
C
          SUBROUTINE CENMAP(BCNTRX,BCNTRY,PX,PY,UBX,UBY,IX1,IY1,HFLG,ACF
     *LAG)
C***********************************************************************
C     SUBROUTINE TO CENTER MAP ON SCREEN
C***********************************************************************
          INTEGER*2 BCNTRX,BCNTRY,PX,PY,UBX,UBY,IX1,IY1,HFLG,ACFLAG
          INTEGER*2 IX,IY,IBUT,ICX,ICY
          CHARACTER*60 CMSTR1,CMSTR2
C
          DATA CMSTR1/'PRESS C TO CENTER MAP ON SCREEN AND EXIT PROGRAM'/
          DATA CMSTR2/'PRESS E TO ONLY EXIT PROGRAM (LEAVE MAP ALONE)'/
          ICX=0
          ICY=0
C
          CALL CLRMENU
          CALL MOVABS(-200,-210)
          CALL TEXT1(CMSTR1)
          CALL MOVABS(-200,-220)
```

```
C
C ENTER UPPER RIGHT CORNER OF BORDER
         CALL READBU(1,1,IBUT,IX,IY)
         UBX=IX
         UBY=IY
C
C DRAW BORDER ON SCREEN
         CALL MOVABS(LBX,LBY)
         CALL RECTAN(UBX,UBY)
C
         CALL MOVABS(-200,-250)
         CALL TEXT1(BSTR4)
C
C READ INSTRUCTION
         CALL READBU(1,1,IBUT,IX,IY)
         IF (IBUT .EQ. 10) THEN
         CALL VALS(255)
         CALL MOVABS(LBX,LBY)
         CALL RECTAN(UBX,UBY)
         CALL VALS(0)
         GO TO 101
         END IF
C
C CALCULATE CENTER OF AREA IN BORDER
         BCNTRX=IABS(UBX-LBX)/2 + LBX
         BCNTRY=IABS(UBY-LBY)/2 + LBY
         CALL CLRMENU
         RETURN
         END
C
C
         SUBROUTINE CLRPRTN
C **********************************************************************
C      SUBROUTINE TO CLEAR A PORTION OF THE SCREEN
C **********************************************************************
         INTEGER*2 IBUT,IX,IY,LBX,LBY,UBX,UBY
         CHARACTER*60 CLSTR1,CLSTR2,CLSTR3,CLSTR4,CLSTR5,CLSTR6
C
         DATA CLSTR1/'MOVE CROSSHAIR TO LOWER LEFTHAND CORNER OF AREA'/
         DATA CLSTR2/'MOVE CROSSHAIR TO UPPER RIGHTHAND CORNER OF AREA'/
         DATA CLSTR3/'PRESS CURSOR BUTTON'/
         DATA CLSTR4/'PRESS F TO CLEAR AREA OUTLINED'/
         DATA CLSTR5/'PRESS A TO START OVER'/
         DATA CLSTR6/'PRESS E TO ABORT AND EXIT'/
C
 101     CALL CLRMENU
         CALL MOVABS(-200,-210)
         CALL TEXT1(CLSTR1)
         CALL MOVABS(-200,-220)
         CALL TEXT1(CLSTR3)
C
C ENTER LOWER LEFT CORNER OF RECTANGLE TO BE CLEARED
         CALL READBU(1,1,IBUT,IX,IY)
         LBX=IX
         LBY=IY
C
         CALL MOVABS(-200,-230)
         CALL TEXT1(CLSTR2)
         CALL MOVABS(-200,-240)
```

```
          CALL TEXT1(CMSTR2)
C
C READ COMMAND AND DETERMINE IF MAP IS TO BE CENTERED
          CALL READBU(1,1,IBUT,IX,IY)
          IF (IBUT .EQ. 12) THEN
            IF (BCNTRX .NE. -250  .AND. BCNTRY .NE. -250) THEN
              CALL SCRORG(BCNTRX,BCNTRY)
                GOTO 99
            END IF
C
C DETERMINE IF ASSISTED CLASSIFICATION WAS USED
            IF (ACFLAG .EQ. 1) THEN
              IF (HFLG .EQ. 1) THEN
                ICX=IABS(UBX-PX)/2 + PX
                ICY=(IABS(UBY-PY)+63)/2 + (PY-63)
              ELSE
                ICX=(IABS(UBX-PX)+100)/2 + (PX-100)
                ICY=IABS(UBY-PY)/2 + PY
              END IF
            ELSE
C MANUAL LEGEND WAS USED
              IF (IY1 .LT. PY) THEN
                ICY=IABS(UBY-IY1)/2 + IY1
              ELSE
                ICY=IABS(UBY-PY)/2 + PY
              END IF
              IF (IX1 .LT. PX) THEN
                ICX=IABS(UBX-IX1)/2 + IX1
              ELSE
                ICX=IABS(UBX-PX)/2 + PX
              END IF
            END IF
            CALL SCRORG(ICX,ICY)
          END IF
 99       RETURN
          END
C
C
          SUBROUTINE CLRMENU
C **************************************************************************
C              SUBROUTINE TO ERASE MENU FROM SCREEN
C **************************************************************************
          CALL WINDOW(-256,-256,255,-200)
          CALL VAL8(255)
          CALL CLEAR
          CALL VAL8(0)
          CALL WINDOW(-256,-256,255,255)
          RETURN
          END
```