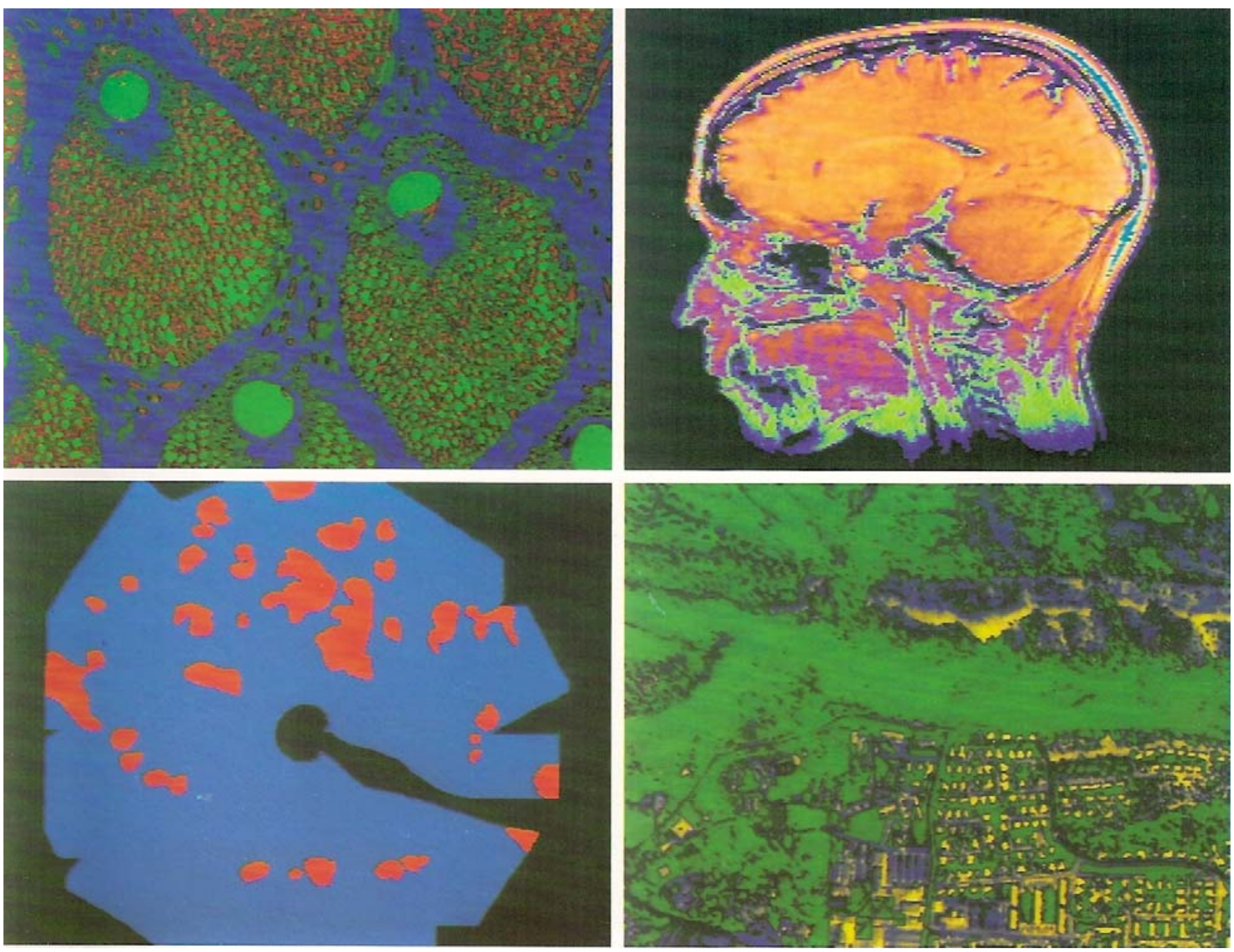
LA-11732-M
Manual

A Manual for Microcomputer Image Analysis



Los Alamos

Los Alamos National Laboratory is operated by the University of California for the United States Department of Energy under contract W-7405-ENG-36.

This work was supported by National Science Foundation Grant BSR86-14769

Cover: *These four pseudocolor images demonstrate that IMAGE can be useful for image analysis at different spatial scales, including microscopic, organismic, and landscape levels. In the upper left is an image of a transverse section of the stem of the palm Iriartea gigantea. In the upper right is a magnetic-resonance-produced image of a brain. The lower-left image is a map of gopher mound distribution in a serpentine grassland community. The lower-right image is an aerial view of a portion of the town of Los Alamos, New Mexico.*

LA-11732-M
Manual

UC-000
Issued: December 1989

A Manual for Microcomputer Image Analysis

*Paul M. Rich**
Douglas M. Ranken
John S. George

Department* of *Biological Sciences, Stanford University, Stanford,* **CA *94305.*

Los Alamos Los Alamos National Laboratory
Los Alamos, New Mexico 87545

SOFTWARE COPYRIGHT

TRADEMARKS

CONTENTS

PREFACE

Recent advances in microcomputer and video technology present us with the opportunity to do powerful image processing and analysis with microcomputers. Much in the same way in which word processing has become commonplace, image processing and analysis can be made widely available at modest cost. For the additional cost of a video camera, a digitizer/display adapter, and a video monitor, a microcomputer can be transformed into a powerful image processing work station. Realization of the full potential of microcomputer image processing and analysis is now primarily limited by the availability of software, especially software suitable for scientific applications.

I first became involved with microcomputer image processing and analysis while I was a graduate student in the Department of Biology at Harvard University. In the course of my studies I became interested in the complex problem of how to characterize the light environment of young trees as they grow in the understory of tropical rain forests. In 1983 I began developing the program *CANOPY*(c) for analysis of the geometry of plant canopy openings through which light can penetrate (Rich In Press). *CANOPY* uses video input of photographic negatives to a microcomputer image analysis system. While developing *CANOPY*, I became intrigued by the prospects of providing scientifically oriented microcomputer image analysis capabilities.

I began working as a research assistant to Dr. John S. George in the Life Sciences Division at Los Alamos National Laboratory, and then worked as a postdoctoral fellow in collaboration with Thomas E. Hakonson and Fairley J. Barnes in the Environmental Science Group, where we developed a broad range of microcomputer image analysis capabilities for various video digitizers and display adapters. Later, as a postdoctoral fellow with Harold A. Mooney at Stanford University, I continued development of microcomputer image analysis capabilities. Throughout our effort, John George played a central role in overall program design and conceptualization of unique and effective algorithms. Douglas M. Ranken joined our effort as a research assistant and has developed many of the more sophisticated software capabilities, including the flood and edge detection routines. Douglas Ranken and I originally wrote *IMAGE* for two specific tasks: 1) measurement of lengths of roots and 2) measurement of areas in photographs, in particular areas covered by vegetation and other land features. We decided to expand *IMAGE* into a single, general-use program that brings together many of our microcomputer image analysis capabilities, and that can be applied to a wide range of scientific problems at many different spatial scales.

It is my hope that this manual will serve as much more than a technical reference to the program IMAGE. I hope that it will inspire further development of microcomputer image processing and analysis as research tools in science.

Paul M. Rich, 26 June 1989,
Department of Biological Sciences,
Stanford University

A MANUAL FOR MICROCOMPUTER IMAGE ANALYSIS

by

Paul M. Rich, Douglas M. Ranken, and John S. George

ABSTRACT

This manual is intended to serve three basic purposes: 1) as a primer in microcomputer image analysis theory and techniques, 2) as a guide to the use of *IMAGE*(c), a public domain microcomputer program for image analysis, and 3) as a stimulus to encourage programmers to develop microcomputer software suited for scientific use. Topics discussed include the principles of image processing and analysis, use of standard video for input and display, spatial measurement techniques, and the future of microcomputer image analysis. A complete reference guide and listing of commands for *IMAGE* is provided. *IMAGE* includes capabilities for digitization, input and output of images, hardware display lookup table control, editing, edge detection, histogram calculation, measurement along lines and curves, measurement of areas, examination of intensity values, output of analytical results, conversion between raster and vector formats, and region movement and rescaling. The control structure of *IMAGE* emphasizes efficiency, precision of measurement, and scientific utility.

CHAPTER I. INTRODUCTION

BACKGROUND

Digital image processing technology has its roots in the unmanned space exploration effort of the National Aeronautics and Space Administration (NASA) during the 1960s. Until recently, image processing required expensive facilities based on mainframe computers. With the advent of microcomputers, solid-state video cameras, and video digitizers, and with ongoing advances in processing speed, graphics display, and data storage, it is now possible to do sophisticated image processing at modest cost. We are faced with a major challenge to provide image processing and analysis software for scientific applications. Much of the initial software that has been developed for microcomputer image processing has been oriented to business and graphics applications, where the primary goal is to produce an image for illustration. Scientifically oriented microcomputer image analysis software that has become available is often limited in capabilities, ineffective for quantitative measurement, and awkward to use. Scientific applications require precise quantification; however, specific needs change depending upon the problems being addressed. The challenge is to provide powerful processing and analysis tools with sufficient flexibility to allow custom application to a particular problem or set of problems.

The program *IMAGE*(c) provides a general set of image processing and analysis capabilities for making one- and two-dimensional spatial measurements. *IMAGE* typically uses standard RS-170 composite video for input and standard RGB video for image output and display. *IMAGE* serves both as a program that is immediately useful for scientific applications and as an example of how image analysis software can be designed for scientific needs. As used here, image processing refers to the set of capabilities that allow input, display, manipulation, and output of digital images. Image analysis refers to the set of capabilities that allow extraction of meaningful information from digital images. The difference between processing and analysis is not distinct, but rather the two are interdependent--image processing provides the basic tools for working with images and image analysis provides specialized tools for image interpretation relevant to a particular problem. Ideally, processing and analysis are integrated so that a user can effectively accomplish a specific task.

OVERVIEW

This manual serves three needs. First, it presents fundamentals of image processing and analysis for scientific applications, in particular using video for image input. Second, it is a comprehensive reference guide for *IMAGE*, a general-use microcomputer image analysis program. Finally, it provides a discussion of programming considerations and the future of microcomputer image analysis. Thus, this manual is directed toward scientists getting started with image analysis, users of the program *IMAGE*, and programmers who face the challenge of designing scientific image analysis capabilities.

Chapter II presents the fundamentals of image analysis, using specific features of *IMAGE* for examples. Chapter III provides a discussion of programming considerations, including compromises involved in different approaches to program control. Chapter IV examines the future of microcomputer image analysis, with consideration of both hardware and software advances. Chapter V provides a basic guide to the operation of *IMAGE*. Chapter VI provides a reference guide to *IMAGE*, with an alphabetical listing of all commands. A bibliography lists basic references for image processing and analysis. Appendix A is a glossary that defines fundamental terminology. Appendices B through E provide a listing of menus screens, hardware specifications, details of output file formats, and a listing of files and utility programs.

USE OF VIDEO FOR IMAGE ANALYSIS

The use of standard RS-170 video for input, display, and storage of images has many advantages, including sufficient resolution to be useful for many scientific applications, well-developed technology for rapid digitization, widespread availability and compatibility of hardware, and modest cost. Because standard video is so widely used, cameras, display monitors, storage devices, and output devices are readily available at reasonable expense, with a broad range of choices for specific features. Video cameras range from inexpensive tube cameras that are suitable for many kinds of measurements, to top-of-the line tube and solid-state cameras that offer minimal geometric distortion, high sensitivity, flat response across the field of view, and excellent signal-to-noise ratios. Video digitizers range from pixel digitizers that can capture an image in several seconds to framegrabbers that can continuously capture and display images at the rate of 30 images per second. Storage devices include video tape recorders that use low-cost magnetic tapes and laser storage media that offer storage with little signal degradation. Output devices include photographic devices, video printers, dot matrix printers, and laser printers. Standard video has significant limitations, including restricted spatial resolution and problems with signal-to-noise ratios. Even with these limitations, standard video is effective for many scientific applications.

IMAGE ANALYSIS FOR SCIENTIFIC APPLICATIONS

Image analysis has applications in every scientific field from microbiology to ecology to astrophysics. *IMAGE* has already been applied to a wide variety of scientific problems, including measurement of root growth in plants and mapping the spatial distribution and dynamics of localized ecological disturbance in plant communities. Applications of *IMAGE* include measurements from images at any spatial scale, from microscope images to satellite remote sensing data. *IMAGE* can also be used for scientific illustration. Use of video for input and storage allows rapid, convenient, and flexible input. For instance, images can be analyzed directly from a microscope equipped with a video camera. Though *IMAGE* generally uses video input, it is also possible to analyze images from other sources, including scanners, magnetic resonance imaging, and existing satellite remote sensing digital data sets.

REMOTE SENSING DATA DISPLAY AND ANALYSIS

For remote sensing applications, sensors are often capable of detecting a series of different spectral ranges. A spectral channel contains image intensity information for a particular spectral range. Often it is useful to display and analyze individual channels, compress multispectral data into a composite image display format, or calculate indices that combine spectral information. For example, vegetation features can be isolated and studied using spectral indices calculated using ratios or differences between red and near-infrared channels. *IMAGE* provides many capabilities that provide for low-cost analysis of remote sensing data. Individual spectral channels can be displayed and analyzed one at a time with *IMAGE*, using up to 8 bits of data per channel. Similar to the way true color can be encoded, digital image data from multiple spectral channels can be compressed and encoded as a 7- or 8-bit composite image for display and analysis (see Chapter IV section on True Color Imaging). Also, indices between different spectral channels can be calculated for display and analysis using *IMAGE***.**

SOFTWARE CAPABILITIES

IMAGE includes a wide range of image analysis capabilities. Processing capabilities include digitization, image and data file input and output, and control of hardware lookup tables (LUTs). Analysis capabilities include image editing, edge detection, calculating histograms, measurement along lines and curves, measurement of areas, and region movement and rescaling. Table I gives a tabular listing of capabilities that are explained in detail in chapters II, V, and VI. All routines are written in the "C" programming language. *IMAGE* uses a user-friendly menu structure, with single keystrokes for commands and cursor control with the keyboard and a mouse or other locator device. To meet the needs for quantitative scientific analyses, *IMAGE* provides precise cursor control on a pixel-by-pixel basis, with access to pixel X,Y coordinates and data values. Results of analyses can be displayed on the computer control screen and saved to ASCII files.

HARDWARE

IMAGE is currently supported on IBM-compatible microcomputers equipped with an Imaging Technology PC Vision, PC Vision Plus, FG-100, or FG-100 1024 video digitizer/display adapter. The basic requirements are an IBM-compatible microcomputer with 640K RAM and a mathematics coprocessor, a mouse, a video digitizer/display adapter with hardware lookup tables, a standard RGB analog monitor, and an RS-170 black and white video camera. See Appendix C for detailed hardware specifications.

Table I. Capabilities of *IMAGE*.

Capability	Specific Routines
Digitization	*Digitize*
Image File Input/Output	*Archive (Load, Save, DOS shell)*
Display LUT Control	*Continuous Tone, Threshold, Slice, and Pseudocolor* (positive/ negative)
LUT Input/Output/Editing	*LUT Utilities*
Classification/Edge Detection	*Threshold, Flood, Edge,* editing routines
Histogram	*Histogram*
Editing	*Line, Trace, Rectangle* (filled and unfilled), *Circle* (filled and unfilled), *Flood*
Line Measurement	*Line*
Curve Measurement	*Trace*
Area Measurement	*Flood, Edge, Trace*
Unit Calibration	*Unit*
Raster to Vector Conversion	*Edge, Trace*
Intensity Data Output	*Trace*
Region Movement and Rescaling	*Mosaic*
Scalar/Whole Image Operations	*Operations (bit shift, linear transform, subtraction, addition, multiplication, division, AND, OR, XOR)*

CHAPTER II. FUNDAMENTALS OF IMAGE ANALYSIS USING *IMAGE*

THE STEPS OF IMAGE ANALYSIS

Even with the broad range of spatial scales and specific requirements of scientific applications, the basic steps of image analysis are the same. These steps are image acquisition and digitization, image rectification and enhancement, image classification, image analysis and interpretation, and data output (Fig. 1). Acquisition and digitization provide the images for digital analysis. Rectification and enhancement improve the accessibility of data, for instance, by increasing contrast, and make corrections for geometric and radiometric distortion and noise. Classification assigns regions of an image to useful categories, isolating objects or features of interest. Interpretation and analysis involve quantitative measurement of the classified image in the context of the problem being addressed. Data output involves tabulating analyses in a useful format. *IMAGE* provides capabilities for all of these steps in a control structure that facilitates proceeding from one step to the next.

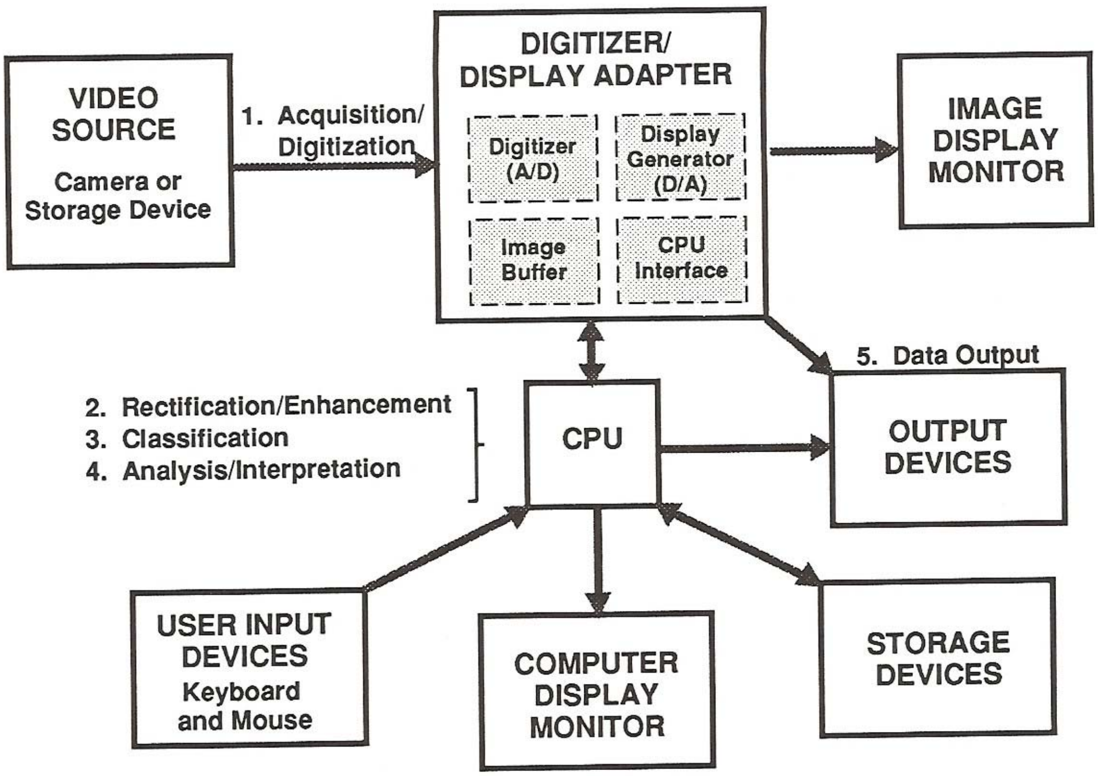


Fig. 1

Conceptual representation of image analysis using video for input. The basic steps are 1) image acquisition and digitization, 2) image rectification and enhancement, 3) image classification, 4) image analysis and interpretation, and 5) data output. Images are input as an analog RS-170 video signal to a digitizer/display adapter that is controlled by a microcomputer and has capabilities for digitization, storage of digital images in an image buffer, and conversion of digital data back to analog for display on an image monitor. The central processing unit (CPU) controls the digitizer/display adapter, transfers information to and from the digitizer/display adapter, allows interactive user input, stores information, displays information on a computer monitor, and outputs analyses and images to printers or photographic devices. Output of images is also possible from the digitizer/display adapter to photographic devices.

THE IMAGE ANALYSIS SYSTEM

The fundamental components of the microcomputer image analysis system used by *IMAGE* are 1) an input device, a black and white video camera, 2) a digitizer/display adapter, a framegrabber that converts a video signal into a digital image, 3) an image display monitor, 4) a microcomputer, and 5) a mouse or other locator device (Fig. 2). *IMAGE* uses a dual screen system, one monitor for image display and one for display of menus, control information, and graphics such as histograms.

FUNDAMENTALS OF VIDEO

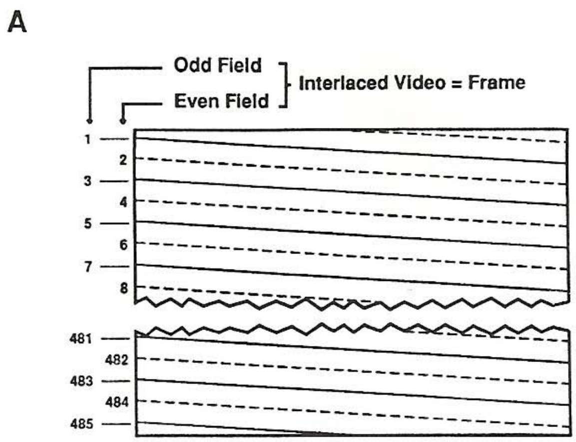
IMAGE typically uses standard RS-170 black and white composite video for image input and standard analog RGB video for color display. A basic knowledge of video technology is useful for understanding the functioning of *IMAGE*.

RS-170 Composite Video

RS-170 is the standard black and white video format used in the United States (Fig. 3). A frame, or individual image, is produced each 1/30 second (30 Hz). The RS-170 standard specifies 525 scan lines of which 40 are used for retrace by the display electron gun and 485 are available for display. When displayed, a scan line is literally a series of locations on a display screen that are lit up as an electron gun sweeps from left to right. An image is produced by sequentially displaying varying intensities along each of the 485 scan lines. The video signal encodes light intensity as a function of position. Position is represented by time and light intensity by voltage, which generally varies

Fig. 2.

Photograph of image analysis system. The basic components are A) a microcomputer (IBM compatible) equipped with a digitizer/display adapter, B) a computer display monitor, C) a keyboard and locator device (mouse), D) an image display monitor, and E) a video camera. Above the image monitor is a slide viewer to allow comparison of photographs with digital images. The video camera is mounted on a copystand. Below the video camera is a backlit film holder for input of slides or negatives. A tape storage device is between the computer and the image display monitor.



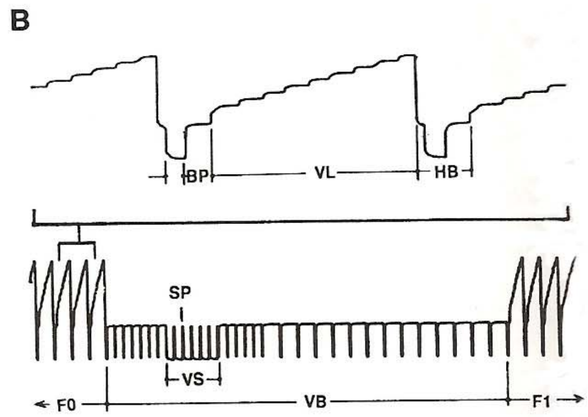


Fig. 3.
A) Schematic of an interlaced video display. Each image is composed of 485 scan lines. Odd scan lines (solid lines) and even scan lines (dashed lines) are alternately displayed to form an interlaced image. A set of odd or even scan lines is called a field and together the two fields form a frame. Fields are refreshed each 1/60 second, forming frames each 1/30 second. B) Oscilloscope trace of an RS-170 composite video signal from a test pattern consisting of nine vertical bands of varying intensity. HS = horizontal sync, BP = back porch, VL = video line (analog data), HB = horizontal blanking, SP = serration pulse, VS = vertical sync, VB = vertical blanching, F0 = field 0, and F1 = field 1.

between 0 and 1.0 V. Also included in the video signal is horizontal and vertical synchronization information. *

To minimize flicker apparent at 30 Hz, the video signal is "interlaced", so that for each frame, first odd and then even scan lines are displayed. Each frame is composed of two fields, one consisting of 242.5 odd scan lines and the other 242.5 even scan lines, with a field produced each 1/60 second (60 Hz). The 60-Hz field rate was chosen to avoid interference from ac current fields. Such interference is sometimes apparent as a slowly shifting horizontal band on cameras which are not phase-locked to ac power. When the analog video signal is converted to a digital image, the digital image can be treated as a simple two-dimensional array of intensity values, so interlace effects need not be considered except for cases where blur may result from motion or other dynamic processes while an image is acquired.

The horizontal resolution of an analog video image is limited by the signal quality, as determined by all hardware--the video camera, storage medium (if used), intervening cables and circuitry, and display technology. Black and white cameras and CRT display tubes can resolve detail approaching or exceeding 1000 video lines. This

* Between video lines, during the time required to shift beam targeting magnets and electronics from one side of a display tube to the other, the video signal drops to what becomes defined as the zero or black level of what is otherwise an ac coupled signal. In composite video, a negative horizontal synchronization pulse is generated during a portion of this blanking interval. Between video fields, a longer negative pulse identifies the vertical synchronization interval. These pulses collectively specify the timing of the video signal. According to the National Television Standards Committee (NTSC) standards, the vertical synchronization pulse should be interrupted by zero voltage pulses (serrated), although this information is not required by most consumer or professional video equipment.

empirically defined quantity is the number of pairs of black and white parallel lines that could be counted across the display monitor at the limit of detection by a human observer*. For the purposes of digital video, it is more useful to define the number of discrete picture elements or pixels that are resolved by the system. Typically, the horizontal resolution is 512 pixels, a convenient number for programming. Horizontal resolution may be limited by the number of photosensitive physical sites in a solid-state sensor; however, some specialized black and white RS-170 (charged coupled device (CCD) cameras exceed 750 pixels for horizontal resolution and specialized linear or area sensors can resolve 1000 to 4000 horizontal pixels.

The vertical resolution of video is limited to 485 pixels, as determined by the number of scan lines. The RS-170 standard specifies the aspect ratio (ratio of vertical/horizontal dimensions) of the video display as 3:4. A typical digital image produced by video digitization would have a resolution of 512 (horizontal) X 480 (vertical) pixel resolution and would have individual pixels with a 5:6 aspect ratio. Displays reflecting a lower resolution, (e.g., 256 X 256, 128 X 128, or 64 X 64) may be useful for some specialized sensors or computer-generated images, or when it is desirable to capture a sequence of images. A resolution of 640 X 480 produces square pixels, which can simplify image spatial analysis and editing operations**.

The intensity information available in the video signal is limited by the dynamic range (the range of light intensities that can be detected) and the signal-to-noise ratio. The dynamic range is generally determined by the video camera, while the signal quality is a function of all hardware subsystems. Humans can only distinguish between 16 and 64 intensity levels. Older, inexpensive black and white video cameras were designed to produce perceptually acceptable image quality and could resolve as few as 10 intensity levels; though high-quality cameras can exceed the range of human vision. Typically 1 byte of intensity information (256 levels) is digitized per pixel, though for most applications 5-7 bits (32-128 levels) are adequate. Until recently, 10+ bit flash digitizers were not available to operate at video rates. However the wideband frequency response required for standard resolution video (10-20 MHz) also introduces additional noise, so additional bits may not be significant. Some scientific cameras can resolve 12-14 bits of intensity information, usually by employing slow-scan video acquisition.

Color Video

Colors are produced on a CRT display monitor by mixing different intensities of red, green, and blue light. For example, yellow is produced by additive mixing of equal intensities of green and red with no blue; shades of gray are produced by mixing equal intensities of red, green, and blue. The inside of the display tube is coated with phosphors that produce photons with a particular wavelength distribution when excited by the electron beam. A traditional tube has triangular clusters of red, green, and blue phosphor dots, while in a Trinitron tube the phosphors are arranged in parallel vertical lines. A shadow mask or vertical grating prevents the electron beam from exciting inappropriate regions of the tube surface. The physical dimensions of the components have traditionally limited the resolution available with color displays; however, improved manufacturing techniques and larger display formats now permit display hardware with resolution exceeding 1000 horizontal pixels. A single electron gun may be used to sequentially excite the red, green and blue phosphors; however, in some tube designs, three electron guns are used simultaneously.

For digital acquisition, storage, and reproduction of color, the light intensity information specific for red, green, and blue may be encoded as separate RGB video signals (Fig. 4). In such RGB systems, each color channel may resemble an RS-170 black and white video signal. In the RGB display system used for *IMAGE*, the display resolution and timing is compatible with RS-170, though this may not be the case with higher resolution RGB systems. Although video intensity information and blanking intervals are present in the red and blue signals, the synchronization pulses typically are not. The display system can be configured to produce composite synchronization on the green channel (which can be used to drive a black and white display). Alternatively, a fourth signal containing only the composite synchronization information may be used to drive an external synchronization input of an RGB display. Electronics Institute of America (EIA) composite synchronization consists of a negative synchronization pulse that is compatible with (and may be driven by) an RS-170 composite video signal. Some

---

* Resolution in video lines can be measured by determining at what spatial location the tips of an extended "ray" or "sawtooth" figure can no longer be distinguished from background. Because the human visual system can resolved differences in contrast of less than 10%, this test essentially measures the modulation transfer function of the video system.

** Some video digitizer/display adapters (for example, the Data Translation *DT2853*) digitize images with square pixels and a horizontal resolution of 512 pixels by sampling only a portion of each video line at a rate of 12.5 MHz.

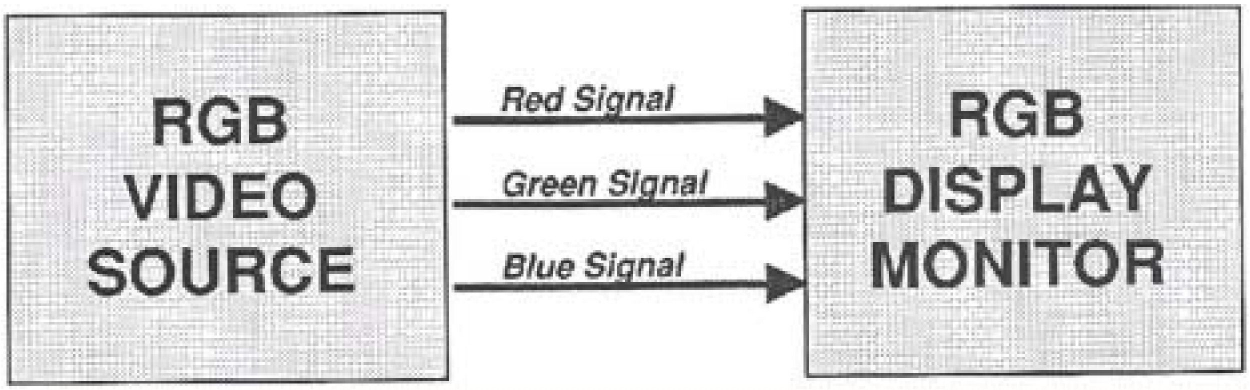


Fig. 4.

Schematic of RGB video. RGB video consists of three video channels, one each for red, green, and blue, *IMAGE* uses RS-170-compatible RGB video for display and output. Commonly, an additional signal carries synchronization information to the display monitor (external synchronization); or synchronization information is carried in the green channel. Without synchronization information. the displayed image will roll and flicker.

computer graphic displays and some cameras require separate horizontal and vertical drive inputs and may require Transistor/Transistor Logic (TTL) voltage pulses (3-5 V). RGB video is a "component video" format, meaning the various components of information required to reproduce a video display are enclosed by separate signals. Other component formats are used; for example, super VHS recording and display systems maintain separate luminance and chrominance (Y/C) channels.

In contrast, RS-170A or National Television System Committee (NTSC) standard color is composite video; all of the information required to reproduce the display is enclosed on a single channel. The NTSC signal is used for television in the United States and Japan. When color television was introduced, video formats were constrained by the Federal Communications Commission (FCC) to be compatible with the installed base of RS-170 black and white sets, and available electronics technology limited the band width usable for signal encoding. Consequently, NTSC video incorporates a "subcarrier" for encoding color; color information is phase encoded by a lower frequency chrominance signal superimposed on the luminance signal. While this system is adequate for many natural images, intense colors often spread beyond their natural boundaries during reproduction, and high spatial frequencies (such as the tweed in a sportscaster's jacket) may fool the chrominance detection circuitry and produce shimmering rainbows. Computer generated graphics, with their intense colors and sharp edges are particularly difficult to properly reproduce with NTSC video, although proper planning can minimize such effects. For example, broadcast graphics typically employ multiple-pixel-thick lines and type fonts, and black borders to delimit chrominance.

With the increasing importance of computer graphics and RGB displays, it is easy to find equipment compatible with analog RGB video, including video printers and film recorders, stillframe recorders, and display monitors. However, very few consumer or professional video tape devices provide analog RGB inputs; most accept composite video and some, such as super VHS, can accept other component formats. To record an analog RGB image, it is necessary to encode the separate color (and synchronization) channels into a composite video signal. For RGB signals in which resolution and timing are reasonably compatible with RS-170 standards, the encoding process can be simple and inexpensive. For example, stillframe recorders or video printers that accept RGB input and produce NTSC output can be used as encoders for some applications. Standalone encoders are also available ranging from $500 for a device designed to encode Apple Macintosh II RGB video to $5000 for a broadcast quality encoder. If the video source is high resolution or employs nonstandard timing, the expense of a device to encode NTSC video may exceed $15,000. The encoder must digitally subsample the video signal (or preferably interpolate between pixels), maintain a frame store, and produce a standard video signal for the digital data. For this reason, it may be cost effective for the original or auxiliary equipment manufacturer to develop a lower resolution NTSC compatible display subsystem that accesses the high-resolution display memory.

Signal-to-Noise Ratio

Noise in the video signal may arise from systematic or stochastic processes in virtually every hardware component. Fluctuations of the illumination system may be significant, particularly for quantitative measurements of transmitted light or fluorescence. The 60-Hz flicker associated with ac-driven sources can set up systematic intensity fluctuation across the video field. At low light levels, the random arrival of photons at the sensor produces "shot noise" that degrades an image. Also, the "dark current" associated with all photo sensors can limit the ability to quantify low light levels. Most sensors suffer from "readout noise" arising in the charge shifting electronics in a solid-state camera or from instability of the electron beam targeting or power supply in tube cameras. Internal amplifiers may be a source of noise, although a more significant problem is the lack of user-selectable signal controls (for gain, offset, linearity, etc.) on most consumer video equipment. Transmission cables can allow noise pickup, particularly when unshielded or long coaxial cables are used in an electronically noisy environment. Many analog recording systems such as video tape recorders, while adequate for capturing images, may be too noisy for critical quantitative work, and timing fluctuations (jitter) associated with such devices may make digitization difficult or introduce sampling artifacts. Digitizer systems can also introduce noise, although this is not a serious problem with most contemporary systems.

A number of technical strategies can reduce noise problems. The first step is to invest in good quality equipment and power supplies suitable for the intended application. Given the range of commercially available equipment, this need not imply major expense. Cables and devices should have proper shielding and should be properly terminated. If long cable runs are required, a video distribution amplifier may be useful. Signal summation or averaging is a useful strategy for image quality limited by stochastic processes such as shot noise. Under optimal conditions, averaging can improve the signal-to-noise ratio proportional to the square root of the number of individual trials contained in the average.

For high-performance applications, more sophisticated technical strategies may be employed. The use of cooled CCD or charge injection device (CID) cameras limits the photosensor dark current and allows long time exposures. By integrating ambient light in the sensor itself, shot noise can be averaged out while readout noise is limited. If greater temporal resolution is required in low-light situations, image intensifiers are available that can be coupled to a video camera via a lens system, or if more efficient light transfer is required, via "proximity focused" coherent fiber optic bundles. Intensifiers of traditional tube design consist of a photocathode input screen, a high-voltage stage to accelerate electrons, and a phosphor window. Second-generation devices typically employ a microchannel plate to achieve high gains, and hybrid devices exist that incorporate desirable features from each design. Such systems can achieve gains (photons in/photons out) of hundreds to hundred thousands with typical values of 10-20 thousand. Inefficient optical coupling schemes may reduce this value by 10-20 fold while optimal fiber optic coupling systems can have greater than 50% transfer efficiency.

Slow-scan cameras (which may also be cooled) employ nonstandard video rates to achieve low noise and high resolution. Slow-scan systems have traditionally been used to achieve higher spatial resolution than generally available within the RS-170 format. However, by limiting the bandwidth of video electronics, it is also possible to filter out high-frequency noise and to employ 12- or 14-bit digitizers to improve system dynamic range and intensity resolution.

Video Image Storage

Analog images stored on tape can seriously degrade over time. Digital storage and reproduction systems are a useful defense against image degradation following acquisition. The use of inexpensive microcomputer-based digitizing and display sub-systems together with existing digital storage media is satisfactory for many applications; however, the sheer volume of data associated with digital imagery can pose serious problems. Also few existing mass storage devices achieve the transfer rates necessary for real-time, continuous acquisition or display. While such capabilities are not essential for many applications, endeavors such as video animation would be greatly simplified by their availability. Ultra-high-density storage systems coupled with parallel digital data transfer or peripherals that produce a video readout should solve this problem in the future.

IMAGE ACQUISITION AND DIGITIZATION

Image acquisition involves the input of images, generally in analog form, for instance as a photograph or video signal. Digitization is the process of converting information from analog to digital formats, an analog-to-digital (A/D) conversion. For our purposes, we will consider image acquisition from the stage of input of a video signal to the video digitizer. This may be secondary acquisition, for instance if the image originated as a photograph or a video recording. In general, images are input to *IMAGE* through a standard black and white RS-170 video camera or from a video recorder. A solid-state video camera can serve to minimize geometric distortion, to minimize uneven response to given light intensities at different locations in the field of view, and to minimize electronic noise. Images are digitized with a framegrabber, which can convert a single analog video frame into a digital image in 1/30 second. The digital image is stored in an image buffer, which can be accessed and modified by the computer and which is used for display. The digitizer simultaneously digitizes and displays the image. When placed in a continuous acquisition mode, the images will be captured and displayed with a lag of one frame time (1/30 second) between acquisition and display. Display is accomplished by a digital-to-analog (D/A) conversion. The digital data in the image buffer is converted back into an analog RGB video signal that is shown on the display monitor.

IMAGE FORMAT

The program *IMAGE* uses images that are digitized at a spatial resolution of 512 (vertical) x 480 (horizontal) pixels, for a total of 245,760 pixels per image. Each pixel has one byte of information associated with it--a 7-bit intensity value, which ranges from 0 to 127, and a graphics overlay bit, which specifies whether graphics overlay is on or off (Fig. 5). An intensity value of 0 represents black, an intensity value of 127 represents white, and values in between represent intermediate gray levels. Each pixel is initially digitized with 256 levels of intensity information (8 bits), but (with the use of hardware LUTs) only 128 levels of intensity information are stored (7 bits). When the graphics bit is on, the pixel is displayed in a user-specified graphics color. Each pixel has an aspect ratio (the ratio of vertical/horizontal dimensions) of 5:6 for digitizers with a 12.5-MHz sampling rate. With a 5:6 aspect ratio, the distance represented by 5 pixels in the horizontal dimension is equal to the distance represented by 6 pixels in the

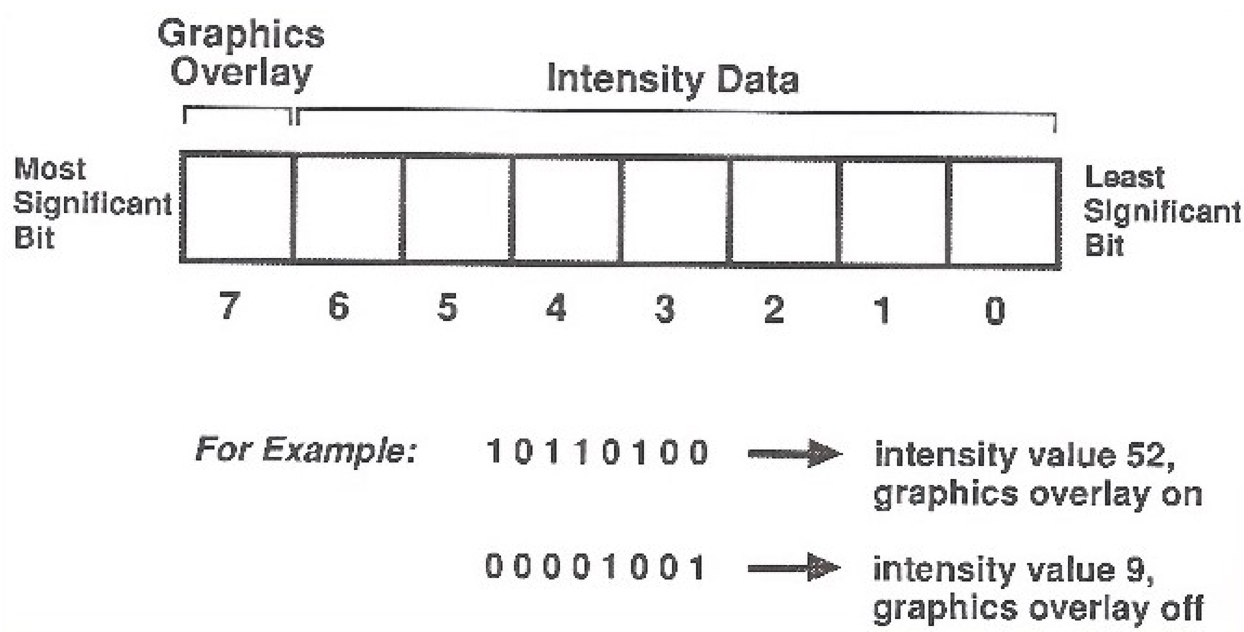


Fig. 5.

Conceptual representation of pixel information storage. A digital image comprises a series of pixels or picture elements. The digital images used in IMAGE have 1 byte of information for each pixel, for which 7 bits me used for intensity data and 1 bit is used to specify whether graphics overlay is on or off. The two examples show binary values and the corresponding conversion to decimal intensity values and specification of graphics overlay on or off.

vertical dimension.

To specify pixel locations, *IMAGE* uses a modified cartesian coordinate system, which is common in computer graphics and image processing, with the origin (0,0) in the upper left corner, increasing X values in the familiar left-to-right direction, and increasing Y values in the not-so-familiar top-to-bottom direction. In other words, the Y axis is flipped. Pixels are numbered from 0 to 511 horizontally and from 0 to 479 vertically. Thus the upper right corner has the coordinates (511,0), the lower left corner has the coordinates (0,479), and the lower right corner has the coordinates (511,479) (Fig. 6). This device coordinate system was adopted so that scan lines would be counted from top-to-bottom, while individual pixels are counted from left-to-right. However, when coordinate data are output using unit calibration, for instance output of trace or edge coordinate files, the origin (0,0) is in the lower left corner, with increasing calibrated X values from left-to-right and increasing calibrated Y values from bottom-to-top.

IMAGE BUFFERS

An image buffer is computer memory that is used for temporary storage of a digital image. Multiple image buffers allow rapid access to more than one image at a time, and are especially useful for operations between whole images. The image buffers used by *IMAGE* allow simultaneous image display and access by the central processing unit (CPU). Thus any changes to an image are immediately displayed. One image buffer is available for the PC Vision and FG-100, two image buffers are available for the PC Vision Plus, and four image buffers are available for the FG-100 1024.

INPUT AND DISPLAY LOOKUP TABLES

Lookup tables (LUTs) enable transformation of input and output data values in real time or near real time. In essence, a LUT is a list of numeric values that correspond to each of the possible data values. A LUT is literally a precalculated discrete function, returning a numeric value for each value that it is given. The data value is used as an index for the array of values in the table. *IMAGE* uses a digitizer/display adapter that has a set of hardware input and output LUTs. The hardware LUTs act in real time during input A/D and output D/A conversions (Fig. 7). For the input LUT, a single hardware 8-bit LUT channel is available. For each display LUT, three 8-bit LUT channels are available that correspond to the three channels of an RGB signal, one each for red, green, and blue. By varying the mix of red, green, and blue display intensities, any data value can be assigned a display color. An input LUT is used by *IMAGE* to transform 8-bit intensity values to 7-bit data values during the A/D conversion. A series of output LUTs are available for transformation of data values to display 1) positive and negative continuous tone images; 2) threshold images, which represent each pixel as black or white, without intervening gray levels, depending upon whether the pixel data values are above or below a threshold value; 3) slice images, which graphically highlight a user-specified range of intensity values; and 4) pseudocolor images, which substitute user-specified colors for intensity values. All LUTs are available instantly within all routines in *IMAGE*.

Input LUTS

Input LUTs can variously be used to transform data as it is digitized. For instance, a photographic negative can be transformed into a positive digital image, the contrast of an image can be increased or decreased, or an intensity range can be displayed as a graphics color. *IMAGE* uses a single input LUT, a positive linear ramp that transforms 8-bit input values to 7-bit data values. For the Imaging Technology digitizer/display adapters, input values 0 through 255 (8 bits) are linearly mapped to data values 0 through 127 (7 bits) to produce a continuous tone digital image (Fig. 8). For the Data Translation *DT2853* input values 0 through 255 (8 bits) are linearly mapped to even data values 0 through 254 (2, 4, 6,…,254) (the 7 most significant bits).

Continuous Tone Display

Continuous tone display enables one to view a monochrome (black and white) digital image as either a negative or a positive. Continuous tone display is achieved by using linear ramp LUTs, in which display values are a linear function of data values. If the slope is positive, a positive image is displayed; if the slope is negative, a negative image is displayed. Changing the slope and intercept of the line has the effect of altering the contrast. *IMAGE* uses a set of two linear ramp LUTs, one for positive and the other for negative display. For Imaging Technology

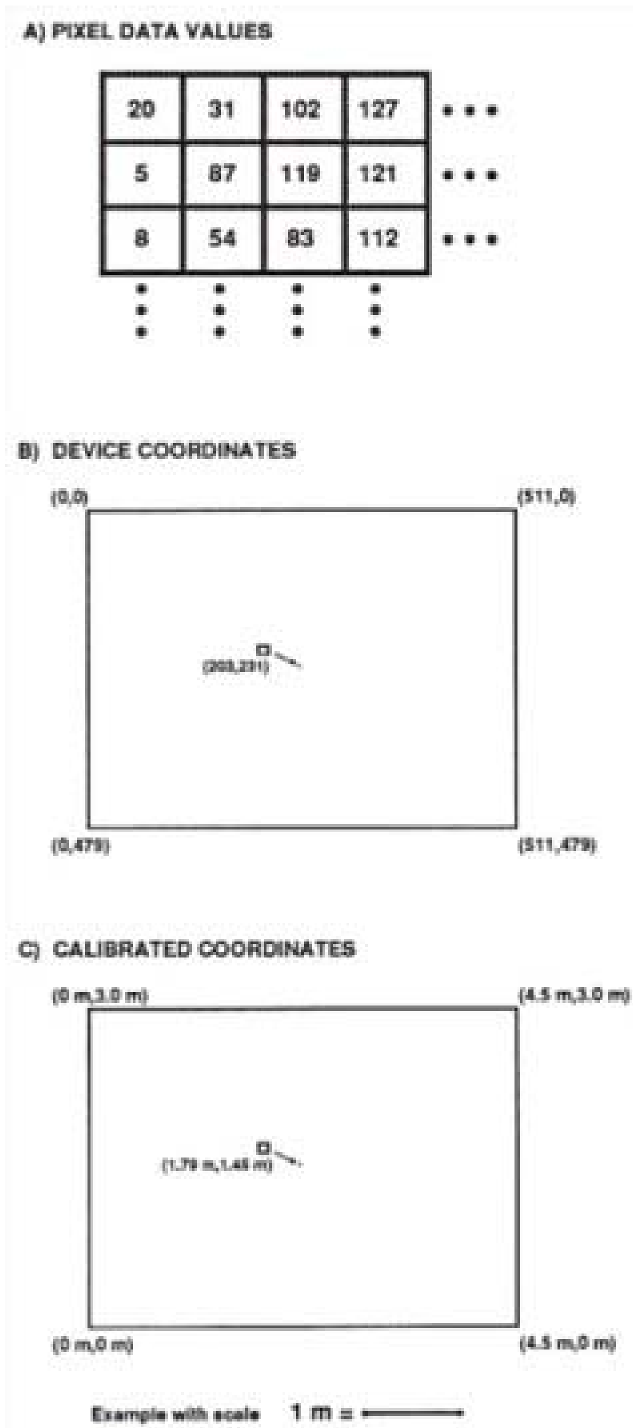


Fig. 6.

A) A digital image is essentially a huge array of data values that correspond to each pixel. The images used in IMAGE are an array of 512 (horizontal) by 480 (vertical) pixels, for a total of 245,760 pixels per image. B) Image processing and computer graphics commonly specify the location of individual pixels in an X,Y device coordinate system, with the origin (0,0) in the upper-left comer, X coordinates increasing from left-to-right, and Y coordinates increasing from top-to-bottom. C) *IMAGE* uses calibrated coordinates with the origin (0,0) in the lower-left corner, X coordinates increasing from left-to-right, and Y coordinates increasing from bottom-to-top. The example shows a calibration in which the image is 3 m high and 4.5 m wide.

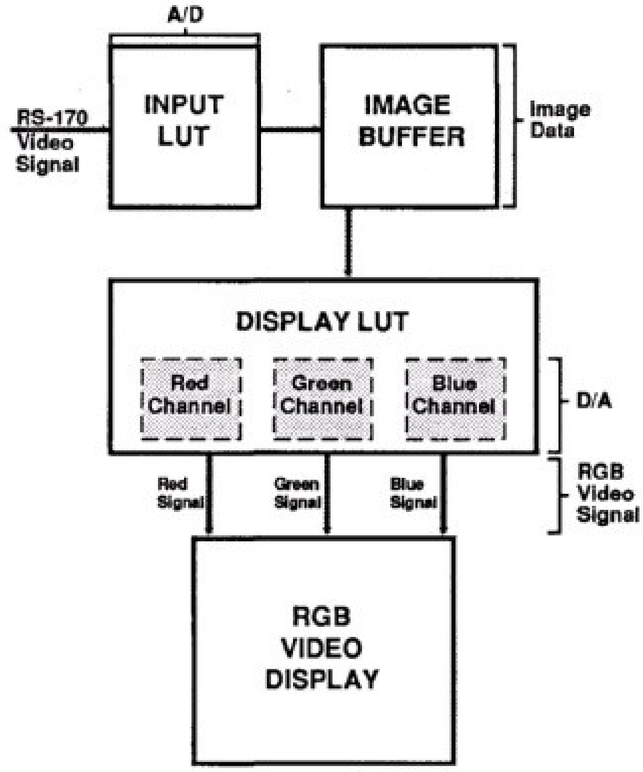


Fig. 7.

Schematic of LUT design. *IMAGE* uses both input and display hardware LUTs. The input LUT converts input values to data values, as the analog RS-170 video signal undergoes A/D conversion. The resulting data values are stored in an image buffer. The display LUT converts data values to display values, as the digital image data undergo D/A conversion to an RGB video signal. The display LUT has separate channels for red, green, and blue.

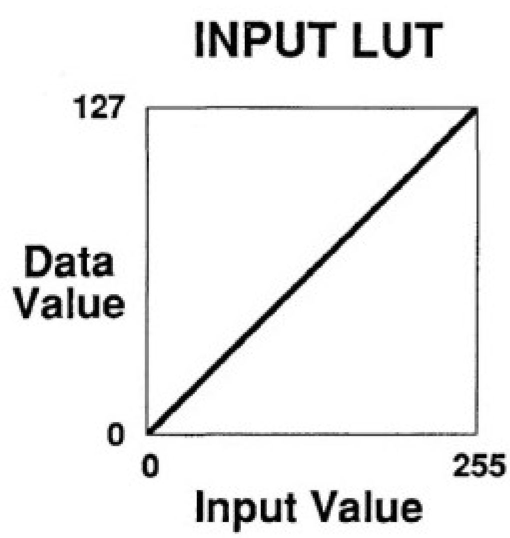
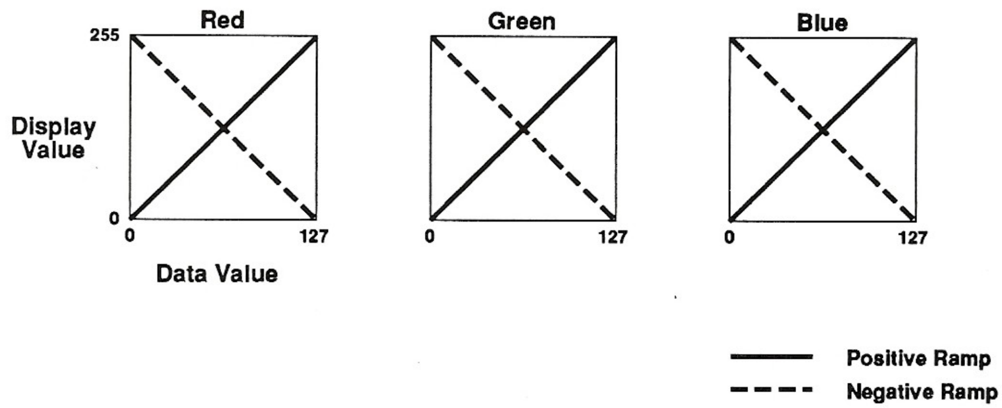


Fig. 8.

Input LUT mapping of input values to data values. *IMAGE* uses a linear ramp input LUT that maps 8-bit input values (0 to 255) to 7-bit data values (0 to 127).

A)



B)

C)

Fig. 9.
Continuous tone display LUTs. *IMAGE* provides both positive and negative linear ramp display LUTs for display of positive or negative continuous tone images. A) The 7-bitdata values (0 to 127) are converted to 8-bit display values (0 to 225) for each of the red, green, and blue channels. B) A positive image is displayed using a positive continuous tone LUT. C) The same image is displayed using a negative continuous tone LUT.

digitizer/display adapters, the positive ramp maps data values 0 through 127 (7 bits) to display values 0 through 255 (8 bits); the negative ramp maps data values 0 through 127 to display values 255 through 0. The red, green, and green channels of each LUT all have identical values, so the display image appears as a continuous gradation of gray levels (Fig. 9).

Threshold Display

Threshold display enables one to isolate image features that appear bright against a dark background or dark against a bright background. Threshold display is achieved by using a LUT in which all data values below a threshold value are assigned one display value and all data values above the threshold value are assigned a different display value. *IMAGE* uses a set of two threshold LUTs, a positive threshold LUT in which the display value for the lower range of data values is black (0) and the upper range is white (255), and a negative threshold LUT in which the lower range is white (255) and the upper range is black (0). As for continuous tone display, the red, green, and blue channels of each LUT are identical (Fig. 10). In *IMAGE*, the threshold can be interactively raised and lowered while viewing the thresholded image in real time. *IMAGE* uses the threshold LUTs in conjunction with a histogram routine to calculate areas within a region of interest that are above and below a threshold value.

Slice Display

Slice display enables one to isolate image features that lie within a range of intensity values. All pixels with data values within the intensity range are highlighted with graphics. Slice display is achieved by using a LUT in which all intensity values within a specified range are assigned the same mix of red, green, and blue display values to produce graphics highlighting. *IMAGE* uses a set of two slice LUTs, a positive slice LUT that shows slices for positive continuous tone images and a negative slice LUT that shows slices of negative continuous tone images (Fig. 11). In *IMAGE*, the slice range can be interactively changed while viewing the slice image in real time (Color Plates IA & IB). *IMAGE* uses the slice LUTs to define objects or features of interest for editing and measurement with the flood, automatic edge detection, and histogram routines.

Pseudocolor Display

Pseudocolor display substitutes color for intensity. Each data value can be assigned a display color. *IMAGE* uses a set of two alternative LUTs for pseudocolor display. The LUTs can be loaded and saved as external ASCII files, or may be manipulated from within the program, which enables user specification of any LUT. The display color for each data value is determined by the mix of red, green, and blue specified by an 8-bit value for each RGB color channel, with 16,777,216 (28x28x28) available colors, and allowing up to 256 colors to be displayed simultaneously (see Fig. 12, color plates IC and II, and cover illustration).

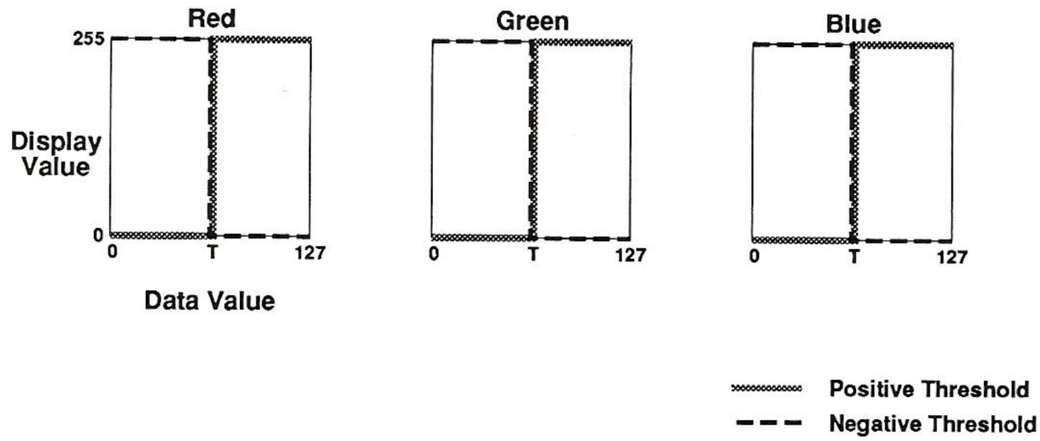
Graphics Overlay

Graphics overlay on digital images is accomplished by reserving a bit plane or series of bit planes to contain graphics. Graphics overlay does not change image data values. This nondestructive graphics overlay is useful for producing colored cursors and for such capabilities as using the flood routine to measure areas. *IMAGE* uses a 1-bit graphic plane to overlay graphics on images. When a pixel's graphic bit is on, the pixel is highlighted in a user-defined color. This is accomplished by assigning all output LUT display values for data values 128 to 255 to be a single color as determined by values specified for the red, green, and blue LUT channels (Fig. 13).

LUT Editing

LUT editing involves specifying a display value that corresponds to each data value. This is often best accomplished with an interactive editor that allows the user to specify X,Y plots of display versus data values for each of the red, green, and blue channels. *IMAGE* provides interactive editing capabilities to produce any possible display LUT. Once a LUT has been created, it can be saved to a file and loaded at any time. Also, LUT files can be

A)



B)

C)

Fig. 10.
Threshold display LUTs. *IMAGE* provides both positive and negative threshold display LUTS. A) The positive threshold LUT displays all data values below a threshold value (T) as 0 and all data values greater than or equal to the threshold as 255 for each of the red, green and blue channels. The negative threshold LUT displays all data values below a threshold value (T) as 255 and all data values greater than or equal to the threshold as 0 for each of the red, green, and blue channels. B) The image from Fig. 9B and 9C is displayed using a positive threshold LUT. C) The same image is displayed using a negative threshold LUT.

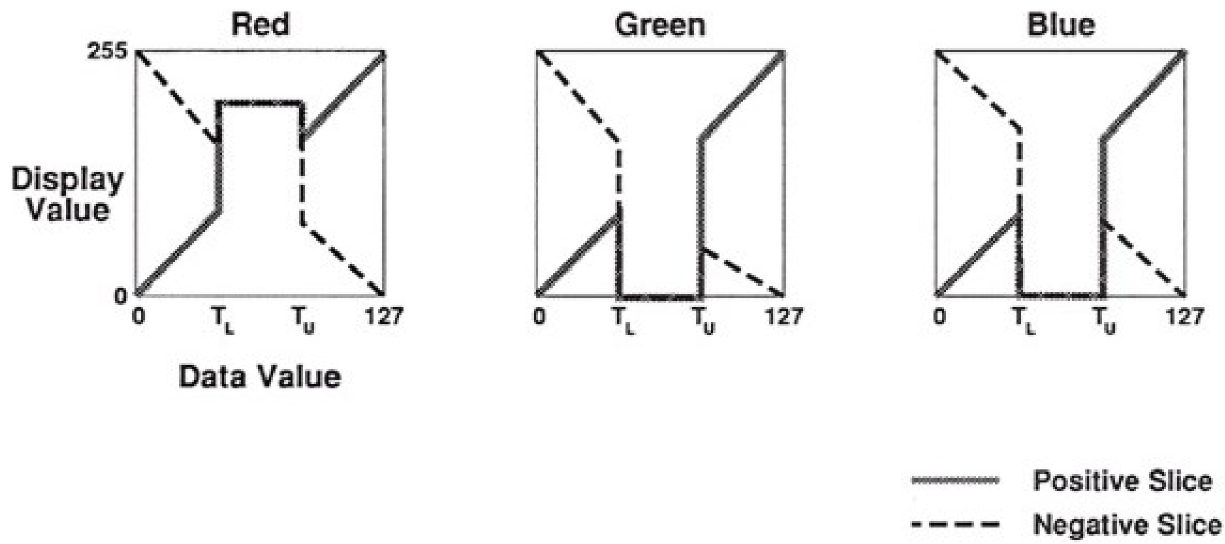


Fig. 11.

Slice display LUTs. IMAGE provides both positive and negative threshold display LUTs. The positive slice LUT highlights a range of data va1ue.s (slice range), from a lower threshold (T_L) to an upper threshold (T_U), with data values outside the range displayed as a positive linear ramp. The negative slice LUT highlights a slice range, with data values outside the slice range displayed as a negative linear ramp. In the example, the slice range is highlighted in red, because green and blue display values are set to 0. Color Plates IA and LE illustrate slice LUT display.

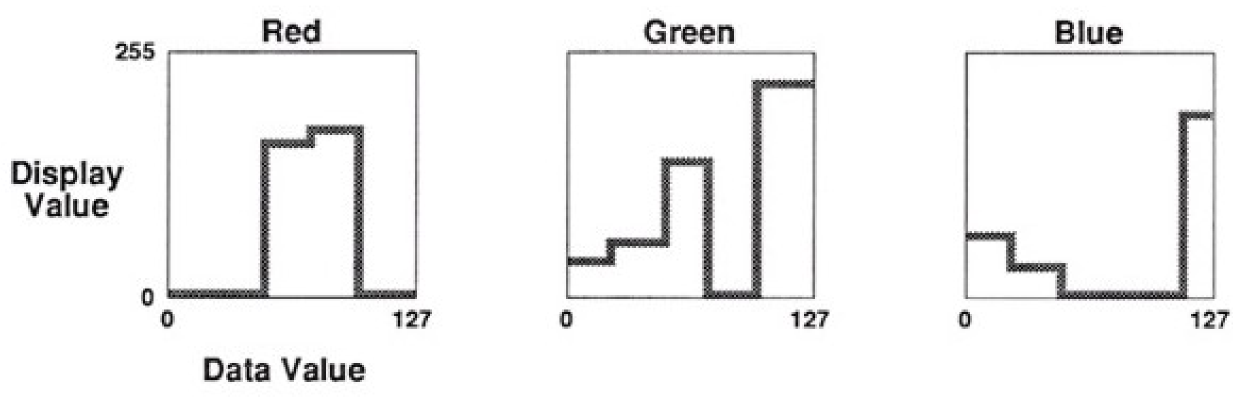
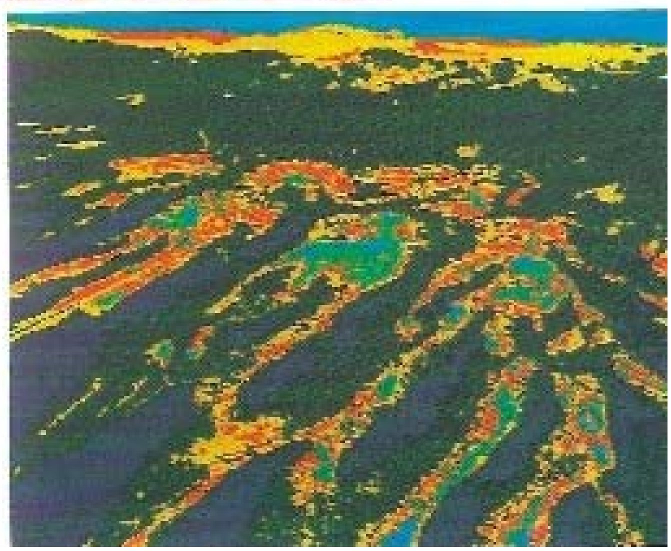


Fig. 12.

Pseudocolor display LUTs. *IMAGE* provides two pseudocolor display LUTs. Pseudocolor display involves substituting color for an intensity value or range of intensity values. This example shows a mapping of six data value ranges to six colors as defined by a mix of red, green, and blue LUT channels. Color Plate IC illustrates pseudocolor display using the example LUT. The pseudocolor LUTs can be loaded with any user-specifiable LUT.

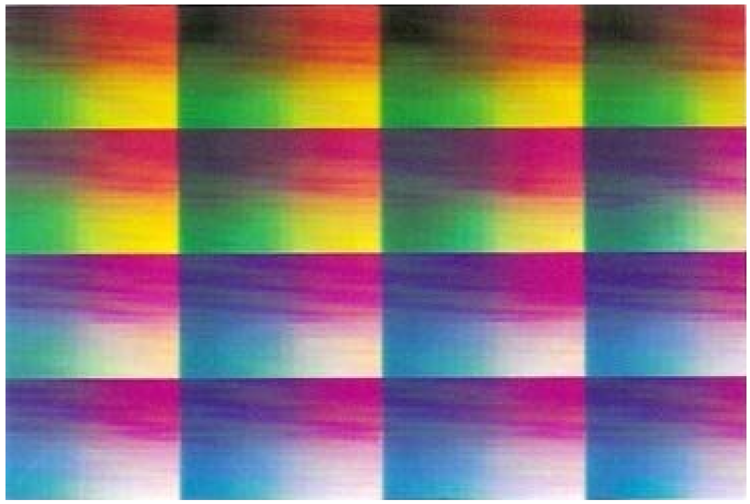
A)

B)



C)

Color Plate I.
A) A positive slice display for the image illustrated in Fig. 9. The slice range is from data values 52 to 88. B) A negative slice display for the same image. C) A pseudocolor display for the same image, using the pseudocolor LUT described in Fig. 12.



Color Plate II.
A representation of color mixing. Colors are produced by mixing different intensities of red, green, and blue. This 12-bit image (produced on a *FG-100* display) contains 4096 different colors. Each of the 16 large rectangles blends red and green into 256 combinations. Blue is added incrementally to these 16 rectangles, ranging from all the way off (upper-left rectangle) to all the way on (lower-left rectangle).

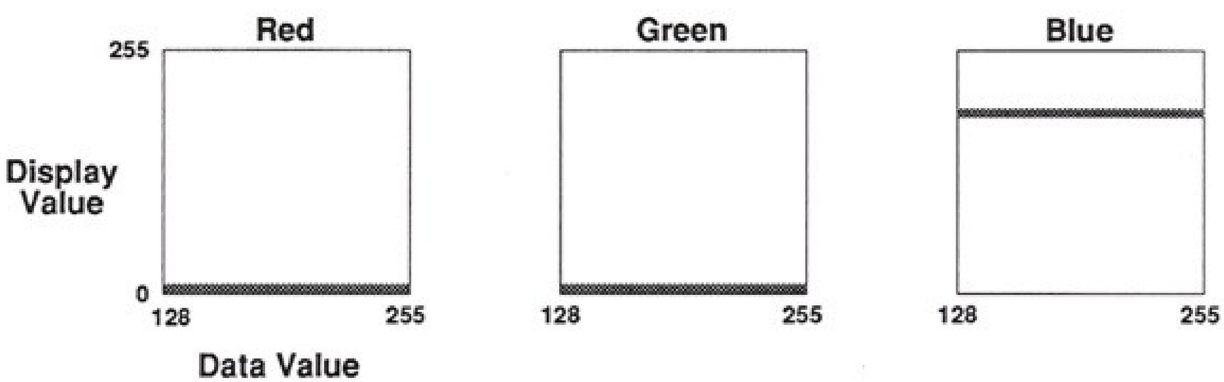


Fig. 13.
Graphics overlay LUTs. *IMAGE* uses the high bit (most significant bit) to define graphics overlay and cursors. Data values 128 to 255 are all mapped to display values that define a graphics color. This example shows a graphics color of blue because red and green are set to 0.

created with simple user programs, for instance a program that outputs an exponential transformation LUT. A library of LUT files can be established, containing LUTs useful for particular applications (e.g., alternative pseudocolor LUTs, exponential and logarithmic transformation LUTs, and other contrast enhancement LUTs).*

Mapping Display LUTs Back to Data

Digital images can be modified by changing each data value to its display value, as specified in a display LUT. For instance, when using a negative ramp, mapping the LUT values back to the data will have the effect of producing data values for a negative image. *IMAGE* has a utility for mapping display values back to data, with the user specifying which of the three output LUT channels to use--red, green, or blue.

EDITING

Editing involves changing the data values of pixels, individually or in regions. Editing can allow the user to block out regions of an image that are to be excluded in analysis or display, and to enhance regions of an image that are not distinct. Editing is one means for classifying images. By modifying data values of a feature to be a single data value, the feature can be made distinct. This is especially useful in cases where a human being can readily recognize a feature, but no simple algorithm exists for automated recognition of the feature. Editing also allows overlay of graphics information and annotation on images. *IMAGE* has routines for drawing points, lines, curves, unfilled and filled rectangles, unfilled and filled circles, and for flooding regions. All routines are interactively controlled using the keyboard and a mouse or other locator device. Generally, a cross-hair cursor or rubber-band graphics cursor is used to specify the precise location of any graphics changes. A cross-hair cursor appears as a graphics cross on the image display; and a rubber-band cursor appears as a graphics form (e.g., a rectangle) that can be stretched to any size and shape. The user can specify any value (the active value) for use when editing (range 0 to 127), or alternatively perform editing with graphics overlay. The active value replaces the data value for each selected pixel. Editing in graphics overlay does not modify data values. Many of the editing routines also have other capabilities. For instance, the line routine also measures linear distance.

LINEAR AND AREA MEASUREMENTS

Precise linear and area measurements are useful for many scientific applications. *IMAGE* includes a variety of capabilities to enable one- and two-dimensional measurements. For convenience, *IMAGE* calculates distances in X pixel lengths or calibrated linear units, and calculates areas in number of pixels or calibrated square units. All routines can be calibrated to make measurements in real units, such as millimeters or kilometers. Measurement routines also provide capabilities for editing data.

Unit Calibration

Unit calibration requires input of 1) a coefficient that converts pixel lengths to real units and 2) a unit label. *IMAGE* permits interactive input of a conversion coefficient and unit label, which are used to calibrate all measurements from all routines. The conversion coefficient is determined by first stretching a rubber-band line cursor along any known reference in an image, for instance a meter stick digitized at the same scale as the image to be analyzed. Then the user specifies the number of units and the unit label that correspond to that distance. All subsequent measurements are expressed in both pixels and calibrated units. Because the pixels are not square, length is expressed as the number of pixels in the X dimension. The current conversion coefficient and unit label are stored in a configuration file, so unit calibration is only necessary when the measurement scale has changed.

Line Measurement

IMAGE allows linear measurement by stretching a rubber-band line cursor across any span to be measured. The line routine also allows drawing lines directly in images and in graphics overlay.

---

*To maintain consistency of operation, for the Data Translation *DT2853*, the LUT editor acts as though the most significant bit is used for graphics overlay, just as it actually is for the Imaging Technology digitizer/display adapters. In the actual LUTs, the odd LUT indices represent graphics values (least significant bit on).

Curve Measurement by Tracing

IMAGE allows measurement of distance by interactive tracing of curves in graphics overlay. Traces are drawn in one of two modes, one in which drawing occurs continuously while a mouse button is pressed, and one in which a series of line segments are connected each time a mouse button or the keyboard <ENTER> key is pressed. The trace routine calculates distance along each traced curve, and optionally calculates the area of the polygon formed by joining the ends of traced curves. Measurements of length and area can be output to ASCII files. X,Y coordinates of inflection points can be saved to ASCII coordinate files; and coordinate files can be loaded to overlay graphic tracings on an image. Optionally, any traced curve can be written to the image in a specified data value.

Area Measurement with Flood Routines

A flood routine searches for all contiguous pixels that lie within a specified range of data values, starting from a seed pixel, and changes the data value or turns on the graphics overlay of each pixel within the region. In *IMAGE*, the flood routine calculates the number of contiguous pixels and displays the area of each region. Regions to be flooded can be specified and previewed using the slice LUT. The slice LUT determines the range of intensity values to be flooded. The user specifies the seed point of a region to be flooded by using a cross-hair cursor. When using the flood routine for measurement, regions are flooded by highlighting in graphic overlay. An automatic scanning mode optionally locates all regions in an image and keeps track of the area of each region. Measurements of area can be output to ASCII files. When using the flood routine for editing, all data values within the region are changed to a single data value.

Automatic Edge Detection

Edge detection involves recognizing a boundary between an object or feature of interest and the background. One form of edge detection recognizes edges as defined by a threshold intensity value. The edge of a dark object against an evenly lit light background is defined by a threshold intensity value above which a pixel is object and below which a pixel is non-object. An edge detection routine based on threshold values works by 1) progressively searching pixel-by-pixel until it finds an edge, defined by adjacent pixels one of which is in the object and one which is outside the object (as determined by the threshold value); 2) then the search continues along the edge, storing the X,Y coordinates of each boundary pixel; 3) and finally stopping the search when a region has been completely circumscribed.

IMAGE has an edge detection routine that searches edges as specified and previewed using the slice LUT. The slice LUT defines an intensity range (upper and lower threshold) that is used by the edge detection routine. The user places a cross-hair cursor either to the left of a feature of interest or within a feature of interest. The edge detection routine first searches to the start of a boundary and then follows the boundary, highlighting the boundary and recording the location of all boundary coordinates. The edge detection routine calculates distance (perimeter) along each boundary, the area enclosed by the boundary, and the X,Y coordinates of the centroid. An automatic scanning mode optionally locates edges of all regions in an image and keeps track of the length, area, and centroid coordinates for each region. Measurements of length, area, and centroid can be output to ASCII files. Coordinates of boundary pixels can be saved to ASCII coordinate files; and coordinate files can be loaded to overlay boundary tracings on an image. Optionally, any boundary curve can be written to the image in a specified data value.

RASTER AND VECTOR CONVERSION

Raster data refers to information that is stored as a series of pixel values, essentially a matrix of intensity values-the usual storage format for digital images used for image processing. Vector data refers to information that is stored as a series of X,Y coordinates that define the boundaries of a series of curves or closed polygons that comprise an image. *IMAGE* includes capabilities for converting from raster to vector formats and from vector to raster formats. The coordinate files saved from the trace and edge routines, the tracings or edges, respectively, are saved in a vector format. These coordinate files can be used as input for vector-based programs. When coordinate files are loaded into the trace or edge routines, they are converted from vector to raster formats. Raster data generated from any source can be converted and displayed in vector format if converted to the proper format (see Appendix D for file formats).

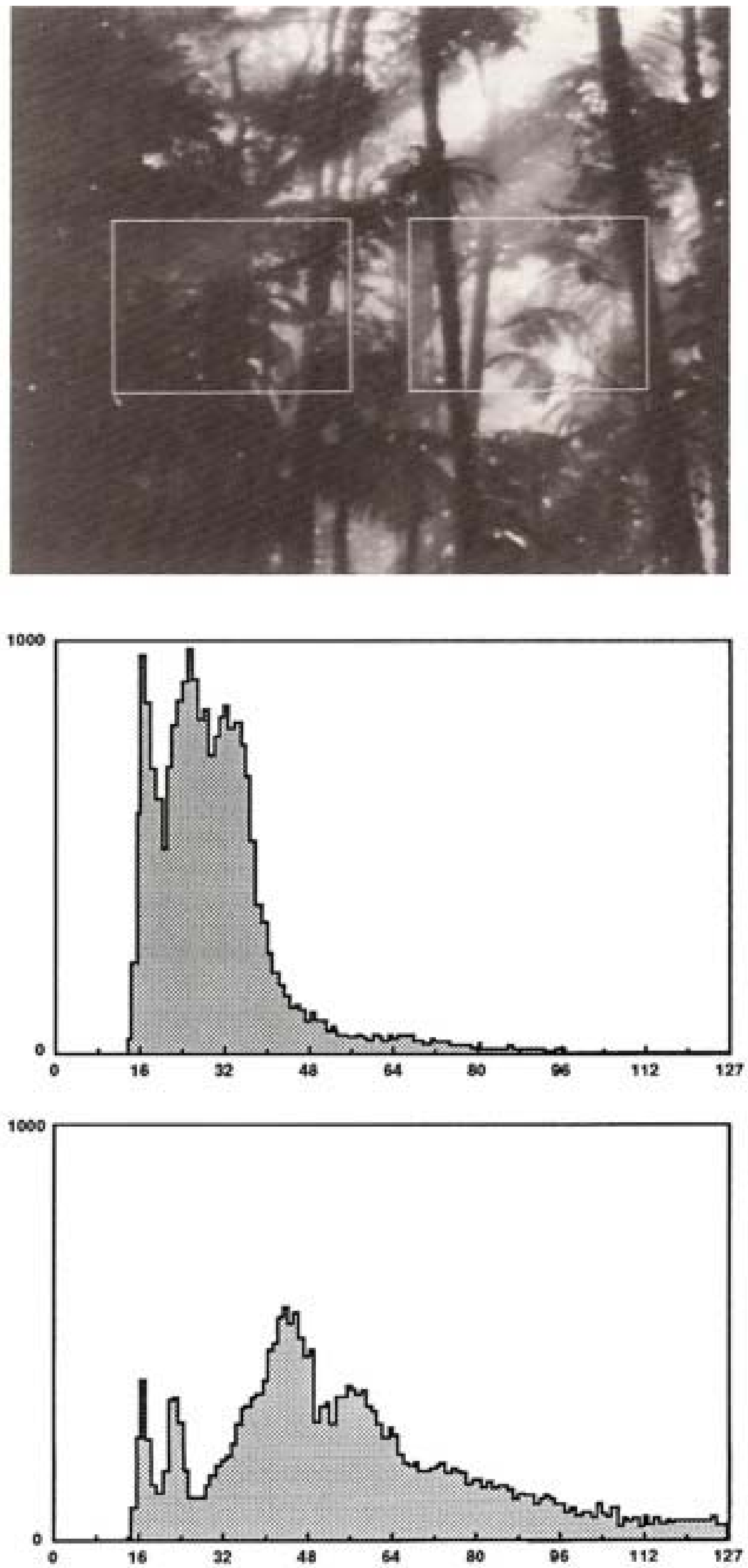


Fig. 14.

Histograms. A histogram describes the distribution of data values within a digital image or region in an image - a tally of the number of pixels with each data value. In this example, an image A) shows two rectangular regions of interest for which histograms shows that B). the left region, has a large number of dark pixels, whereas C), the right region, has a large number of light pixels.

HISTOGRAMS

A histogram is used to represent the distribution of intensity values within an image or region of an image. The distribution is generally represented in a bar graph showing the number of pixels for each possible data value (Fig. 14). The histogram bar graph enables one to rapidly view the contrast range and any obvious groupings, such as a skew toward light or dark pixels. *IMAGE* allows one to calculate a histogram of any rectangular region within an image. The region of interest is defined using a rubber-band rectangle cursor. The histogram is displayed as a bar graph, and results are optionally output to a data file. The histogram routine also allows one to calculate the number of pixels for any range of data values and the area represented by that number of pixels. Also, the histogram routine can be used in conjunction with the slice LUT to view and change the range of data values on the image screen while the number of pixels and area are calculated.

EXAMINATION OF INTENSITY DATA

Some applications require examination and output of the intensity values of a series of pixels. For example, this is useful when scanning electrophoresis gels, wherein different proteins will appear as dark regions spaced along a white background. In *IMAGE*, intensity values of individual pixels can be examined using any of the interactive cursors, for instance, the cross-hair cursor in the trace routine. *IMAGE* provides capabilities to output files with pixel coordinates and data values along any line or curve specified using the trace routine. The intensity file output can then be used for further analysis or entered into a plotting program.

REGION MOVEMENT AND RESCALING

It is often useful to manipulate the size, shape, and position of regions within an image. Such changes may be desirable for many reasons, including 1) to correct geometric distortions, 2) to size images to a common scale, or 3) to piece together a series of images into a single image (Fig. 15).

IMAGE includes a mosaic mode, which allows the movement and rescaling of regions of interest. Any rectangular region within an image can be copied to any other rectangular region in the same image, or in an image in another data buffer. The capability of being able to fit together a series of images or regions of images in a single composite image gives the mosaic mode its name. The region to be copied (source region) is rescaled in size and aspect ratio depending upon the relative size and aspect ratio of the region to which it is copied (target region). Pixels are replicated or deleted proportional to the respective increase or decrease in size. For example, an increase

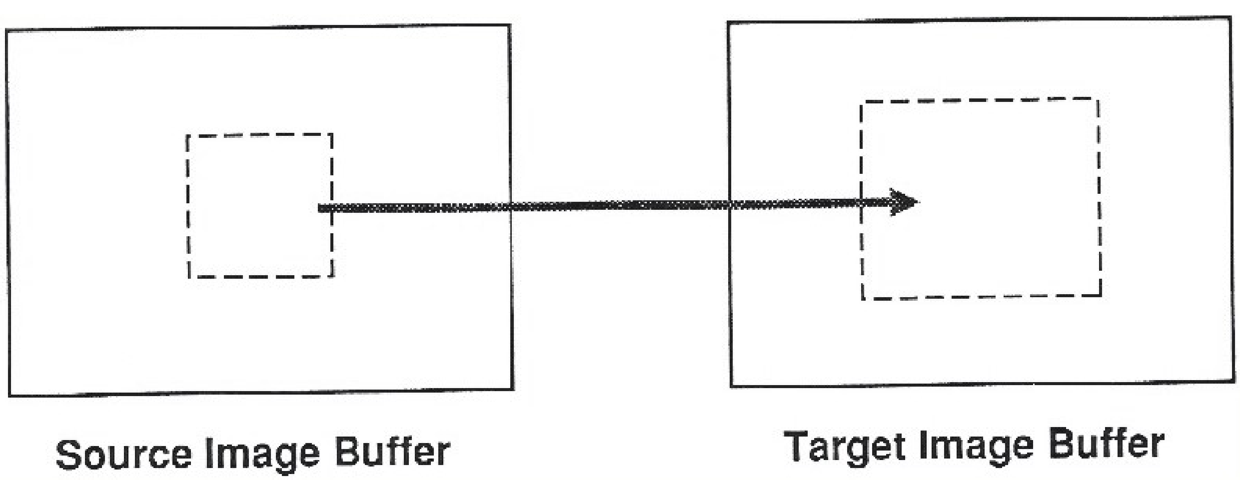


Fig. 15.
Conceptual representation of image region movement and rescaling. Region movement and rescaling involves copying a rectangular region in a source image buffer to a rectangular region in a target image buffer. The second rectangular region can be a different size or shape than the first region, allowing for both movement of regions and rescaling in the X and Y dimensions independently.

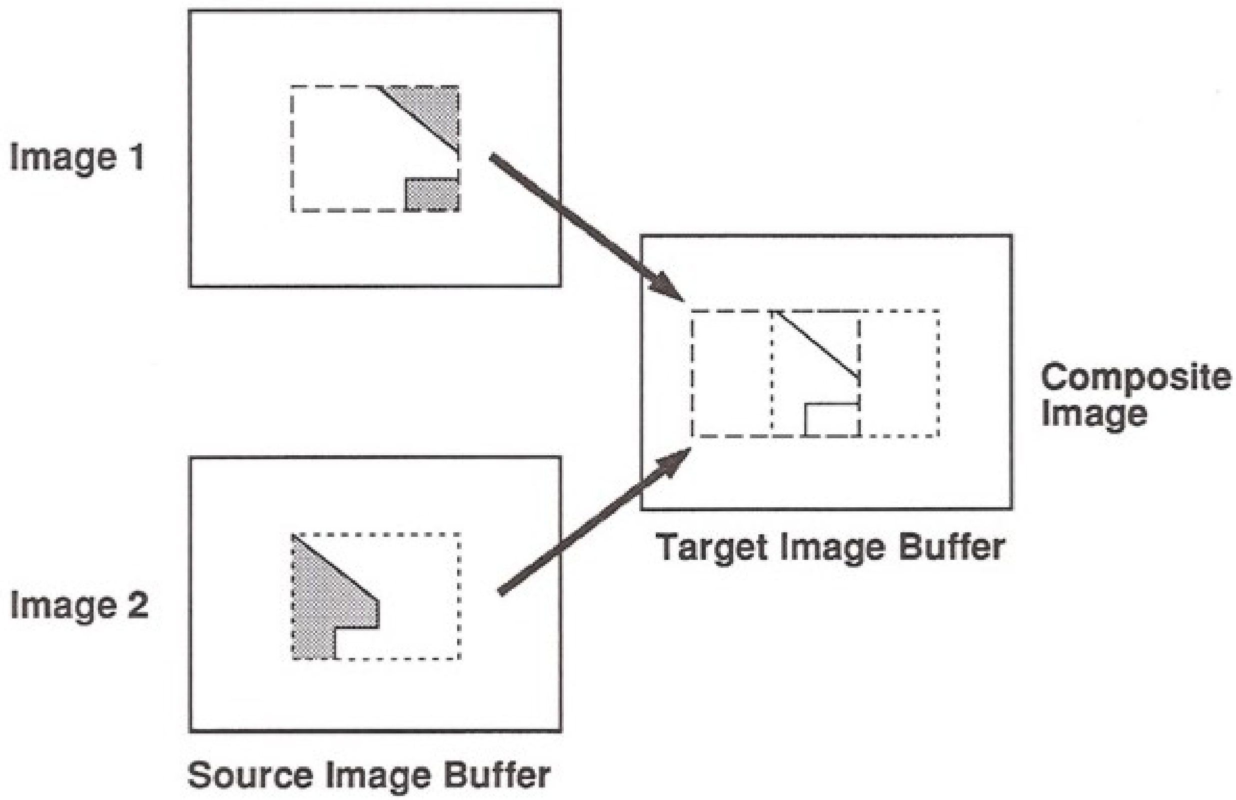


Fig. 16.

Conceptual representation of mosaic operation. A mosaic operation refers to region movement and rescaling that allows complex edges in regions to be fitted together precisely in a jigsaw-puzzle fashion. This is accomplished by specifying a transparency range, a range of intensity values that are not copied from the source image buffer to the target image buffer. Regions that are not to be copied from a source image buffer can be edited to have a data value within the transparency range (shown in shaded areas). The transparency range prevents regions in one image from being copied over regions from another image. Thus, composite images can be formed from source images that are at different scales and that fit together along complex edges.

in size of 1/3 in the horizontal and vertical dimensions would cause every third source pixel to be replicated in the target region, with no change in aspect ratio. Similarly, a decrease in size by 1/2 in only the vertical dimension would cause every other source pixel in the vertical dimension to be deleted in the target region, and would half the aspect ratio.

Lists of source and target regions can be used to copy and rescale a series of regions, for instance to produce a composite image from a series of different regions in different images. Each list specifies the data buffers and X,Y coordinates (upper left and lower right) that define a series of rectangular source and target regions. These lists can be saved and loaded as ASCII files. A transparency range can be specified to allow irregular edges to be fit together precisely in a composite image, even though the mosaic routine only works with rectangular regions. A transparency range refers to a range of data values that are not copied from the source to the target region.

The use of source/target lists and transparency ranges is best illustrated with an example, wherein a series of source images are pieced together to produce a single target image (Fig. 16). The source images may be at different scales and may have irregular edges that fit together in a jigsaw-type pattern. All portions of each source region that are not to be copied to the target image can be edited to 0, and the transparency range can be set to 0. A list of source and target regions is then produced, with coordinates of source regions specified relative to internal references and coordinates within the target region in absolute locations that will produce a single composite image. Transparency prevents parts of different source images from being copied to the same target region, even though the rectangular coordinates of the different target regions must overlap along irregular edges. During the mosaic operation, each source region is successively rescaled and placed in its proper position within the target composite image.

ARITHMETIC AND LOGICAL OPERATIONS

Arithmetic and logical operations can be performed between a scalar and a digital image or between two digital images. Scalar operations are performed successively on intensity values for each pixel of an image, for instance adding a constant to each pixel value. Whole image operations are performed with pixel-by-pixel correspondence, such that operations are performed between the pixel value at a given position in one image and the pixel value at the same position in a second image. For example, one image can be subtracted from another image.

IMAGE includes various scalar and whole image operations. Among the scalar operations, a linear transform allows linear rescaling of an image, with the user specifying the slope and intercept of the transform (Fig. 17). This is equivalent to a linear contrast stretch. For example, in conjunction with an examination of the histogram, the user can choose a slope and intercept that spreads the data values over the full range of possible data values, thus increasing the visible contrast. The linear transform allows one to add, subtract, multiply, and divide constants from images, depending upon the choice of slope and intercept. A bit-shift operation allows the user to shift image data values one bit to the left or right. A bit shift to the left has the same effect as multiplying by two and a bit shift to the right has the same effect as dividing by two. This capability is included primarily to allow conversion of images in the 7-bit data format of *IMAGE* to and from 8-bit formats of other programs. Bit shifting to the left will put images in a format that can be displayed by many other programs. Bit shifting to the right will allow 7-bit data to be used in *IMAGE* (with the loss of the least significant bit).

Among the whole image operations available in *IMAGE*, the arithmetic operations (addition, subtraction, multiplication, and division) allow rescaling and correction of intensity values as a function of position. For instance, a correction image can be subtracted from a data image to compensate for unequal lighting across the field of view. Bitwise logical operations act on the individual bits of an 8-bit pixel, producing new values according to the truth tables shown in Table II. Logical whole image operations (AND, OR, and XOR) allow pixel-by-pixel bitwise comparison of two images (Fig. 18). For instance, a whole image XOR (exclusive or) will show any differences between two images because any time two bits are different, the XOR operation returns 1 and any time two bits are the same, the XOR operation returns 0.

DATA INPUT AND OUTPUT

Convenient and effective data input and output is essential for any image analysis system. *IMAGE* reads and writes a variety of data files, including image files, flood result files, trace and edge coordinate files, trace and edge result files, histogram files, intensity files, and mosaic coordinate list files. Images are stored in a binary format, which literally contains a linear list of all byte values within an image. All other data output files are in ASCII format, which, because it is a standard format, allows files to be created and edited from a text editor, and which allows files to be used in other analysis programs such as statistics and graphics packages. Appendix D gives a detailed description of all file formats.

Table II. Logical bitwise operations truth tables.

a b	a AND b	A OR B	A XOR b
0 0	0	0	0
0 1	0	1	1
1 0	0	1	1
1 1	1	1	0

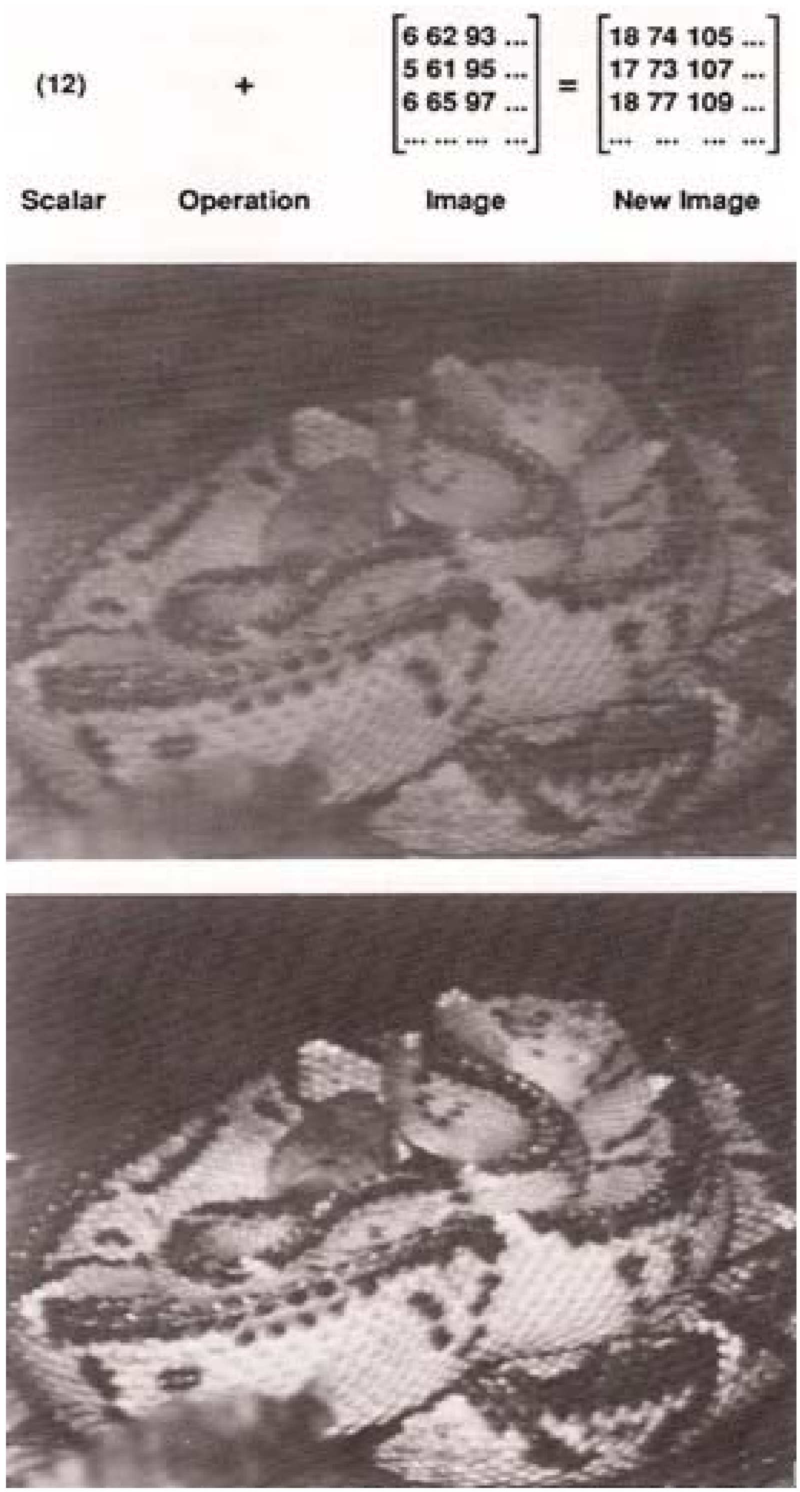


Fig. 17.

Scalar operations on images. Scalar operations involve performing the same operation (e.g., division) between a scalar number and each of the data values of a digital image. A) In this example, 12 is added to each of the data values of an image. A linear contrast stretch involves two scalar operations, multiplication and addition. Linear contrast enhancement involves a linear transform, $D' = aD + b$, where D' is the transformed data value, D is the original data value, a is the slope, and b is the intercept. A low contrast image is shown B) before a linear contrast stretch and C) after a linear contrast stretch.

A)

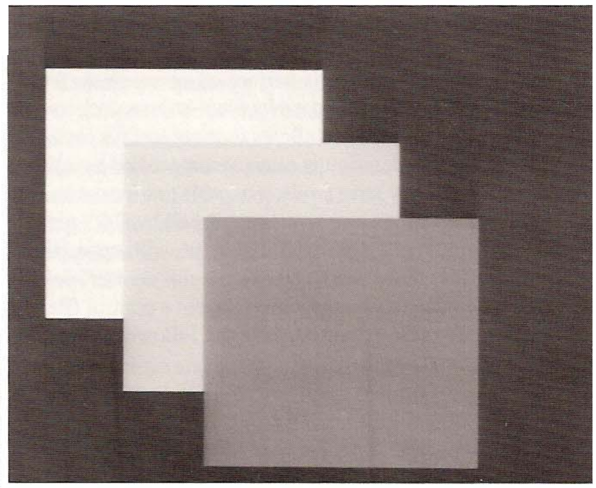
$$\begin{bmatrix} 131 & 109 & 53 & \dots \\ 102 & 98 & 41 & \dots \\ 115 & 97 & 44 & \dots \\ \dots & \dots & \dots & \dots \end{bmatrix} \text{ XOR } \begin{bmatrix} 131 & 108 & 64 & \dots \\ 102 & 98 & 41 & \dots \\ 93 & 97 & 44 & \dots \\ \dots & \dots & \dots & \dots \end{bmatrix} = \begin{bmatrix} 0 & 1 & 117 & \dots \\ 0 & 0 & 0 & \dots \\ 56 & 0 & 0 & \dots \\ \dots & \dots & \dots & \dots \end{bmatrix}$$

Image 1 **Image 2** **New Image**

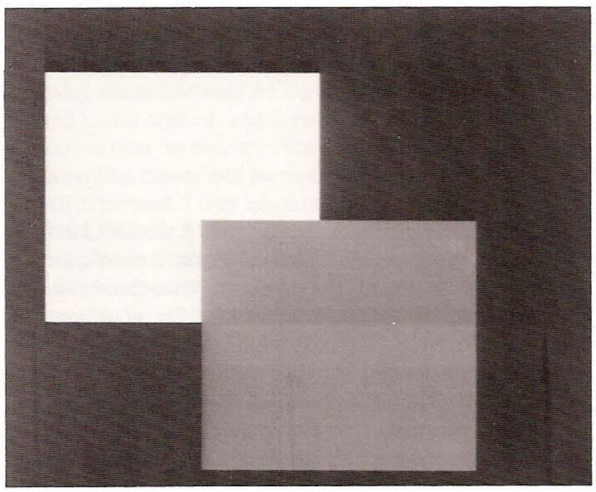
B)

Decimal Value	Binary Value
115	0 1 1 1 0 0 1 1
XOR 93	0 1 0 1 1 1 0 1
46	0 0 1 0 1 1 1 1

C)



D)



E)

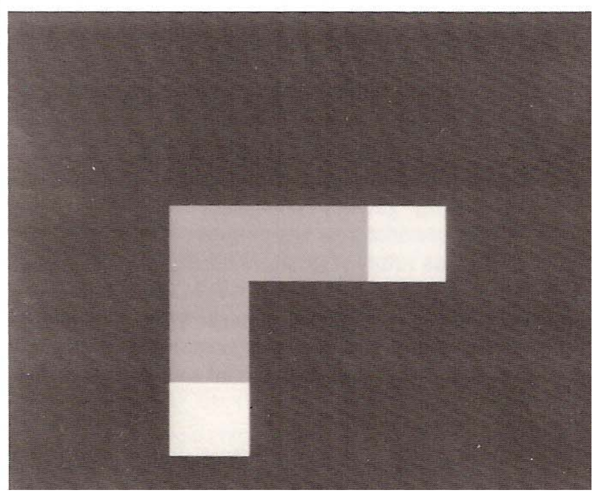


Fig. 18.
Whole image operations. Whole image operations involve operations on a pixel-by-pixel basis between two images to form a third image. Operations are performed on pixels that correspond to the same image positions within different images. A) In this example a logical XOR operation is performed between two images. Part B) shows that a logical XOR operation operates in a bitwise fashion to return 0 if two corresponding bits are the same and 1 if two bits are different. A logical XOR between two images C) and D) results in an image E) that shows differences between the original images.

IMAGE OUTPUT

Images can be output to various hardcopy devices, including photographic devices, video printers, and graphics printers. Photographic devices and video printers can directly use the digitizer/display adapter RGB video output to produce high-quality hardcopies. In addition, the utility program *I2TIF* can be used to produce a standard graphics TIF file that can be used to produce hardcopies on graphics printers (see *I2TIF* in Appendix E). The trace and edge coordinate files can be used to produce line plots of features in images.

AUTOMATIC PROCESSING AND ANALYSIS

Automatic processing and analysis are useful for many applications, especially applications that involve repeated processing steps. The flood and edge detection modes of *IMAGE* have automatic scan modes, in which a flood or edge detection operation is performed successively for each pixel location in an entire digital image. Fully automatic operation is only possible where human decisions are not necessary or decisions can be made ahead of time. Semiautomatic operation allows user control for certain steps and automatic control for others. Such automatic and semiautomatic operation is often accomplished by using a control file to control processing and analysis. *IMAGE* does not currently have explicit capabilities for automatic processing and analysis using control files, however automatic operation can be accomplished by DOS redirection of input. A control file can be produced that contains ASCII characters for all keyboard input, with appropriate ASCII values for control characters such as <ESC>. Control files can be produced using a simple user program or with a text editor that allows specification of ASCII control characters. To run *IMAGE* from a control file, DOS redirection of input (<) is specified on the command line when *IMAGE* is started, as in the following example:

IMAGE < CONTROL

where *CONTROL* is the name of a control file.

CONFIGURATION

Whenever *IMAGE* starts, it reads in a configuration file that contains user-definable default values. Included in the configuration file is a specification of the digitizer register and buffer address, which depends upon the configuration of both the digitizer and the computer. The configuration file specifies the LUTs (continuous tone, threshold, slice, and pseudocolor) that will be available at startup. For instance, if one is working with photographic negatives, it may be desirable to set the default continuous tone display as a negative ramp. The color of the graphics overlay and cursor can be specified as a mix of red, green, and blue intensities. The color of the slice highlighting can similarly be specified as a different mix of red, green, and blue intensities. The aspect ratio can be specified to allow exact calibration of measurements with respect to camera and digitizer distortion. Filenames for two external pseudocolor lookup tables can be specified. A transparency range (lower and upper threshold) can be specified for use in the mosaic routine. The current calibration coefficient and unit label are stored for use in all measurement routines. A mouse driver command (or other initialization command) can also be included to allow automatic loading of a mouse driver when *IMAGE* is started. When *IMAGE* is terminated, any changes in configuration values are saved.

OTHER PROCESSING

IMAGE provides a variety of utilities that facilitate processing and analysis. Image buffers can be entirely cleared of data values or graphics overlay, one can instantly display different image buffers, and one can copy images between buffers. Images can be saved to disk or left in the image buffer for access by other programs that have processing and analysis capabilities not included in *IMAGE*. For instance, *IMAGE* does not currently contain various common image processing capabilities such as convolutions, but such routines can readily be written by a user or obtained in many commercial software packages.

CHAPTER III. PROGRAMMING CONSIDERATIONS

CONTROL STRUCTURE DESIGN

Menu Structure

The primary concern in designing the control structure of an image analysis system is to provide an effective means for accessing the various necessary software capabilities. For the control structure of *IMAGE*, our philosophy has been to focus on the efficiency and precision of control. Toward this end, we chose a relatively flat menu structure, as opposed to a more intricate tree structure. A flat structure means that many capabilities are simultaneously accessible. In general, there are only two layers of menus to choose from: a main menu that includes major categories of capabilities available and submenus, for instance the trace menu, that include all of the specific choices related to a particular mode or set class of capabilities. Choices are made by typing a single keystroke mnemonic, except for buffered input, such as input of filenames. Considerable attention has been given to always make available a complete set of related capabilities that may be needed simultaneously, for instance the ability to load images and switch between image buffers while in the mosaic mode. Various essential capabilities are always available from within any mode, in particular the ability to switch between different display LUTs. This avoids the inefficiency and annoyance of having to leave the current menu, go back through a series of menus to enter a command, and then return to continue with an analysis. An attempt has been made to allow logical and efficient progression through the various steps involved in analysis, providing an integrated control structure.

Keyboard Versus Mouse Control

We choose to use the mouse only for cursor control on the image screen and some interactive graphics (e.g., the histogram integration routine) because we believe that keyboard control is more efficient for command choice. By using the mouse only for cursor control, we also avoid the need to use the mouse simultaneously for menu choice and cursor control, which can be confusing and inefficient. The learning curve for keyboard control may be slightly slower than for mouse control, but this is largely compensated by using mnemonics for the names of commands, providing menus and prompts that list all possible choices, and using interactive graphics routines where possible. An experienced user can more rapidly enter a series of keystrokes than a series of mouse choices. A mouse or other locator device is very effective for cursor control at a coarse level, but the keyboard arrow keys are preferable for precise pixel-by-pixel control. In *IMAGE*, cursor control can be accomplished simultaneously with the keyboard and a mouse. When used in conjunction with a display of X,Y coordinates, absolute precision in cursor placement is possible by using the mouse for most control, but using the keyboard for fine adjustments.

Dual Screen Systems

IMAGE uses a dual screen system, with one screen for image display and the other screen for menus, control information, and graphics. A dual screen system means that the image is always visible while commands are simultaneously being entered or results of analyses are being displayed. For instance, one can examine graphic representation of a histogram for a region of interest on the computer monitor, while simultaneously examining the location and appearance of the region of interest on the image monitor.

Interactive Graphics

Interactive graphics routines can often greatly facilitate the efficiency of control. Various interactive graphics have been incorporated in *IMAGE*. Cross-hair cursors are used to specify individual pixel locations. Rubber-band cursors are used to specify regions of different sizes and shapes. Intensity thresholds are set interactively while displaying effect using the threshold and slice LUTs. The number of pixels that lie within a range of data values is calculated with an interactive histogram integration routine. The LUT editor allows interactive specification of hardware LUTs. In all cases, the interactive graphics rapidly communicates complex feedback information to the user as the user specifies an operation, measurement, or configurational change.

Combining Capabilities for Efficiency

For efficiency and ease of operation, many of the routines in *IMAGE* have a combination of uses or effects. For instance, many of the measurement and editing routines are combined. In a more complex example, the mosaic mode combines capabilities to rescale regions of interest in the X and Y dimensions (both zoom and aspect ratio change capabilities), while allowing movement of regions of interest, and permitting specification of data value ranges that will not be copied from one region to another. Efficiency is attained by making a wide variety of capabilities simultaneously available in a flexible and intuitive control structure.

ACCESS TO DATA

One of the greatest strengths of *IMAGE* is the way in which appropriate data is made accessible. The user always has access to the coordinates and intensity values for pixels, as specified by cursors. Results of analyses are displayed on the screen and can optionally be output to ASCII data files. Such accessibility to data offers considerable power and flexibility.

DEDICATED VERSUS GENERAL-USE PROGRAMS

There is generally a compromise between providing dedicated versus general-use analysis capabilities. For example, *CANOPY©* is a program dedicated solely to interpretation of hemispherical canopy photographs, for the specialized task of estimating the penetration of light through openings in plant canopies. A full description of *CANOPY* is provided in Los Alamos National Laboratory Report LA-11733-M, "A Manual for Analysis of Hemispherical Canopy Photography," by Paul M. Rich (1989). In contrast, *IMAGE* provides a general set of image processing and analysis capabilities for making one- and two-dimensional spatial measurements. The highest level of efficiency for a particular scientific application is generally attained by designing a dedicated program, however the utility of such a program for other applications is generally very restricted. There will always be a need to provide customized capabilities for certain specialized scientific applications. The advantage of a program such as *IMAGE* is that it can be applied to a wide range of scientific applications, without major compromises in efficiency.

SOFTWARE PORTABILITY

There is often a compromise between providing software that can readily be adapted for different hardware and taking full advantage of particular hardware capabilities. In writing *IMAGE*, we have attempted to maintain a high degree of portability, allowing the transfer of capabilities to other computers and other hardware. *IMAGE* is written in the "C" programming language, which is readily transferred between different computers. Source code for *IMAGE* is currently contained in seven modules. All digitizer/display dependent routines, such as register initialization, are restricted to a single source code module. All other modules are largely hardware independent. Transfer of *IMAGE* to alternative hardware only requires development of source code for the low-level digitizer/display dependent routines, including register control, LUT control, and reading and writing pixel values. In addition, some hardware may require minor changes in other modules, for instance changes in macros that specify the X and Y resolution of the display. In many cases, *IMAGE* can be adapted for alternative hardware in the matter of only a few days of programming. *IMAGE* requires the use of hardware display lookup tables to take full advantage of many capabilities.

TRUE COLOR IMAGING

Though *IMAGE* can only input one video channel at a time, it is possible to do true color imaging. True color imaging involves 1) digitizing separate red, green, and blue frames for a given image, either as the separate channels of an RGB signal or using filters; 2) combining the three frames so that red, green, and blue are encoded in a single image; and 3) setting up appropriate user-specified LUTs to display the true color images. The RGB source may be from RGB output cameras, or may be the decoded output of an NTSC compatible color camera. For still images, each color frame can be acquired sequentially by manually switching color channel input. There are digitizers and display subsystems for true color that use 8 bits per color channel, however display memories currently supported by *IMAGE* use only 8 bits per pixel. Color compression algorithms have been developed which allow 24 bits of color information to be compressed into 8 bits of display memory (using an appropriate output lookup table) with little discernible loss of image quality, although these are capabilities not now incorporated in *IMAGE*.

SOFTWARE LIBRARIES AND COMMAND LINE PROGRAMS

Individual software routines are best written in a way such that they can easily be incorporated in customized image analysis programs. A library of useful routines can be established that contains both low-level digitizer-dependent routines and high-level processing and analysis capabilities. Such a library can be expanded as more capabilities are developed. Also, small command line programs are often useful to accomplish simple tasks, such as to initialize the digitizer, capture images, save and load images, and specify different display LUTs. Use of such command line programs avoids the need to enter a larger program such as *IMAGE* every time one wants to accomplish a simple task. *IMAGE* is written as a series of discrete "C" functions that constitute a library of microcomputer image analysis capabilities. These functions can readily be incorporated in other custom software. Provided with *IMAGE* is a series of command line utility programs for image processing and analysis (see Appendix D Listing of Files and Utility Programs).

CHAPTER IV. THE FUTURE OF MICROCOMPUTER IMAGE ANALYSIS

HARDWARE ADVANCES

Technological advances are proceeding at a rapid rate, and will have a major influence on the future of microcomputer image analysis. Advances in microcomputer processing speed, graphics displays, and storage capabilities will enable increasingly sophisticated and rapid image analysis. Solid-state video cameras, in both RS-170 and non-standard formats will offer improved means for image input. Laser disc storage technology will enable rapid storage and retrieval of large numbers of images in databases, in either analog or digital formats. Digitizer and display adapters have increasing hardware capabilities that enable real-time processing. Of considerable importance will be add-on hardware adapters that enable specialized processing capabilities, including capabilities for parallel processing and three-dimensional imaging.

Microcomputer Technology

Advances in microprocessor technology will undoubtedly continue to produce faster, more powerful machines. The 32-bit architecture employed in IBM-compatible 80386 machines and the Apple Macintosh II series will probably remain the standard longer than have 8- and 16-bit machines. Thirty-two-bit processors are adequate for a broad range of applications; 64-bit mainframes are now common only for high-resolution scientific supercomputing. One welcome enhancement will be the emerging availability of operating systems with extended memory addressing, relaxing the 640K-RAM limit built into DOS. The 16-bit AT bus structure will remain viable for the near future, because of a large installed base and because of the many compatible hardware adaptors available for instrumentation. On 32-bit machines, this basic architecture is often enhanced by the addition of high-speed cache memories and a local bus between memory and processor boards. The microchannel architecture of the IBM PS/2, as well as VME, Q-BUS, and NUBUS, will gain increasing utility as designers build video subsystems implemented with little change across a variety of standard bus structures.

The need for high resolution displays for computer aided design and for "windowed" consoles has driven the development of display technology with 1024 X 1024 or greater pixel resolution at moderate prices. Image memories of one megabyte or more are now commonplace and reasonably inexpensive, because of the development of higher density memory chips with lower power consumption, and driven by the increasing complexity of application software and operating systems. While existing microcomputer mass storage systems are not entirely satisfactory for video processing, current trends in the development of nonmagnetic storage media, such as optical disks, promise to deliver sufficient storage density. Advanced information transfer strategies currently used in specialized storage devices will eventually permit data access and transfer at video rates (i.e., 10-50 million pixels/second). Such developments, combined with advances in video and analog electronic technology, will undoubtedly produce more powerful and more useful systems.

Video Technology

Digital video technology has relaxed the need for sensors and displays to adhere to a standard video format, and the increasing utility of machine vision systems has produced a market for specialized sensors. Sensors consisting of 512 X 512 or 1024 X 1024 arrays have become more common. Cameras exist for applications where higher acquisition rates are required and lower resolution is acceptable. High-performance scientific CCD cameras have been available for several years; however moderately priced cooled CID and RS-170 CCD systems have only recently entered the market. Components for intensified video systems, which allow low light level imaging, have been available in the past; but now improved devices and assembled systems are commercially available. The existence of such high-performance intensified video systems at moderate cost will undoubtedly lead to wider scientific application and impact.

The increasing availability and falling prices of high-resolution video display equipment will reduce the motivation to adhere to RS-170-compatible video timing, with its attendant compromises. A first step is to adopt a 60-Hz noninterlaced display format, which would allow horizontal lines, a single pixel high, to be displayed without flicker. For applications requiring greater horizontal or vertical resolution, it is necessary to increase the scan rate or the bandwidth of video electronics. Multisync monitors, with analog RGB input, ensure maximum flexibility to accommodate a range of display resolutions. Such displays have been popularized as system monitors because they can be used with display adaptors ranging from IBM CGA to VGA standards. These devices often use multipin inputs for RGB; but adaptor cables are available that allow component input from separate Berkeley Nucleonics

Corporation (BNC) cables. This feature will undoubtedly be incorporated in future multisync monitors as demand is established.

D/A Conversion

D/A conversion technology used to drive video displays may be considered mature. Fast 8-bit D/A flash digitizers are readily available; higher-resolution devices are not necessary because existing devices exceed perceptual limits. However, as sensor signal-to-noise ratios improve, there will be increasing need for high resolution analog-to-digital (A/D) converters. High-sensitivity photodiode arrays have been used to detect low-intensity optical signals, for example to detect changes in indicator dyes in biological systems that range from a few percent to a few parts per hundred thousand. In order to achieve this sensitivity with video, improved camera electronics, amplifiers, and A/D systems will be required. Digital systems can serve as buffers, relaxing perceptual requirements for image comprehension. For example, a noninterlaced camera operating at 30 Hz produces a bothersome flicker if displayed directly; this can be eliminated by routing the signal through a dual-ported frame memory that allows independent image acquisition and display operations to proceed simultaneously. A non-interlaced display eliminates some problems associated with dynamic processes. High-resolution images produced by linear sensors or slow scan cameras may be generated too slowly to perceive as an image, but these are readily accumulated in a digital memory. The importance of alternative input formats has been acknowledged by several framegrabber manufacturers including Imaging Technology, which recently introduced a variable scan version of the FG-100. Recently, 12-bit digitizers capable of operating at 20 megasamples per second have been introduced; however, as previously noted, A/D systems may not be the major obstacle to achieving such a large dynamic range. Because of such concerns, many high resolution cameras may incorporate onboard digitizers and produce parallel digital output signals.

Dedicated Image Processing Hardware

Another trend that will become increasingly important is the development of self-contained dedicated subsystems for high-performance image processing. Several manufacturers have marketed high-end systems that employ a microcomputer as a host for mass storage and user interface for a hardware image processing system. Such systems typically incorporate a high-speed internal bus (separate from the host bus), and many employ "pipelined" hardware processors dedicated to specific processing tasks. Programmable logic arrays allow a single hardware device to be programmed for a variety of applications. For even more flexibility, a general purpose array processor can be coupled to the video subsystem to enhance processing speed. In the future, parallel devices, such as transputers, will be used to endow the local subsystems with even greater intelligence. Relatively inexpensive subsystems are now available which incorporate some of these design features. For example, the FG-100 is an AT-resident, one-board system with an arithmetic unit capable of certain LUT-driven pipeline processing. Recently an AT-compatible external subsystem was introduced with modular pipeline processing features and a local VME bus and AT bus translator. This design allows the system to function as an extension of the AT bus, but without performance-limiting design compromises, and may serve as a model for future inexpensive, high-performance image processing subsystems.

SOFTWARE ADVANCES

Microcomputer image analysis will continue to be limited by the availability of suitable scientific software. This is part of a general trend in the computer industry, in which hardware advances outpace software development. Only recently has it been realized that a more organized approach to software development is essential. Attention must be given to developing software that is portable and flexible. Software must be written in a modular way, such that capabilities are extensible to new hardware and such that these capabilities can be combined in unique ways to address particular problems. Automation will be necessary for many applications, and will require sophisticated processing for pattern recognition and decision making. An image analysis language is needed that is specific for scientific image analysis design. Such a fourth-generation language could be used to construct customized image analysis software using libraries, menu generators, and data output specifiers, and would be simple to learn and use. An image analysis language would allow scientists to design programs customized for specific needs.

IMAGE is presented as an example to inspire programmers to develop even more powerful and efficient microcomputer image analysis programs. Various capabilities of *IMAGE* can be made more complete, including addition of standard image processing capabilities, such as convolution, rotation, and Fourier transforms. A continuing need exists for effective pattern recognition and classification of images. Complete support of true color imaging and development of capabilities for three-dimensional imaging are also high priorities. As with many other types of scientific software, there is a need for integration of image analysis software with spreadsheets, databases, statistical analysis, and graphics display and output. Output of data to ASCII files is the first step, but true integration requires that data be written directly in a spreadsheet and capabilities be provided for data manipulation, statistical analysis, and graphics output. Databases systems are needed that relate images to text and other data, including integration of image analysis capabilities with geographic information systems.

CONCLUSION

Microcomputer image analysis is coming of age. Already many powerful image analysis capabilities are available for scientific applications. Many scientific problems have been difficult or unapproachable in the past because of the difficulty of analyzing two- and three-dimensional data. Image processing systems were previously available only on expensive mainframe computers. Hardware and software advances will enable increasingly sophisticated image processing and analysis on microcomputers, with capabilities that were not even possible on mainframes in the past. *IMAGE* provides many capabilities useful for scientific applications. The design of *IMAGE* can be used as a model for development of future microcomputer image analysis software. Future microcomputer image analysis packages ideally would include a complete set of image processing capabilities, be fully integrated at all levels of analysis and output, incorporate a flexible and efficient control structure, and be available at modest expense.

CHAPTER V. GUIDE TO OPERATION OF *IMAGE*

HARDWARE SETUP

The basic hardware setup involves 1) configuring the digitizer/display image buffer and register I/O addresses, 2) installing the video digitizer display card in the microcomputer, and 3) connecting of cables from the digitizer to the image display and from the video camera to the video digitizer. Refer to the digitizer/display manual for details of this setup. In general, the image buffer should be located at address D0000 hexadecimal and the register I/O address should be located at the factory default. During installation it is important to determine whether memory conflicts may arise between the digitizer/display and any other devices attached to the computer. The digitizer/display adapter must be installed in an expansion slot within the microcomputer. The red, green, and blue outputs from the digitizer card should be connected to the red, green, and blue inputs, respectively, of the RGB display monitor. For display synchronization, it is suggested that a separate cable be connected from the digitizer synchronization output to the display external synchronization input (cable optional with the PC Vision Plus). It is also possible to run a synchronization signal from the green output. The camera output should be connected to the digitizer input via a cable with appropriate BNC connectors and adapters. See Appendix C for hardware specifications.

SOFTWARE SETUP

Creation of Program Directory and Copying Files

As with most software, make a back-up copy of all files before installation. Then create a directory on the hard drive for *IMAGE* (e.g., *C:\IP\IMAGE*). Copy the following files into the *IMAGE* directory: *IMAGE.EXE*, *IMAGE.CNF*, *P1.LUT*, *P2.LUT*. A customized batch file (e.g., *RUNIM.BAT*) can be created to set up *PATH* and *APPEND* commands (to allow access to the *IMAGE* director from other directories), and to change to the appropriate directory when starting (see DOS manual). Note, *APPEND* is available only for DOS Version 3.3 and higher. The following is an example of the contents of a custom startup batch file:

```
PATH C:\;C:\IP\IMAGE;C:\DOS;C:\BAT;
APPEND C:\IP\IMAGE;
C:
CD \IP\DATA
```

The startup batch file could be stored in the directory *C:\BAT* along with other useful batch files. To use *IMAGE* with versions of DOS previous to 3.3, it may be desirable to run *IMAGE* from the *C:\IP\IMAGE* subdirectory and transfer image and data files to other directories at a later time.

A custom initialization batch file can be created to automatically run a series of commands each time the program IMAGE is started. For example, resident drivers for a mouse or other locator device can be loaded this way. The configuration file *IMAGE.CNF* includes the filename of the initialization batch file. The initialization batch file should be included in the program directory for *IMAGE*. The following is an example of an initialization batch file:

```
MOUSE
CPANEL
```

This initialization batch file loads the mouse driver (*MOUSE.COM*) and resident mouse control panel program (*CPANEL.COM*).

To Start *IMAGE*

1) Run a customized batch file if one has been set up. Otherwise, type the appropriate *PATH* command, etc.
2) Type IMAGE to start *IMAGE*. A credit/copyright screen will be displayed and you will be prompted to press any key to continue. After you press any key, the main menu screen will appear. When you start *IMAGE*, type -r on the command line to prevent resetting and clearing of the digitizer, especially if you wish to continue working on an image that is already loaded into the digitizer memory, for example,

C:\IP\DATA>**IMAGE -r<ENTER>**

After starting the computer with a cold boot, *IMAGE* should be run without the **-r** option, to initialize the digitizer/display registers.

3) Select commands from the keyboard with single keystrokes as indicated in the menu. In general, escape **(<ESC>)** will end or abort a process at any stage.

The first time *IMAGE* is run on a newly configured system, the configuration file should be modified to include the correct image buffer and register input/output (I/O) addresses. Starting the program with incorrect addresses will probably freeze the system and necessitate turning off the power and then rebooting. The configuration can be modified by following these steps:

1) Start *IMAGE* with the -r option to avoid resetting the digitizer.
2) Go from the credit screen to the main menu screen by pressing any key. Select **<Alt-C>** (*CONFIGURATION*).
3) Select choices 0 and 1 sequentially and enter the appropriate image buffer and register I/O addresses. Press **<ESC>** to exit the configuration menu. *IMAGE* should now run correctly.

Experienced computer users can simply edit the configuration file, referring to the file format in Appendix D.

MENU STRUCTURE

IMAGE uses a menu structure that allows a relatively large number of commands to be available from a single main menu. Commands are generally entered by typing a single keystroke. A simple mnemonic corresponds to each keyboard choice. The submenus include a configuration menu, menus for the trace and edge routines, and a mosaic menu. Escape **<ESC>** is generally used to exit a mode or abort an operation. Various prompts, lists of choices, messages, and control information will appear at appropriate times.

CURSOR CONTROL

Cursor control on the image screen can be accomplished with the mouse or arrow keys on the keyboard. The mouse can be used for coarse control and the keyboard for fine pixel-by-pixel control. In general, with the mouse, movement up/down and right/left moves the cursor accordingly; the right button combined with movement of the mouse stretches the cursor to different shapes; and the left button completes an action (e.g., draws a filled rectangle when in filled rectangel mode). On the keyboard, the arrow keys move the cursor one pixel in the corresponding direction; **<Ctrl-Left>** and **<Ctrl-Right>** move the cursor ten pixels to the left or right, respectively; and **<PgUp>** and **<PgDn>** move the cursor ten pixels up or down, respectively. Slash / toggles between this cursor movement mode and a cursor stretch mode in which the arrow keys**, <Ctrl-Left>**, **<Ctrl-Right>**, **<PgUp>**, and **<PgDn>** allow the cursor to be stretched. The **<ENTER>** key completes an action, analogous to the left button on the mouse.

CONTROL INFORMATION

The menu name is always displayed at the top of the computer monitor, the current active set of LUTs (continuous, threshold, slice, or pseudocolor) and the active mode (e.g., trace, histogram, etc.) are displayed on the bottom line. The second line from the bottom displays the currently active image buffer and the name of the image, if the image has been saved or loaded. Additional control information is shown in the main menu, including whether positive or negative LUTs are currently in use and the current active value, which specifies the intensity value used for editing. When the threshold LUT is used, the threshold value is displayed at the bottom of the screen; and when the slice LUT is used, the upper and lower thresholds of the slice range are displayed. When cursors are used a display appears that shows appropriate cursor coordinates and data values corresponding to pixels at those coordinates. For a cross-hair cursor, the coordinates and data value for the pixel are at the center of the cross. For a line cursor, the coordinates and data values are at the endpoints. For a rectangular cursor, the coordinates and data values are at two diagonal corners. For a circular cursor, the coordinates and data value are at the center of the circle. When in a measurement mode (e.g., flood mode) control information appears that summarizes the measurements that have been made, including conversion to real units if unit calibration has been performed.

CHAPTER VI. REFERENCE GUIDE TO *IMAGE* CAPABILITIES

The program *IMAGE* provides general-purpose image analysis capabilities suitable for a range of scientific applications. These capabilities include image digitization, storage, and display; image rectification; and quantitative analysis. Control of *IMAGE* is accomplished using a series of single keystroke commands, each of which is mnemonic for the action to be performed. The following is a complete alphabetical listing of mnemonic commands. Numeric, alternate-key, and control-key commands follow the alphabetical listing.

A ARCHIVE

Purpose

Load or save an image file, or provide a *DOS* shell.

Mode of Operation

A menu will pop up to allow choice between loading an image, saving an image, and entering a *DOS* command shell.

L loads an image file.
S saves an image file.
D enters a *DOS* command shell.

After pressing **L** or **S**, enter a valid filename. Type **<ESC>** to terminate the command choice at any point. After loading or saving an image, the filename is displayed in the lower left portion of the control screen. Files are always loaded from and saved to the current image buffer. When loading an image, any image in the current image buffer will be destroyed. When saving an image, no protection is currently included to prevent overwriting files with the same name. Images are stored in binary format with no header (see Appendix D for Image File Format). After entering **D**, any *DOS* command can be entered. For instance this feature can be used to view listings of files, change directories, make directories, copy files, and delete files.

B BUFFER

Purpose

Toggle between image buffers. (Not available for digitizer/display adapters with only one image buffer.)

Mode of Operation

The current image buffer is displayed in the lower-left portion of the control screen, before the current filename, and also displayed after the **BUFFER** menu choice. Image I/O and changes occur in the currently active image buffer. The number of available image buffers depends upon the digitizer/display adapter being used.

C CIRCLE MODE

Purpose

Draw filled or unfilled circles.

Mode of Operation

First, a menu will pop up to allow choice between filled circle modes:

1 enters a mode to draw filled circles.
2 enters a mode to draw unfilled circles.

A rubber-band circle cursor shows on the image screen. The coordinates and data value of the center point, and the radius are shown on the control screen. Press **<ENTER>** or the left mouse button draws the circle on the image using the current active value. Press **<ESC>** to exit circle mode.

D DIGITIZE

Purpose

Toggle digitization on and off.

Mode of Operation:

A standard RS-170 black and white video signal is converted into a digital image. Digitization proceeds at the rate of one frame each 1/30 second. Images are displayed on the image display monitor, with a 1/30 second delay between digitization and display. The video camera must be on and properly attached to the video input connector for digitization to proceed. Though 8 bits of intensity information per pixel are initially digitized, only 7 bits are stored with the use of a linear ramp LUT. Digitization is stopped by pressing **D** again.

E EDGE MODE

Purpose

Automatically find and trace edges of objects, as defined by a range of intensity values. For each object the object number, perimeter length, bounded area, and centroid are calculated. Edge coordinates and measurements can be saved and loaded to disk. Also, the values of all boundary pixels can be changed to a specified value (the active value).

Mode of Operation

A cross-hair cursor is displayed on the image screen and the edge menu appears on the control screen. The X,Y coordinates and data value of the cursor location are shown on the control screen. Pressing the left button of the mouse or **<ENTER>** initiates a search for the edge of an object and subsequent following and non-destructive tracing of the object boundary. Search for an object edge occurs to the right of the cursor if the cursor is outside an object and to the left of the cursor if the cursor is inside an object. Objects can be previewed using the slice LUT (choice **3**). An object is defined as a group of adjacent pixels that are included within the range defined by the low and high data value thresholds of the slice LUT (see explanation under **3, SLICE LUT,** at the end of this chapter). The boundary trace is drawn just on the inside of an object. After each boundary search, values are displayed for the total number of objects, the length (perimeter) of the current object boundary trace, the area of the current object, the centroid X,Y coordinates for the current object, the cumulative length of all boundaries, and the cumulative area of all objects. Values are shown in pixels, and in actual length if unit calibration has been performed. Pressing the right mouse button erases the boundary trace and resets the length counter to zero. The following commands can be entered while in edge mode:

- **0** causes the length counter to be reset to zero.
- **E** erases all boundary traces.
- **S** saves all X,Y coordinates of the boundary to an ASCII file.
- **L** loads the X,Y coordinates of previously saved boundary from an ASCII file (see Appendix D for Edge Coordinate File Format).
- **O** outputs the results of all boundary searches (object number, length, area, and centroid X,Y coordinates) to an ASCII file (see Appendix D for Edge Result File Format). When saving coordinate files or result files, the user is prompted as to whether output should be in device coordinates and units (choice **1**) or calibrated coordinates and units (choice **2**).
- **<Ctrl-D>** draws the boundary trace permanently to the image using the current active value.
- **A** automatically scans the image and locates all objects.
- **<ESC>** exits edge mode and erases all boundary traces.

The following general capabilities are also available from the edge menu: **V** active value, **1** continuous LUT, **2** threshold LUT, **3** slice LUT, **4** pseudocolor LUT, **<ALT-1>** toggle continuous LUT NEG/POS, **<ALT-2>** toggle threshold LUT NEG/POS, **<ALT-3>** toggle slice LUT NEG/POS , **<ALT-4>** toggle pseudocolor LUT ONE/TWO.

F FLOOD MODE

Purpose

Locates the boundaries of an object, as defined by a range of intensity values, and changes all object pixel values to a specified value (the active value).

Mode of Operation

A cross-hair cursor shows on the image screen. The coordinates and data value of the cursor location, X,Y, are shown on the control screen. The cursor can be moved to any location, or seed point, where a flood is initiated. Pressing the left button of the mouse or **<ENTER>** initiates a flood of any object under the cursor object, searching out the boundaries of an object, changing all pixel values to the current active value, and calculating the area of the object. Objects can be previewed using the slice LUT (choice **3**). An object is defined as a group of adjacent pixels that are included within the range defined by the low and high data value thresholds of the slice LUT (see explanation under **3, SLICE LUT**). After each flood operation, values are displayed for the total number of objects, the area of the current object, and the cumulative area of all objects. Values are shown in pixels, and in actual length if unit calibration has been performed.

- **0** or right mouse button causes the area counter to be reset to zero.
- **E** erases the graphics floods (when the active value is the graphics plane, see under V, ACTIVE VALUE, in this chapter).
- **O** outputs the results of all flood operations (object number, coordinates of seed points, and areas) to an ASCII file (see Appendix D for Flood File Format). When saving result files, the user is prompted as to whether output should be in device units (choice **1**) or calibrated coordinates and units (choice **2**).
- **<ESC>** exits flood mode. If exiting while the active value is the graphics plane, a prompt will appear asking if the graphic plane should be cleared. Enter **Y** for yes or **N** for no after the prompt.

H HISTOGRAM MODE

Purpose

Calculate and display the histogram of intensity values for all pixels within a user-specified region of interest. Within a region of interest, the number of pixels and area for any range of data values can be calculated.

Mode of Operation

A rubber-band rectangle cursor shows on the image screen. This rectangle defines the region of interest for which the histogram will be calculated, inclusive of all pixels under the rectangular cursor itself. The coordinates and data values for the upper left and lower right corners (X1,Y1 and X2,Y2) are shown on the control screen. Pressing the **<ENTER>** key or the left mouse button starts calculation of a histogram of data values for the defined region of interest. If the threshold LUT (choice **2**) is in use, the number of pixels is displayed for white, black, and total pixels within the region of interest. The actual area is displayed for white, black, and total area within the region of interest if unit calibration has been performed. If any other LUT is being used (choices **1**, **2**, or **4**), a plot of the histogram is shown on the control monitor. The Y-axis of the plot is scaled to the number of pixels for data value with the largest number of pixels. The following commands are available while a histogram plot is being displayed:

- **Y** allows rescaling to any user specified maximum Y. The user is prompted for the maximum number of pixels to be shown on the Y axis. After manual rescaling of data values, any categories that go off the scale are highlighted in a different color from the rest of the plot.
- **A** "autoscales" the Y axis back to the number of pixels for the data value with the largest number of pixels.
- **I** allows integration of the total number of pixels for a range of data values. If the unit calibration has been performed, the actual area of the pixels is displayed. The cursor at the base of the plot defines the range to be integrated, and can be moved and stretched, just as other line-type cursors. If the slice LUT is on (choice 3), the slice range defines the low and high

H HISTOGRAM MODE (continued)

data value thresholds and pixels within the range are highlighted on the image screen.
<ESC> exits back to histogram mode while the histogram plot is being displayed.

The **<ESC>** key also exits histogram mode.

L LINE/MEASURE MODE

Purpose:
Draw lines and calculate linear distance.

Mode of Operation
A rubber-band line cursor shows on the image screen. The coordinates and data values of the endpoints (X1,Y1 and X2,Y2) are shown on the control screen. The length of the line is shown in pixels, and in actual length if unit calibration has been performed. Press **<ENTER>** or the left mouse button to draw the line on the image using the current active value. Press **<ESC>** to exit line mode.

M MOSAIC MODE

Purpose
Copy any region within an image to any other region, either within the same image buffer or between image buffers. While copying, the region can be rescaled in both X and Y dimensions to change size and shape. Images with irregular edges can be fitted together in a jig-saw puzzle or mosaic fashion. (Not available for digitizer/display adapters with only one image buffer.)

Mode of Operation
The mosaic menu appears. The X and Y dimensions are scaled independently, such that a mosaic operation can change the size and aspect ratio of any region of interest, while at the same time copying that region to any region in either image buffer. The X and Y scale factors and the ratio of change in aspect ratio are displayed on the control screen. The target region can overlap entirely or in part with the source region. A range of data values can be specified such that data values within the range of the source region of interest will not be written on top of data values of the target region (e.g., if 0 is transparent, any pixel in the source region of interest with a value of 0 will not be written to the target region of interest). The following commands are available while in mosaic mode:

T allows selection of a new range of transparency values (allowable range 0 to 127). The default transparency range is stored in the configuration file. The user is prompted for a lower and upper limit of transparency. Entering **U** after the lower limit prompt causes the transparency to be undefined (all values to be written).
R selects the source and target regions of interest. First, the user is prompted for the source region of interest and then the target region of interest. The user can enter the number of regions of interest from the source and target lists (range 1 to 12) or by pressing **L** for the locator device (mouse). The active source and target region are shown on the left center of the control screen.
E allows editing of the list of sources and targets. The user will be prompted for the number of the region of interest, the source buffer (**1** or **2**), the source region coordinates as selected with a rectangle cursor, the target buffer (**1** or **2**), and the target region coordinates as selected with a rectangle cursor. Both source and target regions include the pixels under the rectangle cursor.
I initiates input of a list file that has been stored previously (see Appendix D for Mosaic List File Format). The user is prompted for an input filename.
O outputs a list file. The user is prompted for an output filename. **B** toggles between image buffer ONE and TWO. **A** loads or saves images.
<ENTER> initiates the mosaic operation. The user is prompted as to whether to proceed to mosaic. Press **Y** for yes to complete the process, and the source region is copied to the target region.

M MOSAIC MODE (continued)

<ESC> aborts a command at any stage.

The **<ESC>** key also exits mosaic mode.

O OPERATIONS MODE

Purpose

Performs logical and arithmetic operations between scalars and images, and between whole images. (Not available for digitizer/display adapters with only one image buffer.)

Mode of Operation

A menu pops up to allow choice between single image operations, multiple image arithmetic operations, and multiple image logical operations:

- **1** performs a single image operation.
- **2** performs a multiple image arithmetic operation.
- **3** performs a multiple image logical operation.

Under single image operations, either a linear transformation (linear contrast stretch) of data values is performed or the image is bit-shifted:

- **1** performs a linear transform. For a linear transform, the user is prompted for a slope and an intercept. Calculated values less than zero are set to zero and calculated values greater than 127 are set to 127.
- **2** performs a bit-shift operation. The bit-shift operation shifts the data values for all pixels one bit to the right or left. The user is prompted as to whether to bit-shift to the right (**R**) or left (**L**).

Under multiple image arithmetic operations, subtraction, addition, division, and multiplication are available:

- **1** subtracts two images.
- **2** adds two images.
- **3** divides two images.
- **4** multiplies two images.

In each case, the operation is performed on a pixel-by-pixel basis, in the order Buffer ONE operated on by Buffer TWO, with the results placed in Buffer TWO. For example, in subtraction Buffer TWO is subtracted from Buffer ONE and the result is placed in Buffer TWO. Again, calculated values less than zero are set to zero and calculated values greater than 127 are set to 127.

Similarly, under multiple image logical operations, the logical operations AND, OR, and XOR (exclusive or) are performed on a pixel-by-pixel basis for Buffers ONE and TWO and the result is placed in Buffer TWO:

- **1** AND two images.
- **2** OR two images.
- **3** XOR two images.

The image in Buffer TWO should be saved before performing a multiple image operation. Press **<ESC>** to abort operations at any stage before choice of the specific operation.

R RECTANGLE MODE

Purpose

Draw filled and unfilled rectangles.

R RECTANGLE MODE (continued)

Mode of Operation

A menu pops up to allow choice between filled and unfilled rectangle modes:

- **1** enters a mode to draw filled circles.
- **2** enters a mode to draw unfilled circles.

A rubber-band rectangle cursor shows on the image screen. The coordinates and data values for the upper left and lower right corners (X1,Y1 and X2,Y2) are shown on the control screen. Press **<ENTER>** or the left mouse button to draw the rectangle on the image using the current active value. Press **<ESC>** to exit rectangle mode.

T TRACE MODE

Purpose

Traces curves in graphics overlay superimposed on an image. For each curve traced, the curve number, curve length, and bounded area are calculated. Curve coordinates and measurements can be saved and loaded to disk. Optionally, data values for all pixels under a curve can be output. Also, the traced curves can be written permanently to an image, changing the intensity values all pixels in the curve to a specified value (the active value).

Mode of Operation

A cross-hair cursor shows on the image screen and the trace menu appears on the control screen. The coordinates and data value of the cursor location, X,Y, are shown on the control screen. Holding the left button of the mouse down causes tracing to be shown non-destructively on the image. The number of traces (separate sets of connected points), the length of the current trace, and the cumulative length of all traces is shown. Lengths are shown in pixels, and in actual length if unit calibration has been performed. The following commands are available while in trace mode:

- **0** causes the length counter to be reset to zero.
- **E** erases all traces.
- **N** starts a new trace segment, when connecting segments from the keyboard.
- **<ENTER>** or the right mouse button draws a trace segment from the current cursor location back to the endpoint of the last trace drawn.
- **A** causes the two endpoints of the current trace to be connected in a closed polygon and the area and cumulative area to be displayed. Areas for traces that cross over themselves will not be calculated correctly.
- **S** saves the X,Y coordinates of inflection points of the trace to an ASCII file.
- **L** loads the X,Y coordinates of previously saved trace from an ASCII file (see Appendix D for Trace Coordinate File Format).
- **O** outputs the results of all boundary searches (object number, length, and area) to an ASCII file (see Appendix D for Trace Coordinate File Format).
- **I** outputs an intensity file, which contains a listing of intensity values of all pixels along traces, in the order in which tracing was performed (see Appendix D for Intensity File Format). When saving coordinate files, result files, or intensity files, the user is prompted as to whether output should be in device coordinates and units (choice **1**) or calibrated coordinates and units (choice **2**).
- **<Ctrl-D>** draws the trace permanently to the image using the current active value.
- **<ESC>** exits trace mode and erases all traces.

U UNIT CALIBRATION MODE

Purpose

Calibrate distance measurements by allowing specification of the number of units for a given distance and a unit label. After unit calibration, line, edge, histogram, flood, and trace modes will show measurements in the specified units.

U UNIT CALIBRATION MODE (continued)

Mode of Operation:

First a calibration length is entered. A rubber-band line cursor shows on the image screen. The coordinates and data values of the endpoints (X1,Y1 and X2,Y2) are shown on the control screen. The length of the line is shown in pixels. Press **<ENTER>** or the left mouse button enters the line length. Next the user is prompted for the number of units corresponding to the line length--any decimal number can be entered. Then the user is prompted for the unit label, up to 20 characters long. Press **<ESC>** to abort unit calibration mode at any stage. Once unit calibration has been performed, any measurements (one- or two-dimensional) will be expressed in the calibrated units. Entering zero for the number of units corresponding to the line length will remove the calibration, such that all calculations are made in only device coordinates. The most recently used unit calibration is stored in the configuration file, and remains in effect when the program *IMAGE* is started in the future.

V ACTIVE VALUE

Purpose

Changes the current active value. The active value is used for all editing operations--line, rectangle, circle, trace, flood, and edge.

Mode of Operation

At the prompt, the user enters a new active value. The active value must be in the range 0 to 127, with 0 as black, 127 as white, and the values in between corresponding to different intensities of gray. If a value outside the range is specified, the active value remains unchanged. Entering a **G** after the prompt will set the active value to the graphics plane (GRA), such that all changes will be written to the graphics plane without changing data within the image data. The active value is displayed after the Active Value menu choice (**V**). The active value defines the value that will be used for any editing operations. The active value can be set from within any mode that uses the active value.

<ALT-C> CONFIGURATION MODE

Purpose

Changes the default configuration for the digitizer in/out address, the image buffer address, the filenames for the two pseudocolor lookup tables, the color of the cursor/graphics overlay, the color of the slice highlighting, the transparency range, the command used for taking directories, and an initialization command for the mouse or other locator device, and the aspect ratio.

Mode of Operation

Configuration changes are written to a configuration file, so changes will remain in effect when the program is restarted (see Appendix D for Configuration File Format). The following commands are available while in configuration mode:

- **0** allows specification of the digitizer/display/I/O address. An incorrect address will generally cause the system to freeze. The I/O address is set in hardware using a series of jumpers (see digitizer/display adapter manual for details).
- **1** allows specification of the image buffer address. The image buffer address is set in hardware using a series of jumpers (see digitizer/display adapter manual for details).
- **2** allows specification of a pseudocolor LUT file to be used for pseudocolor LUT ONE.
- **3** allows specification of a pseudocolor LUT file to be used for pseudocolor LUT TWO.
- **4** allows specification of the cursor/graphics overlay color. The color is specified by a series of intensity values (range 0 to 255) for each of the red, green, and blue channels.
- **5** allows specification of the slice highlight color, also specified by a series of intensity values (range 0 to 255) for each of the red, green, and blue channels.

- **6** allows specification of the transparency range (allowable range 0 to 127), the range of intensity values that will not be copied in a mosaic routine (see **M**, MOSAIC, earlier in this chapter). The user is prompted for a lower and upper limit of transparency. Enter **U** after the lower limit prompt to cause the transparency to be undefined (all values to be written).
- **7** allows specification of an initialization command that is automatically run when *IMAGE* starts up. This command can be any executable file or batch file.
- **8** allows specification of the aspect ratio of individual pixels (vertical over horizontal dimensions), for fine tuning measurements to correct for distortion in the camera or digitizer. Without such distortion, the aspect ratio should be 0.833333.
- **<ESC>** exits the configuration mode.

<ALT-L> LUT UTILITIES

Purpose

Edit hardware LUTs, and input and output of user-specifiable external LUT files. These routines can be used to modify pseudocolor display, as well as to produce and save any possible display LUT.

Mode of Operation

The user can load, save, or edit either of the two pseudocolor LUTs. The contents of the pseudocolor LUTs can be any user-specifiable LUT. After entering **<Alt-L>,** the default pseudocolor LUT is automatically displayed. **<Alt-4>** toggles between pseudocolor LUT ONE and pseudocolor LUT TWO (see **<Alt-4>,** PSEUDOCOLOR LUT ONE/TWO, at the end of this chapter). The LUT load, save, and editor capabilities all operate on the currently active pseudocolor LUT. From the LUT utility menu, the following commands are available:

- **L** loads a LUT file.
- **S** saves a LUT file.
- **E** enters the LUT editor.
- **<ESC>** exits the LUT utilities menu.

The LUT editor is an interactive routine that shows an X,Y plot, with data values on the X axis and display values on the Y axis. The top of the screen shows the current channel being edited (red, green, blue, or all), the data and display values at the current cursor location, and whether editing is being performed in the data or graphics planes (graphic or data). The following commands can be entered while in the LUT editor:

- **R** allows editing of the red channel.
- **G** allows editing of the green channel.
- **B** allows editing of the blue channel.
- **A** allows editing of all channels simultaneously.
- **<ALT-G>** toggles between editing the data values without the graphics bit on (data values 0 to 127) and data values with the graphics bit on (data values 128 to 255).*
- **<Alt-4>** toggles between the pseudocolor LUTs ONE and TWO.
- **S** sets a “stretch” endpoint to draw a linear ramp.
- **F** toggles between a cursor follow mode and a free mode.
- **<ESC>** exits the LUT editor.

In general, the user specifies the display value for each given data value, either for the red, green, and blue channels separately to produce pseudocolor, or for all channels to produce shades of gray. This is accomplished by moving a cursor to draw the shape of the LUT function, using either the mouse or keyboard. Pressing the left

* To maintain consistency of operation, for the Data Translation *DT2853*, the LUT editor acts as though the most significant bit is used for graphics overlay, just as it actually is for the Imaging Technology digitizer/display adapters. In the actual LUTs, the odd LUT indices represent graphics values (least significant bit on).

button while moving the mouse permits drawing of any curve. Moving the cursor to a location and pressing **<ENTER>** will also specify a display value for a given data value. The mouse right button is used to create ramps (lines) of data/display values. First, the cursor must be located at one endpoint of the ramp, then, the right button is depressed and held while the cursor is moved to the other endpoint, and finally, the button is released. Alternatively, from the keyboard, entering **S** (for stretch) will establish one endpoint, then, the cursor can be moved to the other endpoint (with either the mouse or keyboard), and finally, entering **S** again will complete the ramp. When a ramp is initiated, the upper right display shows a message "RAMP ON." The free mode allows the cursor to be moved to any position; whereas in follow mode, right and left cursor movement will cause the cursor to follow the current curve of data/display values. In follow mode, up and down movement at the keyboard allows the cursor to be positioned above or below the curve and pressing **<ENTER>** updates the data/display value to correspond to the cursor position. The mouse left button can be used to draw any curve in follow mode, just as in the free mode.

Typically, data values 128 to 255 are all set to a single display color or gray level; however, special effects can be obtained by mapping different display values to these data values-for instance, displaying graphic overlay in continuous shades of red (without any green or blue) so that basic image features are still distinct when graphic overlay is on.

External LUT files can also be edited using a text editor (see Appendix D for LUT File Format).

<CTRL-R> RESET SYSTEM

Purpose

Reset all or part of the system.

Mode of Operation

A reset menu appears. The following commands are available from the reset menu:

- **1** resets the digitizer registers only.
- **2** resets all LUTs only.
- **3** redisplays the main menu.
- **4** resets the system and clears all images. When the system is reset, the digitizer registers, LUTs, and global variables are all reset.

<ALT-U> UTILITIES MODE

Purpose

Provide utilities to clear all data and graphics overlay for an image, clear only graphics overlays, copy between image buffers, and map LUTs to the data values.

Mode of Operation

- **1** clears the data and graphics values to zero for the active image buffer.
- **2** clears only the graphics plane for the active image buffer,
- **3** initiates copying one image buffer to another. After pressing **3**, the user will be prompted for the source and destination image buffers (only available for digitizer/display adapters with multiple image buffers).
- **4** maps the LUT values to the data values. After pressing **4**, the user is prompted as to whether to map the contents of the red (choice R), green (choice G,) or blue (choice B) LUT to data.

<ALT-U> UTILITIES MODE (continued)

1	**CONTINUOUS LUT**	**<Alt-1>**	**CONTINUOUS LUT POSITIVE/NEGATIVE**
2	**THRESHOLD LUT**	**<Alt-2>**	**THRESHOLD LUT POSITIVE/NEGATIVE**
3	**SLICE LUT**	**<Alt-3>**	**SLICE LUT POSITIVE/NEGATIVE**
4	**PSEUDOCOLOR LUT**	**<Alt-4>**	**PSEUDOCOLOR LUT ONE/TWO**

Purpose

Specifies which LUT is in use. Four sets of LUTs are available – continuous tone, threshold, slice, and pseudocolor.

Mode of Operation

Any lookup table can be set from within any mode.

1 turns on continuous tone display (a linear ramp).

2 turns on threshold display (threshold LUT). While the threshold LUT is in use, + and - respectively increase or decrease the threshold, while showing the current threshold on the screen in real-time. This interactive threshold specification is used to define regions for area measurements in the histogram mode.

3 turns on slice display (slice LUT). Pixels within the slice range are highlighted in color on the image (see Configuration File Format section to change highlight color). The low and high threshold values that define the slice range are shown on the bottom line of the control screen. Pressing + and - respectively increases and decreases the thresholds that define the range of the slice. Pressing **<Spacebar>** toggles between whether + or - changes only the low threshold (asterisk * by the low threshold display), changes only the high threshold (asterisk * by the high threshold display), or changes both thresholds (asterisks ***** by both threshold displays). The **<Tab>** key sets the low threshold to 64 and the high threshold to 127. A **<Shift-Tab>** key combination sets the low threshold to 0 and the high threshold to 63. The low and high thresholds are used to define objects or regions of interest for the flood, edge, and histogram modes.

4 turns on pseudocolor display (pseudocolor LUT). Pseudocolor display allows substitution of color for intensity values. Thus individual intensity values or ranges of intensity values can be assigned display colors. The pseudocolor LUTs are loaded from external files. External LUT files can be edited using the LUT editor (see <**Alt-L**>, LUT UTILITIES) or a text editor (see Appendix D for LUT File Format).

<ALT-1> toggles between negative and positive continuous tone display. NEG (negative) or POS (positive) is displayed in the menu to indicate the active continuous LUT.

<ALT-2> toggles between negative and positive threshold display. NEG (negative) or POS (positive) is displayed in the menu to indicate the active threshold LUT.

<ALT-3> toggles between negative and positive slice display. NEG (negative) or POS (positive) is displayed in the menu to indicate the active slice LUT.

<ALT-4> toggles between pseudocolor display lookup tables ONE and TWO. ONE or TWO is displayed in the menu to indicate the active pseudocolor LUT.

ACKNOWLEDGMENTS

We would like to thank Fairley J. Barnes, Kenneth V. Bostick, Thomas E. Hakonson, Patrica L. Kennedy, and Paul A. Johnson for their considerable administrative and scientific assistance. We thank Harold A. Mooney for support and collaboration at Stanford University, where we are using *IMAGE* for analysis of close-range remote sensing data to study local ecological disturbance patterns. Laurel O. Sillerud and Robert L. Kranz provided additional collaboration and support to develop image analysis capabilities to study nuclear magnetic resonance and mineralogy, respectively. Liane S. Guild thoroughly tested *IMAGE* and suggested various improvements. We also thank Karyn Ames for editing the *IMAGE* manual, Ruth Bigio for graphics support, and Wendy Burditt for composition and layout of the manual. This work was supported, in part, by a Los Alamos Director Funded Postdoctoral Fellowship, the Los Alamos National Environmental Research Park, the United States Department of Energy contract W-7405-ENG-36 to Los Alamos National Laboratory, and National Science Foundation Grant BSR86-14769.

BIBLIOGRAPHY

Andrews, H. 1970. Computer techniques in image processing. Academic Press. New York.

Baxes, G. A. 1984. Digital image processing: a practical primer. Prentice-Hall. Englewood Cliffs, NewJersey.

Cannon, T. M. and B. R. Hunt. 1981. Image processing by computer. Scientific American. October.

Cannon, T. M. 1987. Digital image processing: a short course. Instructor's notes. Los Alamos National Laboratory.

Castleman, K. R. 1979. Digital image processing. Prentice-Hall. Englewood Cliffs, New Jersey.

Ekstrom M. P. (ed) 1984. Digital image processing techniques. Academic Press. New York

Gonzalez, R. and P. Wintz. 1977. Digital Image Processing. Addison-Wesley. Reading, Massachusetts.

Green, W. B. 1983. Digital image processing: a systems approach. Van Nostrand Reinhold. New York.

Imaging Technology. 1985. *PC Vision* frame grabber manual. Imaging Technology Incorporated. Woburn, Massachusetts.

Imaging Technology. 1986. *FG-100-AT* user's manual. Imaging Technology Incorporated. Woburn, Massachusetts.

Imaging Technology. 1987. *PCVISIONplus* frame grabber user's manual. Imaging Technology Incorporated. Woburn, Massachusetts..

Newman, W. M. and R. F. Sproull. 1979. Principles of interactive computer graphics. McGraw-Hill. New York.

Pavladis, T. 1982. Algorithms for graphics and image processing. Computer Science Press, Rockville, Maryland.

Pratt, W. K. 1978. Digital image processing. John Wiley and Sons. New York.

Rich, P.M. 1989. A manual for analysis of hemispherical canopy photography. Los Alamos National Laboratory report LA-11733-M. Los Alamos, New Mexico.

Rosenfeld, A. 1969. Picture processing by computer. Academic Press. New York.

Rosenfeld, A. 1976. Digital picture processing. Academic Press. New York.

APPENDIX A. GLOSSARY

A/D conversion: transformation of information from an analog to a digital form. (see *digitization, D/A conversion*)

analog: information stored in a continuous form, for instance a video image stored as a voltage wave form. (see *digital*)

analog image: an image stored as a continuous (analog) signal, for instance a video signal. (see *digital image*)

AND: a bit level logical operation, whereby if both of two bits have the value 1, the resultant value of the operation between the two bits is 1; otherwise the resultant value is 0. (see *OR, XOR*)

arithmetic operation: performance of addition, subtraction, multiplication, division on data values in a digital image or series of images. (see *scalar operation*, *whole image operation*)

aspect ratio: the ratio of vertical and horizontal dimensions. A standard (RS-170) video image aspect ratio is 3:4. Individual pixels of a 512 (horizontal) x 480 (vertical) pixel video image have an aspect ratio of 5:6, such that 5 pixels distance in the horizontal dimension is equivalent to 6 pixels distance in the vertical dimension.

bit: the smallest digital unit, which can only have the value 0 or 1 (on or off). Eight bits make a byte. (see *byte*)

bit-shift operation: a process whereby all bits of data values in a digital image are shifted to the left or right. A single bit-shift operation to the left is equivalent to multiplying by two; a single bit-shift operation to the right is equivalent to dividing by two.

blanking interval: a period of zero voltage in a video signal, corresponding to shutdown of electron output while the electron beam is shifted across the display.

brightness: as used here, the subjective perceived magnitude of light at given position in an image, as contrasted with intensity which is an absolute magnitude. (see *intensity*)

buffer: computer memory that is allocated for temporary storage of a particular type of information. (see *image buffer*)

bus: a series of lines, or wires, in a computer that connects to *CPU* to other parts of a computer system. A variety of signals can pass in these lines, including power and data. (see *VME*, *Q-BUS*, *NUBUS*, and *microchannel architecture*) *byte*: a digital unit consisting of 8 bits (see *bit*). Because each of the 8 bits can have either of two values (0 or 1), a byte can represent 256 (28) levels of information. Thus a single byte can represent values ranging from 0 to 255. In the case of *IMAGE*, each pixel is assigned one byte, and 7 of the 8 bits in the byte are used for intensity values, with 128 intensity values possible (27), and with values ranging from 0 to 127. The eighth bit is used for graphics overlay (two overlay values--graphics on or off).

CCIR: the black and white standard for video in Europe, consisting of a 1/100-second frame time interlaced signal. The faster scan rate, to prevent interference with the 50-Hz ac current, allows a higher vertical spatial resolution at the expense of more flicker as compared with RS-170. (see *RS-170*)

CGA: color graphics adapter, an IBM graphics display standard that has various graphics modes, including resolutions of 640 x 200 pixels with 2 colors and 320 x 200 pixels with 4 colors. (see *EGA*, *VGA*)

channel: a video signal path in or out of a video system. RGB systems have three channels, with separate channels for red, green, and blue. (see *RGB*)

CCD: charge-coupled device, a semiconductor array often used as a solid-state video sensor. Readout is accomplished by digital shift register techniques, in which accumulated charge through a series of "wells" or sites where charge is temporarily stored.

CID: charge-injection device, often used as a solid-state video sensor. Readout is accomplished by measuring charge on the pair of leads converging on a photosensitive site.

chrominance: as used here, color information encoded in a video signal. (see *luminance*)

color encoding: representing color information by some combination of parameter values, in particular values for red, green, and blue intensity or values for intensity (*luminance*) and hue (*chrominance*). Alternatively, converting an RGB component video signal into NTSC compatible format.

composite video: video that includes intensity information and synchronization information in a single signal.

contrast: the range of intensity values in an image. High contrast refers to an image with a large range and low contrast refers to an image with a small range.

contrast enhancement: a process by which the range of intensity values in an image is increased.

convolution: a mathematical operation that changes a pixel value on the basis of surrounding pixel values. The new value is calculated by multiplying surrounding pixels by an array of constants, or kernel, summing, and scaling. The value of each pixel in an image is systematically calculated in this way. Convolutions can have the effects such as increasing sharpness, blurring, or smoothing edges in an image.

CPU: central processing unit, the part of a computer that controls the flow and manipulation of all digital information.

D/A conversion: transformation of information from a digital to an analog form. For instance, digital images are converted from analog to digital for the purpose of display. (see *digitization*, *A/D conversion*)

data value: the intensity value for a pixel that is stored in an image buffer. (see *intensity value*, *display value*, *lookup tables*)

digital: information stored as a series of discrete units, such that each unit is represented as a numerical (binary) value. (see *analog*)

digital image: an image stored as a series of discrete (digital) brightness values. The video image is stored as a 512 (horizontal) by 480 (vertical) pixel array. (see *analog image*)

digital number: as used here, equivalent to *intensity value*.

digitization: the process of converting an analog signal into digital (binary) form. For example, in the case of digitization of standard (RS-170) video using *IMAGE*, a single frame is converted from an analog *interlaced signal* to a digital non-interlaced signal, with a 512 (horizontal) x 480 (vertical) pixel spatial resolution.

display value: the intensity value for a pixel that is shown on the image monitor, which may be different than the data value, depending upon the output lookup table in use. For display using *IMAGE*, three display values, one each for the red, green, and blue channels, correspond to a given data value. (see *intensity value*, *display value*, *lookup table*).

dynamic range: the range of intensity values present in original image data. (see *intensity value*, *data value*)

EGA: enhanced graphics adapter, an IBM graphics display standard that has various graphics modes are available, including a resolution of 640 x 350 pixels with 16 colors and digital RGB output. (see *CGA*, *VGA*)

EIA: Electronics Institute of America, an organization that sets electronic standards. EIA composite synchronization consists of a negative synchronization pulse that is compatible with (and may be driven by) an RS-170 composite video signal.

external synchronization: when video timing is determined by a device other than the device using it. For example, some video cameras phase-lock to a synchronization signal provided externally. External synchronization also refers to the case of a video display monitor that derives its synchronization from a separate input, rather than from a composite video signal.

F connector: a standard connector type used for video cables. (see *BNC connector*, *RCA connector*)

FCC: Federal Communications Commission, the United States governmental commission that sets communications standards and regulations.

field: in an interlaced video signal (e.g., for RS-170 signals), the set of odd or even lines into which a frame is divided. For standard (RS-170) video signals, one field is presented each 1/60 second. Two fields, one even and one odd, make up a frame or complete video image. (see *RS-170*, *interlaced signal*) *filter*: a digital or analog operation that changes the intensity or spatial characteristics of an image (see *convolution*).

flicker: a perceived periodic change in an image. Flicker appears on the image displays used by *IMAGE* because display lines are only refreshed every 1/30 of a second, within the frequency range of human perception. Flicker is most apparent for narrow (one-pixel width) horizontal features.

frame: an individual video image. In interlaced signals (e.g., for RS-170) two fields, one even and one odd, make up a frame, and a frame is presented each 1/30 second.

frame buffer: as used here, equivalent to *image buffer*.

gray level: as used here, equivalent to *intensity value*.

histogram: the distribution of intensity values within an image or region of an image, generally represented graphically as the number of pixels for each possible data value.

histogram equalization: the process by which the contrast of an image is increased by reassigning new intensity values in proportion to the frequency of occurrence of intensity values, such that contrast is increased for the most frequent intensity values.

image analysis: as used here, the set of capabilities that allow extraction of meaningful information from digital images. (see *image processing*)

image buffer: computer memory that is used for temporary storage of an image. The image buffers used by *IMAGE* allow simultaneous image display and access by the CPU. One image buffer is available for the *PCVision* and *PCVisionPlus* (12.5-MHz version) digitizer/display adapters, two image buffers are available for the *PCVision Plus* (10-MHz version) and *DT2853* adapters; and four image buffers are available for the *FG-100 1024* adapter.

image processing: as used here, the set of capabilities that allow input, display, manipulation, and output of digital images. (see *image analysis*)

intensity: the magnitude of light at a given location in an image. As used here, equivalent to radiance.

intensity value: as used here, the digital value associated with a pixel, equivalent to gray level. *IMAGE* uses 128 intensity values (7 bits) per pixel, with a range from 0 to 127, with 0 representing the darkest possible intensity value and 127 representing the brightest possible intensity value.

interlaced signal: a type of video in which odd and even fields are alternately displayed to reduce flicker. (see *RS-170*)

jitter: noise resulting from timing irregularities. Jitter can be a significant problem when working with video tape recorders.

logical operation: a process whereby data values in an image or series of images are changed according to a set of fundamental logical rules. (see *scalar operation*, *whole image operation*, *AND*, *OR*, *XOR*) *line*: a series of pixels that correspond to the same vertical position. This series of pixels is displayed in a single sweep of a video monitor electron gun. In an RS-170 signal, a maximum of 485 lines are displayed.

lookup table: (LUT) tables of intensity values that are used to reassign intensity values during digitization (*input LUT*) or display (*output LUT*), essentially discrete functions. For example, a negative digital image can be displayed as a postive image by using an output LUT that reassigns bright pixels as dark pixels and dark pixels as bright pixels. Hardware LUTs allow real-time reassignment of intensity values. For *IMAGE*, hardware LUTs are used to assign each *data value* three *display values*, values for red, green, and blue.

luminance: as used here, the intensity information in a video signal, integrated over all spectral bands.

MCA: microchannel architecture, the 32-bit bus standard for IBM PS/2 computers.

multisync: a display device capable of synchronizing to a range of horizontal and vertical drive frequencies, corresponding to different display resolutions.

NUBUS: The Apple Macintosh II 32-bit computer bus.

NTSC: acronym for National Television System Committee. The color composite video standard used in television in the United States and Japan. NTSC is an RS-170 compatible color encoding method, also referred to as RS-170A, that was adopted in 1953. NTSC encodes color information using a single RS-170 compatible signal, whereas RGB typically encodes color using three RS-170 signals, with separate red, green, and blue channels. Two other color video standards, PAL and SECAM, are in use in other countries. (see *RS-170*, *RGB*)

OR: a bit-level logical operation, whereby if either or both of two bits have the value 1, the resultant value of the operation between the two bits is 1; if both bits are 0, the resultant value is 0. (see *AND*, *XOR*)

PAL: Phase Alternation Line, the color video standard used in Great Britain and western Europe, except France. (see *NTSC* and *SECAM*)

phase-locked: two periodic signals that are synchronized in phase.

pixel: the smallest unit of storage in a digital image, with both spatial and spectral attributes (derived from picture element). A digital image is composed of an array of pixels, and each pixel is assigned a specific location and an intensity value or series of intensity values. In the case of *IMAGE*, each pixel is assigned an intensity data value and is displayed using three LUT values, with separate values for red, green, and blue intensity.

proximity focused: image transfer by direct contact between adjacent coherent fiber optic image guides.

pseudocolor: the substitution of color for intensity levels. By the use of output LUTs, it is possible to assign colors to data values by specifying a corresponding mix of red, green, and blue display values.

Q-BUS: the bus used in Digital Equipment Corporation (DEC) *microVAX* computers.

radiance: as used here, equivalent to *intensity*.

raster data: information stored as a series of intensity values for each pixel in an image. Raster data are in a format that can be displayed directly on an image monitor. For image processing, images are generally stored and manipulated as raster data. (see *vector data*)

RCA connector: a plug-type standard connector type used for video cables. (see *BNC connector*, *F connector*)

RGB: a type of color video in which three separate analog video signals, one for red, one for green, and one for blue, are used to produce a color image. RGB systems maintain more true color information than NTSC systems. *IMAGE* uses video digitizer/displays with RGB output. Each output channel carries an RS-170 signal.

readout noise: electronic noise introduced by circuitry that "reads" the charge accumulated in photosensitive sites of a sensor.

real time: digital or analog processes that appear instantaneous.

resolution: as used here, the number of horizontal and vertical pixels that define an image. *IMAGE* uses images with resolutions, as is determined by the digitizer/display adapter. For example, one common fomat consists of a 512 (horizontal) x 480 (vertical) pixel array. Resolution can also be measured in video lines, defined as the number of pairs of black and white parallel lines that could be counted across the display monitor at the limit of detection by a human observer.

RS-170: the black and white standard video format used in the United States. The standard specifies an interlaced signal, with separate odd and even fields combined to produce a frame each 1/30 second, and with 485 lines of information displayed per frame (242.5 lines per field). The interlacing reduces flicker. The standard specifies 525 scan lines/frame, of which 40 lines are used for retrace by the display electron gun. A 60-Hz frame rate was chosen to avoid interference with 60-Hz ac power.

*RS-170*A: equivalent to *NTSC*.

rubber-band cursor: a cursor is a graphics marker that demarcates a location or region on a computer screen. Rubber-band cursors can be interactively changed to different sizes, shapes, and positions. For instance a rectangular rubber-band cursor is used in many routines in *IMAGE* to demarcate rectangular regions of interest on the image monitor.

scalar operation: an arithmetic or logical operation that combines a single (scalar) value with data values in a digital image to produce new data values. For example, a scalar value can be subtracted from all data values in a digital image to produce a new image. (see *arithmetic operation*, *logical image operation*, *whole image operation*)

SECAM: the color video standard used in France and eastern Europe. (see *NTSC* and *PAL*)

shot noise: noise due to the discrete and stochastic nature of photon arrival.

signal-to-noise ratio: the relative proportion of an analog signal that contains useful information, the signal, as opposed to nonuseful information (*noise*).

slice: as used here, a range of data values displayed as a single display value or color. In cases where an image feature is defined by a range of intensities, a slice allows isolation of the feature. In *IMAGE*, an interactive slice display lookup table routine allows the user to specify the upper and lower limits of the slice range while viewing the range as a color on the screen.

slow-scan video: video signal format which require more than 1/30 second per frame, often used for high-resolution or high-precision output.

super VHS: a similar, but higher-resolution version of the VHS format. (see *VHS*)

synchronization pulse: a voltage pulse that signifies the end of a line or field of the video signal.

threshold: as used here, a intensity level that divides an image into two ranges of intensity levels, above and below the threshold. In *IMAGE*, an interactive threshold display lookup table routine allows the user to specify a threshold while displaying an image that represents one range as black and the other as white.

TIFF: Tagged Image File Format, a digital image format popularized by the Apple Macintosh computer, widely used by desktop publishing applications.

TTL: transistor/transistor logic, a family of digital logic devices characterized by certain timing limits and voltage ranges. A digital pulse is in the range of 3-5 V.

value: as used here, equivalent to *intensity value*.

vector data: information stored as a series of coordinates, which may, for instance, define a series of line segments or polygons. Vector data generally takes up less memory and can be more readily scaled to different sizes than raster data. (see *raster data*)

vertical synchronization pulse: a voltage pulse signifying the end of a video field.

VGA: video graphics array, an IBM graphics display standard that has various graphics modes are available, including a resolution of 640 x 480 pixels with 16 or 256 colors and analog RGB output. (see *CGA*, *EGA*)

VHS: the most popular 1/2" video cassette tape format.

VME: a 32-bit standard bus structure used in some models of SUN Microsystems computers, as well as other commercial devices.

whole image operation: an arithmetic or logical operation that combines all data values for all pixels in a digital image with corresponding data values for another digital image to produce a new image. For example, one digital image can be subtracted from another image on a pixel-by-pixel basis to produce a new image. (see *arithmetic operation*, *logical image operation*, *scalar operation*)

XOR: a bit-level logical operation, *exclusive or*, whereby if two bits have different values, the resultant value of the operation between the two bits is 1, and if they have the same value, the resultant value is 0. XOR of two images on a pixel-by-pixel basis will result in an image that shows differences between images. (see *AND*, *OR*)

zoom: an operation in which an image or portion of an image is increased or decreased in size. Increase in size is often accomplished by replicating pixels, and decrease in size by deleting pixels.

APPENDIX B. MENU SCREENS

```
MAIN MENU                                                   IMAGE version 1.3
                                  COMMANDS

             DIGITIZE        D          OPERATIONS        O
             ARCHIVE         A          LUT UTILITIES     <Alt-L>
             LINE            L          CONFIGURATION     <Alt-C>
             RECTANGLE       R          UTILITIES         <Alt-U>
             CIRCLE          C          RESET SYSTEM      <Ctrl-R>
             TRACE           T
             EDGE            E          CONTINUOUS LUT    1 POS
             FLOOD           F          THRESHOLD LUT     2 POS
             HISTOGRAM       H          SLICE LUT         3 POS
             UNITS           U          PSEUDOCOLOR LUT   4 ONE
             MOSAIC          M          BUFFER            B ONE
             QUIT          <ESC>        ACTIVE VALUE      V 127

                              ENTER CHOICE: _


BUFFER 1: image.img
SLICE LUT                    *LOW: 43  HIGH: 85
```

```
CONFIGURATION MENU                                          IMAGE version 1.3




                      I/O ADDRESS          0    300 hex
                      FRAME ADDRES         1    D0000000
                      PSEUDOCOLOT LUT1     2    p1.lut
                      PSEUDOCOLOT LUT2     3    p2.lut
                      GRAPHICS COLOR       4    r=0 g=0 b=0
                      SLICE COLOR          5    r=180 g=0 b=0
                      TRANSPARENCY         6    0 TO 0
                      MOUSE COMMAND        7    initim
                      ASPECT RATIO         8    0.83333


                              ENTER CHOICE: _
```

```
TRACE MENU                                                  IMAGE version 1.3
                                  COMMANDS

                              RESET COUNTER    0
                              ERASE TRACE      E
                              NEW SEGMENT      N
                              AREA             A
                              SAVE TRACE       S
                              LOAD TRACE       L
                              OUTPUT RESULT    O
                              INTENSITY        I
                              DRAW TRACE    <Ctrl-D>
                              ACTIVE VALUE     V     127


                               OBJECTS:      5
                                LENGTH:    315.5 pix       2.722988.cm
    CURSOR:        X, Y      TOT LENGTH:   1444.2 pix      12.463919 cm
    LOCATION    164, 71            AREA:   4745.0 pix       0.294522 sq cm
    VALUE          61         TOT AREA:  12443.0 pix       0.772337 sq cm


BUFFER 1: image.img
CONTINUOUS LUT                                                  TRACE MODE
```

```
EDGE MENU                                                  IMAGE version 1.3



                                  COMMANDS

                              RESET COUNTER    0
                              ERASE EDGE       E
                              SAVE EDGE        S
                              LOAD EDGE        L
                              OUTPUT RESULT    O
                              DRAW TRACE    <Ctrl-D>
                              ACTIVE VALUE     V     GRA



                               OBJECTS:      4
                                LENGTH:    632.2 pix       4.896832.km
    CURSOR:        X, Y      TOT LENGTH:   1953.1 pix      15.128128 km
    LOCATION    232, 185           AREA:   6731.4 pix       0.336547 sq km
    VALUE          23         TOT AREA:  19347.2 pix       0.967293 sq km
```

```
MOSAIC MENU                                                  IMAGE version 1.3

                             LIST: list1.lst
  COMMANDS                     SOURCE LIST                      TARGET LIST
TRANSPARENCY      T       # B   X1,    Y1   X2,    Y2  # B  X1,    Y1   X2,  Y2
REGION SELECT     R       1 1  159,     0  356,   479  1 2 206,     0  305, 239
EDIT LIST         E       2 1  161,     9  368,   385  2 2 306,     0  455, 179
INPUT LIST        I       3 1  108,   135  420,   388  3 2 306,   120  455, 239
OUTPUT LIST       O       4 1    64   172  460,   290  4 2 256,   240  455, 299
BUFFER SELECT     B       5 1   89,   106  489,   353  5 2 256,   300  455, 419
ARCHIVE           A       6 1  150,   135  354,   376  6 2 256,   360  355, 479
BEGIN MOSAIC  <ENTER>     7 1  203,     9  295,   479  7 2 206,   240  255, 479
QUIT           <ESC>      8 1   96,    71  367,   412  8 2  56,   300  205, 479
                          9 1  117,   115  384,   339  9 2  56,   240  205, 359
ENTER CHOICE: _          10 1  127,   185  398,   295 10 2  56,   180  205, 239
                         11 1  117,    40  397,   376 11 2  56,     0  205, 179
                         12 1  206,   118  301,   347 12 2 156,     0  205, 119

          #  B   X1,  Y1   X2,   Y2
SOURCE:   2  1  161,   9  386,  385          TRANSPARENCY RANGE 0-0
TARGET:   2  2   06,   0  405,  179          X SCALE FACTOR: 0.48077
                                             Y SCALE FACTOR: 0.47745
                                              ASPECT CHANGE: 1.00694
BUFFER 1: IMAGE1.IMG
CONTINUOUS LUT
```

APPENDIX C. HARDWARE SPECIFICATIONS*

CURRENTLY SUPPORTED DIGITIZER/DISPLAY ADAPTERS

Imaging Technology *PCVision*:** this framegrabber has a resolution of 512 (horizontal) x 480 (vertical) with 8 bits of information per pixel, 8-bit digitization of RS-170 video, digitization of a single frame in 1/30 second, pseudocolor RGB display capabilities, 4 hardware LUTs, 4 hardware display LUTs. This adapter can store a single frame and can be accessed by the host CPU while it is displaying images. The bus is *XT* compatible. The *PCVision* will not synchronize properly with a video tape recorder without time base correction. The I/O address should generally be set to hexadecimal FF00 and the image buffer to hexadecimal A0000 or D0000. Three hardware configuration changes are necessary for the *PCVision* adapter to operate correctly with *IMAGE*: 1) remove chip ICA85 (ALF374R) and reinstall it in the upper position of its socket; 2) move jumper J66 to position B; and move jumper J65 to upper position (jumper J65 under piggyback memory module).

Imaging Technology *PCVisionplus*:** this framegrabber has a resolution of 512 (horizontal) x 480 (vertical) with 8 bits of information per pixel and two image buffers (10-MHz version), or optionally a resolution of 640 (horizontal) x 480 (vertical) pixels with 8 bits of information per pixel and only one image buffer (12.5-MHz version). Both versions have 8-bit digitization of RS-170 video, digitization of a single frame in 1/30 second, pseudocolor RGB display capabilities, 8 hardware input LUTs, 8 hardware display LUTs, and hardware zoom (2x) and pan capabilities. The image buffer(s) can be accessed by the host CPU while simultaneously displaying images. The bus is XT or AT compatible. The *PCVisionplus* will generally synchronize properly with a video tape recorder. The I/O address should generally be set to hexadecimal 300 and the image buffer to hexadecimal A0000 or D0000.

Imaging Technology ** *FG-100*: this framegrabber has a resolution of 512 (horizontal) x 480 (vertical) pixels with 12 bits of information per pixel (10-MHz version), or optionally a resolution of 640 (horizontal) x 480 (vertical) pixels with 12 bits of information per pixel 8-bit digitization of RS-170 video, digitization of a single frame in 1/30 second, pseudocolor RGB display capabilities, hardware input and display lookup tables which can be variously partitioned (including into 16 256 byte lookup tables), hardware zoom (2X, 4X, 8X) and pan capabilities, and real-time digital feedback of data values through feedback lookup tables. This adapter can store one frames (1 image buffers) and can be accessed by the host CPU while it is displaying images. The bus is AT compatible. The FG-100 will generally synchronize properly with a video tape recorder. The I/O address should generally be set to hexadecimal 300 and the image buffer to hexadecimal A0000 or D0000.

Imaging Technology *FG-100 1024*:** this framegrabber has a resolution of 512 (horizontal) x 480 (vertical) with 12 bits of information per pixel, and four image buffers (10-Mhz version) or optionally a resolution of 640 (horizontal) x 480 (vertical) pixels with 12 bits of information per pixel and two image buffers (12.5-MHz version). Both versions have 8-bit digitization of RS-170 video, digitization a single frame in 1/30 second, pseudocolor RGB display capabilities, hardware input and display LUTs which can be variously partitioned (for example, into sixteen 256-byte LUTs), hardware zoom (2X, 4X, 8X) and pan capabilities, and real-time digital feedback of data values through feedback LUTs. The image buffers can be accessed by the host CPU while simultaneously displaying images. The bus is AT compatible. The *FG-100-1024* will generally synchronize properly with a video tape recorder. The I/O address should generally be set to hexadecimal 300 and the image buffer to hexadecimal A0000 or D0000.

---

* Reference herein to specific commercial products does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government, or by any agency thereof, or by the authors. Product information is provided for informational purposes only.

** Imaging Technology Incorporated, 600 West Cummings Park, Woburn. MA 01801; telephone: (800)532-3500 or (617) 938-8444.

Data Translation[*] ***DT2853:*** This framegrabber has a resolution of 512 (horizontal) x 480 (vertical) pixels with 8 bits of information per pixel, 8-bit digitization of RS-170 video, square pixels produced by sampling only the center portion of each video line at rate of 12.5 MHz, digitization of a single frame in 1/30 second, pseudocolor RGB display capabilities, eight hardware input LUTs, eight hardware display LUTs, and feedback LUT capabilities. This adapter can store two frames (two image buffers) and can be accessed by the host CPU while simultaneously displaying images. The bus is AT compatible. The DT2853 will generally synchronize properly with a video tape recorder. The I/O address generally should be set to hexadecimal 300 and the image buffer to hexadecimal D00000 (in extended memory).

ANALOG RGB VIDEO MONITOR

Sony[**] ***PVM 1342Q*****:** this high resolution analog RGB video monitor has underscan capability (to show full images) and has external synchronization. The red, green, and blue channels of the digitizer/display adapters should be connected to the respective red, green, and blue RGB input channels of the video monitor, and the synchronization cable (brown) should be connected to the external synchronization input. All connections should be properly terminated with the resistance switch in the on position.

VIDEO CAMERA

Cohu[***] ***4800 Solid-State Camera*****:** this solid-state RS-170 video camera uses a CCD (charged couple device) array with a resolution of 754 (horizontal) x 488 (vertical) pixels, high sensitivity, broad spectral response, and excellent signal-to-noise ratio. We use this camera with a Nikkor 55-mm microlens fitted with a C mount adapter.

[*] Data Translation Incorporated, 100 Locke Drive, Marlboro. MA 01752; telephone: (508)481-3700.
[**] Sony Corporation of America. Headquarters, 1600 Queen Anne Road, Teaneck, NJ 07666; telephone: (201)833-5200.
[***] Cohu Incorporated, 5755 Kearny Villa Road, P. O. Box 85623, San Diego. CA 92123; telephone: (619)277-6700.

APPENDIX D. FILE FORMATS

CONFIGURATION FILE FOMAT

The configuration file *IMAGE.CNF* stores the default startup information. It must be accessible for *IMAGE* to run. The file is an ASCII file with the following format:

digitizer register in/out address (in hexadecimal),
image buffer address (in segment offset hexadecimal),
the continuous LUT (0 for positive, 1 for negative),
the threshold LUT (2 for positive, 3 for negative),
the slice LUT (4 for positive, 5 for negative),
the pseudocolor LUT (6 for ONE, 7 for TWO),
the intensity of red in the graphics/cursors (range 0 to 255),
the intensity of green in the graphics/cursors (range 0 to 255),
the intensity of blue in the graphics/cursors (range 0 to 255),
the intensity of red in the slice highlighting (range 0 to 255),
the intensity of green in the slice highlighting (range 0 to 255),
the intensity of blue in the slice highlighting (range 0 to 255),
the aspect ratio,
the filename for pseudocolor LUT ONE,
the filename for pseudocolor LUT TWO,
the lower transparency threshold (range 0 to 127, or -1 for undefined),
the upper transparency threshold (range 0 to 127, or -1 for undefined),
the mouse command (or any command to be executed before running *IMAGE*),
the default calibration (#units/X-pixel distance; 0 if not defined),
and the default unit label,

with blanks between each of the configuration parameters. For example:

300 D0000000 0 2 4 6 0 0 200 180 0 0 0.833333 P1.LUT P2.LUT 0 0
INITIM 0.001

specifies:
I/O register address to be 300 hex,
image buffer address to be D000:0000 hex,
continuous LUT to be positive,
threshold LUT to be positive,
slice LUT to be positive,
graphics/cursor color to be blue (red 0, green 0, blue 200),
slice highlighting to be medium red (red 180, green 0, blue 0),
0.833333 as the aspect ratio
P1.LUT to be the filename for pseudocolor LUT ONE,
P2.LUT to be the filename for pseudocolor LUT TWO,
a transparency range from 0 to 0,
INITIM as the initialization command,
0.001 units/X-pixel distance as the calibration,
and "m" as the unit label,

EDGE COORDINATE FILE FORMAT

Edge coordinate files consist of ASCII lists of the number of boundaries highlighted, the calibration coefficient, the unit label, and the X,Y coordinates for all pixels along the boundaries highlighted in edge mode. All elements of the list are separated by spaces or carriage returns. The first line is of the form:

#OBJECTS CALIBRATION UNITLABEL

where *#OBJECTS* is the number of boundaries highlighted, *CALIBRATION* is the number of units per pixel, and *UNITLABEL* is the unit label. If device coordinates are used, then *CALIBRATION*=0.0 and *UNITLABEL* can be any string (though it must still be present). The lists of X,Y coordinates are in paired columns, such that columns one and two will have the X,Y coordinates of the first boundary, columns three and four will have the X,Y coordinates of the second boundary, etc., with the number of columns dependent on the number of boundaries highlighted:

$$X_1\ Y_1\ X_2\ Y_2\ X_3\ Y_3\ ....$$

In cases where boundaries are different lengths two double quotation marks ("") are used as place holders so that the edge coordinate files can easily be read into spreadsheets, statistics programs, and graphics programs. The X,Y coordinates can be stored in either device coordinates or calibrated coordinates. Device coordinates have the origin (0,0) in the upper left corner of the image, with X increasing from left-to-right and Y increasing from top-to-bottom. Calibrated coordinates have the origin (0,0) in the lower left corner of the image, with X increasing form left-to-right and Y increasing from bottom-to-top. Edge coordinate files can be saved and loaded from edge mode, and serve as one form for raster to vector and vector to raster conversion. The edge mode will highlight a pixel corresponding to each X,Y coordinate specified in an edge coordinate file, whereas the trace mode will connect coordinates with line segments. Thus the edge mode can be used for pixel-by-pixel graphics overlay on an image.

EDGE RESULT FILE FORMAT

Edge result files are ASCII lists of results calculated in the edge mode. Each line has the format:

$$OBJECT\#\ LENGTH\ AREA\ X_c\ Y_c$$

where *OBJECT#* is the number of the boundary (or object), *LENGTH* is the perimeter, *AREA* is the area bounded by the boundary, and X_c, Y_c are the coordinates of the centroid. The units can be either device coordinates (pixels) or calibrated coordinates, depending upon the choice made when saving the file from edge mode. Device coordinates have the origin (0,0) in the upper left corner of the image, with X increasing from left-to-right and Y increasing from top-to-bottom. Calibrated coordinates have the origin (0,0) in the lower left corner of the image, with X increasing from left-to-right and Y increasing from bottom-to-top.

FLOOD RESULT FILE FORMAT

Flood result files are ASCII lists of results calculated in the flood mode. Each line has the format:

$$OBJECT\#\ X_{seed}\ Y_{seed}\ AREA$$

where *OBJECT#* is the number of the flooded object, X_{seed} and Y_{seed} are the coordinate of the seed point (location where the flood was initiated), and *AREA* is the area flooded. The area units can be in either device coordinates (pixels) or calibrated coordinates, depending upon the choice made when saving the file from flood mode. Device coordinates have the origin (0,0) in the upper left corner of the image, with X increasing from left-to-right and Y increasing from top-to-bottom. Calibrated coordinates have the origin (0,0) in the lower left corner of the image, with X increasing from left-to-right and Y increasing from bottom-to-top.

HISTOGRAM FILE FORMAT

Histogram files are ASCII lists of the number of pixels corresponding to each intensity value. There are 128 lines, with a single long integer on each line. Line 1 corresponds to the number of pixels with intensity value 0, line 2 to the number of pixels with intensity value 1, etc.

IMAGE FILE FORMAT

Images are stored as binary files with one byte of information per pixel, resolution 512 (horizontal) x 480 (vertical) pixels. Some digitizer/display adapters produce images with other resolutions (512 x 512 pixels, 640 x 480 pixels, or 640 x 512 pixels). For Imaging Technology digitizer/display adapters, for each byte of information, the most significant bit contains graphics overlays and the 7 least significant bits contain the intensity value of the pixel (i.e., 128 levels, 0 to 127 possible). For Data Translation digitizer/display adapters, for each byte of information, the least significant bit contains graphics overlays and the 7 most significant bits contain the intensity value of the pixel (also, 128 levels, 0 to 127 possible). For use in other programs, it may be necessary to bit-shift data one bit to the left or to adjust LUTs accordingly.

INTENSITY FILE FORMAT

Intensity files consist of ASCII lists of pixel coordinates and pixel data values for lines or curves specified in the trace mode. Each line has the format

X Y VALUE

where *X,Y* are the pixel coordinates and *VALUE* is the pixel data value. The X,Y coordinates can be in either device coordinates (pixels) or calibrated coordinates, depending upon the choice made when saving the file from trace mode. Device coordinates have the origin (0,0) in the upper left corner of the image, with X increasing from left-to-right and Y increasing from top-to-bottom. Calibrated coordinates have the origin (0,0) in the lower left corner of the image, with X increasing from left-to-right and Y increasing from bottom-to-top. Values range from 0 to 127.

LUT FILE FORMAT

LUT files consist of ASCII lists of the red, green, and blue display values (range 0 to 255) corresponding to each of the data values. There are 256 lines with three integers per line. Each line corresponds to a data value and the three integers correspond to display value intensities of red, green, and blue, respectively. Thus if line 33 is 20 200 50, data value 32 will be displayed as the color with intensity red 20, green 200, and blue 50. Note that line 1 corresponds to data value 0 and line 256 corresponds to data value 255. The LUT values for data values 128 to 255 are graphic values, and should generally be all the same, for instance 127 127 127 for medium gray, which shows up well against bright colors. Any user-specified LUT file can be loaded, whether it genuinely specifies a pseudocolor LUT or some other possible LUT.

MOSAIC LIST FILE FORMAT

Mosaic lists are ASCII files that consist of 12 lines, with each line specifying the image buffer and coordinates for a source and target region of interest. Each line has the format

SOURCEBUFFER SX1 SY1 SX2 SY2 TARGETBUFFER TX1 TY1 TX2 TY2

where *SOURCEBUFFER* is the source buffer (1, 2, 3, or 4, depending on how many buffers are available), *SX1,SY1* are the upper left corner source region X,Y coordinates, *SX2,SY2* are the lower right corner source region X,Y coordinates, *TARGETBUFFER* is the target buffer (also 1, 2, 3, or 4), *TX1,TY1* are the upper left corner target region X,Y coordinates, and *TX2,TY2* are the lower right corner target region X,Y coordinates. The coordinates must be in device coordinates.

TRACE COORDINATE FILE FORMAT

Trace coordinate files consist of an ASCII list of the number of traces, the calibration coefficient, the unit label, and the X,Y coordinates for the endpoints of a series of connected line segments. All elements of the list are separated by spaces or carriage returns. The first line is of the form

#OBJECTS CALIBRATION UNITLABEL

where *#OBJECTS* is the number of traces highlighted, *CALIBRATION* is the number of units per pixel, and *UNITLABEL* is the unit label. If device coordinates are used, then *CALIBRATION*=0.0 and *UNITLABEL* can be any string (though it must still be present). The lists of X,Y coordinates are in paired columns, such that columns one and two will have the X,Y coordinates of the first trace, columns three and four will have the X,Y coordinates of the second boundary, etc., with the number of columns dependent on the number of traces highlighted:

$$X_1\ Y_1\ X_2\ Y_2\ X_3\ Y_3 \ldots.$$

In cases where the number of X,Y coordinates in a trace are different, two double quotation marks ("") are used as place holders so that the edge coordinate files can easily be read into spreadsheets, statistics programs, and graphics programs. The X,Y coordinates can be stored in either device coordinates or calibrated coordinates. Device coordinates have the origin (0,0) in the upper left corner of the image, with X increasing from left-to-right and Y increasing from top-to-bottom. Calibrated coordinates have the origin (0,0) in the lower left corner of the image, with X increasing from left-to-right and Y increasing from bottom-to-top. Trace coordinate files can be saved and loaded from trace mode, and serve as one form for raster to vector and vector to raster conversion. The trace mode will highlight pixels in the image by connecting all X,Y coordinates in each paired column specified in a trace coordinate file, whereas the edge mode will highlight only pixels without connecting between coordinates. Thus the trace mode can be used to draw any curve on an image by specifying inflection points.

TRACE RESULT FILE FORMAT

Trace result files are ASCII lists of results calculated in the trace mode. Each line has the format:

OBJECT# LENGTH AREA

where *OBJECT#* is the number of the trace (or object), *LENGTH* is the length of the trace, *AREA* is the area bounded by the trace, with the endpoints connected. The area will have no meaning if a trace crosses over itself. The units can be either device coordinates (pixels) or calibrated coordinates, depending upon the choice made when saving the file from the edge mode. Device coordinates have the origin (0,0) in the upper left corner of the image, with X increasing from left-to-right and Y increasing from top-to-bottom. Calibrated coordinates have the origin (0,0) in the lower left corner of the image, with X increasing from left-to-right and Y increasing from bottom-to-top.

APPENDIX E. LISTING OF FILES AND UTILITY PROGRAMS

IMAGE FILES

IMAGE.EXE the executable program
IMAGE.CNF the configuration file
P1.LUT an example pseudocolor LUT file
P2.LUT an example pseudocolor LUT file

UTILITY PROGRAMS

Utility programs can be run from the *DOS* command line. Many of the programs have arguments that must be specified on the command line. When specifying arguments, each argument must be separated with a space. For example,

HIST 23 20 103 209

would calculate the histogram for the region of interest defined by the rectangle with the X,Y coordinates (23,20) in the upper left corner and (103,209) in the lower right corner. A hardware specific set of utility programs is available for a given digitizer/display adapter. Depending upon the I/O and image buffer addresses used, the utility programs may require recompiling. The following listing summarizes the utility programs and the command line arguments. Optional arguments are shown in square brackets.

AND: does a logical AND operation with a pixel value. Arguments: **X Y ANDVALUE**, where **X,Y** are the coordinates of the pixel to be complemented (X axis range 0 to 511, Y axis range 0 to 479) and **ANDVALUE** is the value that is ANDed to the pixel value (range 0 to 255). No defaults.

ANDALL: does a logical AND operation with values €or all pixels in an image. Argument: **ANDVALUE**, where **ANDVALUE** is the value that is ANDed to the pixel values (range 0 to 255). No defaults. Only available for the *DT2853* and *Series 100* adapters.

ARC: draws an arc or circle. Arguments: **X Y RADIUS [VALUE START END ASPECT]**, where **X,Y** are the centerpoint coordinates, **RADIUS** is the radius in pixels along the X dimension, **VALUE** is the intensity value to be written (range 0 to 255), **START** and **END** are the starting and ending angles in degrees, and **ASPECT** is the aspect ratio. The defaults are **VALUE**=255, **START**=0, **END**=360, and **ASPECT**=0.83333.

BOX: draws a rectangle. Arguments: **X_1 Y_1 X_2 Y_2 [VALUE]**, where **X_1,Y_1** and **X_2,Y_2** are the coordinates of two diagonal corners that define a rectangle (X axis range 0 to 511, Y axis range 0 to 479) and **VALUE** is the intensity value to be written (range 0 to 255). The default is **VALUE**=255.

BOXCUR: produces a rubber-band rectangular cursor that can be used to draw filled rectangles. The cursor is controlled using either the mouse or keyboard. Mouse movement up/down and right/left moves the cursor accordingly; the right button combined with movement of the mouse stretches the cursor to different shapes; and the left button draws a filled rectangle. On the keyboard, the arrow keys move the cursor one pixel in the corresponding direction; **<Ctrl-Left>** and **<Ctrl-Right>** move the cursor 10 pixels to the left or right, respectively; and **<PgUp>** and **<PgDn>** move the cursor 10 pixels up or down, respectively. Slash (/)toggles between this cursor movement mode and a cursor stretch mode in which the arrow keys, **<Ctrl-Left>**, **<Ctrl-Right>**, **<PgUp>**, and **<PgDn>** allow the cursor to be stretched. Press **<ENTER>** to draw a filled rectangle. Press **<ESC>** to stop the program. Argument: **[VALUE]**, where

VALUE is the intensity value to be written (range 0 to 255). The default is **VALUE**=255.

BUFFER: changes to the specified image buffer. Argument: **BUFFER#**, where **BUFFER#** is the number of the buffer (1, 2, 3, or 4, depending on how many image buffers are available). No defaults. Not available for digitizer/display adapters with only one image buffer.

CIRCLE: draws a circle or ellipse. Arguments: **X Y RADIUS [VALUE ASPECT]**, where **X,Y** are the coordinates of circle centerpoint (X axis range 0 to 511, Y axis range 0 to 479), **RADIUS** is the radius in pixels along the X dimension, **VALUE** is the intensity value to be written (range 0 to 255). The defaults are **VALUE**=255.

CIRCUR: produces a rubber-band circular cursor that can be used to draw filled circles or ellipses. The cursor is controlled using either the mouse or keyboard. Mouse movement up/down and right/left moves the cursor accordingly; the right button combined with movement of the mouse stretches the radius of the cursor; and the left button draws a filled circle or ellipse. On the keyboard, the arrow keys move the cursor one pixel in the corresponding direction; **<Ctrl-Left>** and **<Ctrl-Right>** move the cursor ten pixels to the left or right, respectively; and **<PgUp>** and **<PgDn>** move the cursor ten pixels up or down, respectively. Slash (/) toggles between this cursor movement mode and a cursor stretch mode in which the arrow keys, **<Ctrl-Left>**, **<Ctrl-Right>**, **<PgUp>**, and **<PgDn>** allow the cursor to be stretched. Press **<ENTER>** to draw a filled circle or ellipse. Press **<ESC**> to stop the program. Argument: **[VALUE]**, where **VALUE** is the intensity value to be written (range 0 to 255) and **ASPECT** is the aspect ratio. The defaults is **VALUE**=255 and **ASPECT**=0.83333.

CLDIG: clears the active display buffer. Argument: **[VALUE]**, where **VALUE** is the intensity value to which the buffer will be cleared (range 0 to 255). The default is **VALUE**=0.

CONV: converts images in quadrant format to strip format. This is used to convert images stored with the default *PCVision* hardware setup to the strip format used in *IMAGE*. Arguments: **OLDFILE NEWFILE**, where **OLDFILE** is the name of the file to be converted and **NEWFILE** is the name of the converted file. **OLDFILE** and **NEWFILE** can be the same name, in which case the old file will be written over. Any valid *DOS* paths and filenames can be specified. No defaults.

C_PIX: complements a pixel. The new pixel value becomes 255 minus the current pixel value. Arguments: **X Y**, where **X,Y** are the coordinates of the pixel to be complemented (X axis range 0 to 511, Y axis range 0 to 479). No defaults.

FEEDBAK: does a feedback operation using a specified input LUT, whereby data values are "piped" through an input LUT to produce new data values. Argument: **LUT#**, where **LUT#** is the number of the input LUT to use for the feedback operation. The input LUT can be loaded using LOADIL. No defaults. Only available for the *DT2853* and *Series 100* adapters.

FLOOD: floods a region to a given value. The region includes all adjacent pixels within a given intensity value range Arguments: **X Y LOWER UPPER [VALUE]**, where **X,Y** are the coordinates of a seed point from which flooding begins (X axis range 0 to 511, Y axis range 0 to 479), **LOWER** and **UPPER** define the lower and upper bounds (respectively) of the intensity value range to be flooded (range 0 to 255), and **VALUE** is the value to be written to the region (range 0 to 255). The default is **VALUE**=255.

GAIN: sets the gain. Argument: **GAINVALUE**, where **GAINVALUE** is the new gain value (range 0 to 100, factory default 49 to 51; *PCVisionplus* only). The default is **GAINVALUE**=50.

GETBUF: determines which image buffer is currently being displayed. No arguments or defaults. Not available for digitizer/display adapters with only one image buffer.

GRAB: turns on continuous digitization. No arguments.

HIST: calculates a histogram for a specified rectangular region of interest. A list is displayed of the number of pixels with each data value. Arguments: **X1 Y1 X2 Y2,** where **X1,Y1** and **X2,Y2** are the coordinates of two diagonal corners that define a rectangular region of interest (X axis range 0 to 511, Y axis range 0 to 479). No defaults.

INITA: (initialize all) initializes the digitizer registers, initializes the LUTs, and clears the data buffers to intensity value 0. LUT 0 is initialized to a positive ramp, LUT 1 is initialized to a negative ramp, LUT 2 is initialized to a positive threshold, and LUT 3 is initialized to a negative threshold. Note this utility does not set up a graphic bitplane, but rather uses all 8 bits. Arguments: **LUT#**, where **LUT#** is the number of the LUT to be made active (range 0 to 3 for PCVISION, 0 to 7 for other digitizer/display adapters). The default is **LUT#**=0.

INITDIG: initializes the digitizer registers. No arguments or defaults.

INITLUT: initializes the LUTs. LUT 0 is initialized to a positive ramp, LUT 1 is initialized to a negative ramp, LUT 2 is initialized to a positive threshold, and LUT 3 is initialized to a negative threshold. Note this utility does not set up a graphic bitplane, but rather uses all 8 bits. Argument: **LUT#**, where **LUT#** is the number of the LUT to be made active (range 0 to 3 for PCVISION, 0 to 7 for other digitizer/display adapters). The default is **LUT#**=0.

I2TIFF: converts an image file to a graphic output TIFF file. This is used to convert images to the standard TIFF file format for printing of image hardcopies on graphics printers. Arguments: **OLDFILE TIFFILE**, where **NEWFILE** is the name of the file to be converted and **TIFFILE** is the name of the TIFF file. Any valid *DOS* paths and filenames can be specified. No defaults.

LINE: draws a line. Arguments: **X1 Y1 X2 Y2 [VALUE]**, where **X1,Y1** and **X2,Y2** are the coordinates of two endpoints (X axis range 0 to 511, Y axis range 0 to 479) and VALUE is the intensity value to be written (range 0 to 255). The default is **VALUE**=255.

LINECUR: produces a rubber-band line cursor that can be used to draw lines. The cursor is controlled using either the mouse or keyboard. Mouse movement up/down and right/left moves the cursor accordingly; the right button combined with movement of the mouse stretches the cursor to different shapes; and the left button draws a line. On the keyboard, the arrow keys move the cursor one pixel in the corresponding direction; **<Ctrl-Left>** and **<Ctrl-Right>** move the cursor ten pixels to the left or right, respectively; and **<PgUp>** and **<PgDn>** move the cursor ten pixels up or down, respectively. Slash, /, toggles between this cursor movement mode and a cursor stretch mode in which the arrow keys, **<Ctrl-Left>**, **<Ctrl-Right>**, **<PgUp>**, and **<PgDn>** allow the cursor to be stretched. Press **<ENTER>**to draw a line. Press **<ESC>** to stop the program. Argument: **[VALUE]**, where **VALUE** is the intensity value to be written (range 0 to 255). The default is **VALUE**=255.

LOAD: loads an image from disk. Argument: **IMAGEFILE**, where **IMAGEFILE** is the name of an image file to be loaded. Any valid *DOS* path and filename can be specified. No defaults.

LOADLUT: loads a LUT file into the currently active LUT. Arguments: **LUT# LUTFILE**, where **LUT#** is the number of the display LUT to be used and **LUTFILE** is the name of a LUT file to be loaded. Any valid *DOS* path and filename can be specified. No arguments or defaults.

LOADIL: loads a LUT file into a specified input LUT. Arguments**: ILUT# ILUTFILE**, where **ILUT#** is the number of the input LUT to be used and **ILUTFILE** is the name of a LUT file to be loaded. Any valid *DOS* path and filename can be specified. No defaults.

LTALL: does a linear transformation of values for all pixels in an image. Arguments: **SLOPE INTERCEPT**, where **SLOPE** is a floating point value that all data values are multiplied by and **INTERCEPT** is a floating point value that is then added to all data values. No defaults. Only available for the *DT2853* and *Series 100* adapters

MOSAIC: copies a rectangular region of interest to another rectangular region of interest, rescaling as necessary. Arguments**: SOURCE SX1 SY1 SX2 SY2 TARGET TX1 TY1 TX2 TY2 [LOWT HIGHT]**, where **SOURCE** is the source buffer number (1, 2, 3, or 4, depending on how many buffers are available), **SX_1,SY_1** and **SX_2,SY_2** are the coordinates of two diagonal corners that define a rectangular source region (X axis range 0 to 511, Y axis range 0 to 479), **TARGET** is the target buffer number (1, 2, 3, or 4, depending on how many buffers are available), and **TX_1,TY_1** and **TX_2,TY_2** are the coordinates of two diagonal corners that define a rectangular target region (X axis range 0 to 511, Y axis range 0 to 479). **LOWT HIGHT** is the limit of transparency range. **LOWT** must be less than or equal to **HIGHT** and be in the range -1 to 127. A value of -1 means that no transparency range is defined. Defaults **LOWT**=-1 and **HIGHT**=-1. Not available for PC Vision or FG-100.

OFFSET: set the offset. Argument: **OFFSETVALUE**, where **OFFSETVALUE** is the new gain value (range 0 to 100, factory default 74 to 76; *PCVISIONplus* only). The default is **OFFSETVALUE**=75.

OR: does a logical OR operation with a pixel value. Arguments: **X Y ORVALUE**, where **X,Y** are the coordinates of the pixel to be complemented (X axis range 0 to 511, Y axis range 0 to 479) and **ORVALUE** is the value that is ORed to the pixel value (range 0 to 255). No defaults.

ORALL: does a logical OR operation with values for all pixels in an image. Argument: **ORVALUE**, where **ORVALUE** is the value that is ORed to the pixel values (range 0 to 255). No defaults. Only available for the *DT2853* and *Series 100* adapters.

PAN: specifies the position of the hardware pan. Arguments: **X Y**, where **X,Y** are the coordinates of the upper left corner of the active buffer display (X axis range 0 to 1023, Y axis range 0 to 511 for *PCVISIONplus*; X axis range 0 to 1023, Y axis range 0 to 1023 for *FG-100-1024*). For the *PCVISIONplus*, PAN only works for 8 pixel boundaries in the X dimension and 2 pixel boundaries in the Y dimension. The default values are X=0 and Y=0. Not available for the *PCVISION* or *DT2853* adapters.

PROTECT: protects a bit plane or set of bit planes. Argument: **BITNUMBER**, where **BITNUMBER** specifies the bitplanes to be protected. For the Imaging Technology digitizer display adapters, any combination of the 8 bit planes can be specified with a 1-byte integer value (range 0 to 256).

For the *DT2853*, bit 0 of **BITNUMBER** specifies protection bit plane 0, bit 1 of **BITNUMBER** specifies protection bit plane 1, bit 2 of **BITNUMBER** specifies protection bit planes 2 and 3, and bit 3 of **BITNUMBER** specifies protection bit planes 4,5,6, and 7. No defaults.

REG: displays the current contents of the digitizer registers. No arguments or defaults.

R_PIX: (read pixel) displays the value of a pixel. Arguments: **X Y**, where **X,Y** are the coordinates of the pixel to be read (X axis range 0 to 511, Y axis range 0 to 479). No defaults.

SAVE: saves an image to disk. Argument: **IMAGEFILE**, where **IMAGEFILE** is the name of an image file to be loaded. Any valid *DOS* path and filename can be specified. No defaults.

SAVEIL: saves the contents of a specified input LUT to a LUT file. Arguments: **ILUT# ILUTFILE**, where **LUT#** is the number of the input LUT to be used and **ILUTFILE** is the name of a LUT file to be saved. Any valid *DOS* path and filename can be specified. No defaults.

SAVELUT: save the contents of the currently active LUT to a LUT file. Argument: **LUT#LUTFILE**, where **LUT#** is the number of the display LUT to be used and **LUTFILE** is the name of a LUT file to be saved. Any valid *DOS* path and filename can be specified. No defaults.

SELIL: selects an active input LUT. Argument: **ILUT#**, where **ILUT#** is the number of the LUT to be made active (range 0 to 3 for *PCVISION*, 0 to 7 for *PCVISIONplus* and *DT2853*, and 0 to 16 for *FG-100* and *FG-100-1024*). The default is **LUT#**=0.

SELLUT: selects a LUT for display. Argument: **LUT#**, where **LUT#** is the number of the LUT to be made active (range 0 to 3 for *PCVISION*, 0 to 7 for *PCVISIONplus* and *FG-100*). The default is **LUT#**=0.

SHIFTL: shifts the data values of the image buffer one bit to the left. This is useful for converting 7-bit images used by *IMAGE* into an 8-bit format. No arguments or defaults.

SHIFTR: shifts the data values of the image buffer one bit to the right. This is useful for converting 8-bit images into the 7-bit format used by *IMAGE*. No arguments or defaults.

SNAP: captures a single image. If the digitizer is in continuous image acquisition mode (see GRAB, earlier in this appendix), then digitization is stopped. No arguments or defaults.

W_PIX: (write pixel) writes a pixel value to data. Arguments: **X Y VALUE**, where **X,Y** are the coordinates of the pixel to be read (X axis range 0 to 511, Y axis range 0 to 479) and **VALUE** is the intensity value to be written (range 0 to 255). No defaults.

W_PIX8: writes 8 adjacent pixel values. Arguments: **X Y VALUE**, where **X,Y** are the coordinates of the pixel to be read (X axis range 0 to 511, Y axis range 0 to 479) and **VALUE** is the intensity value to be written (range 0 to 255). No defaults. *PCVISIONplus* only.

XCUR: produces a cross-hair cursor that can be used to examine individual pixel coordinates and data values. The cursor is controlled using either the mouse or keyboard. As the cursor is moved to a given pixel, the pixel **X,Y** coordinates and intensity value are displayed. Mouse movement up/down and right/left moves the cursor accordingly. On the keyboard, the arrow keys move the cursor one pixel in the corresponding direction; **<Ctrl-Left>** and **<Ctrl-Right>** move the cursor 10 pixels to the left or right, respectively; and **<PgUp>** and **<PgDn>** move the cursor ten pixels up or down, respectively. Press **<ESC>** to stop the program. No defaults or arguments.

XOR: does a logical XOR (exclusive OR) operation with a pixel value. Arguments: **X Y XORVALUE**, where **X,Y** are the coordinates of the pixel to be complemented (X axis range 0 to 511, Y axis range 0 to 479) and **XORVALUE** is the value that is XORed to the pixel value (range 0 to 255). No defaults.

XORALL: does a logical XOR (exclusive OR) operation with values for all pixels in an image. Argument: **XORVALUE**, where **XORVALUE** is the value that is XORed to the pixel values (range 0 to 255). No defaults. Only available for the *DT2853* and *Series 100* adapters.

ZOOM: controls the hardware zoom. Argument: **ZOOMVALUE**, where **ZOOMVALUE** is the number of times that the image is to be zoomed (1 or 2 for the *PCVISIONplus*; 1, 2, 4, or 8 for the *FG-100*). The default is **ZOOMVALUE**=1. Not available for the *PCVISION* or *DT2853*.

APPENDIX F. COMMAND SUMMARY

ARCHIVE (A)

L	LOAD IMAGE	**S**	SAVE IMAGE
D	DOS SHELL	**<ESC>**	EXIT/ABORT

ARITHMETIC OPERATIONS (see OPERATIONS)

BUFFER CONTROL

B BUFFER (ONE/TWO) TOGGLE

CONFIGURATION (<ALT-C>)

0	DIGITIZER I/O ADDRESS	**5**	SLICE HIGHLIGHT COLOR
1	BUFFER ADDRESS	**6**	TRANSPARENCY RANGE
2	PSEUDOCOLOR ONE LUT FILE	**7**	INITILIZATION COMMAND
3	PSEUDOCOLOR TWO LUT FILE	**8**	ASPECT RATIO
4	CURSOR/GRAPHICS-OVERLAY COLOR	**<ESC>**	UPDATE AND EXIT

CURSOR CONTROL

KEYBOARD		**MOUSE**
<LEFT>	LEFT ONE	MOVEMENT LEFT
<RIGHT>	RIGHT ONE	MOVEMENT RIGHT
<UP>	UP ONE	MOVEMENT UP
<DOWN>	DOWN ONE	MOVEMENT DOWN
/STRETCH/MOVEMENT)	TOGGLE	<RIGHT-BUTTON> STRETCH
<ENTER>	COMPLETE ACTION	<LEFT-BUTTON> COMPLETE ACTION

DIGITIZATION

D DIGITIZE

DISPLAY LUT (see *LUT DISPLAY CONTROL, LUT UTILITIES, UTILITIES*: MAP LUT TO DATA)

EDGE DETECTION (see *MEASURE*: EDGE; *LUT DISPLAY CONTROL*: THRESHOLD, SLICE)

EDITING

C	CIRCLE	**R**	RECTANGLE
	1 FILLED		1 FILLED
	2 UNFILLED		2 UNFILLED
L	LINE (see MEASUREMENT)	**T**	TRACE (see MEASUREMENT)
E	EDGE (see MEASUREMENT)	**F**	FLOOD (see MEASUREMENT)
V	ACTIVE VALUE		(enter values 0-127 or G for graphics plane)

ENHANCEMENT (see *LUT DISPLAY CONTROL, LUT UTILITIES, UTILITIES:* **MAP LUT TO DATA**)

HISTOGRAM (H)

Y	RESCALE Y AXIS	**A**	AUTOSCALE
I	INTEGRATE	**<ESC>**	EXIT/ABORT

INPUT OF IMAGES (see *ARCHIVE*)

INTENSITY DATA MEASUREMENT (see *MEASUREMENT:* **TRACE**)

LOGICAL OPERATIONS (see *OPERATIONS*)

LUT DISPLAY CONTROL

1 CONTINUOUS TONE DISPLAY <ALT-1> NEGATIVE/POSITIVE TOGGLE

2 THRESHOLD DISPLAY <ALT-1> NEGATIVE/POSITIVE TOGGLE
+ INCREASE THRESHOLD
- DECREASE THRESHOLD

3 SLICE DISPLAY <ALT-1> NEGATIVE/POSITIVE TOGGLE
+ INCREASE THRESHOLD
- DECREASE THRESHOLD
<SPACEBAR> (UPPER/LOWER/UPPER-AND-LOWER THRESHOLD) TOGGLE
<TAB> RANGE 64-127
<LEFT-TAB> 0-63

4 PSEUDOCOLOR DISPLAY <ALT-1> LUT (ONE/TWO) TOGGLE

LUT UTILITIES <ALT-1>

L	LOAD LUT FILE	**S**	SAVE LUT FILE
E	ENTER LUT EDITOR	**<ESC>**	EXIT

R	RED CHANNEL
G	GREEN CHANNEL
B	BLUE CHANNEL
A	ALL CHANNELS
S	<RIGHT-MOUSE-BUTTON> STRETCH ENDPOINT
F	FREE FOLLOW TOGGLE
<ENTER>	/<LEFT-MOUSE-BUTTON> CHANGE VALUE
<ALT-G>	EDIT GRAPHICS VALUES
<ALT-4>	PSEUDOCOLOR DISPALY (ONE/TWO) TOGGLE
<ESC>	EXIT

MEASUREMENT

- **U** UNIT CALIBRATION
- **L** LINE
- **E** EDGE
  - **O** RESET LENGTH COUNTER
  - **E**/ <RIGHT-MOUSE-BUTTON> ERASE BOUNDARY TRACES
  - **S** SAVE EDGE COORDINATE FILE
  - **L** LOAD EDGE COORDINATE FILE
  - **O** OUTPUT RESULTS
  - **A** AUTOMATICALLY SCAN
  - **<CTRL-D>** DRAW BOUNDARY TRACE
  - **<ESC>** EXIT
- **T** TRACE
  - **0** RESET LENGTH COUNTER
  - **E** ERASE TRACES
  - **N** START NEW SEGMENT (keyboard)
  - **S** SAVE TRACE COORDINATE FILE
  - **L** LOAD TRACE COORDINATE FILE
  - **O** OUTPUT RESULTS
  - **A** AREA MEASUREMENT
  - **I** OUTPUT INTENSITY FILE
  - **<CTRL-D>** DRAW BOUNDARY TRACE
  - **<ESC>** EXIT
- **F** FLOOD
  - **0** RESET COUNTER
  - **E** ERASE TRACES
  - **O** OUTPUT RESULTS
  - **<ESC>** EXIT

OPERATIONS (O)

- **1** SINGLE IMAGE
  - **1** LINEAR TRANSFORM
  - **2** BIT-SHIFT
- **2** MULTIPLE IMAGE
  - **1** SUBTRACT
  - **2** ADD
  - **3** DIVIDE
  - **4** MULTIPLY
- **3** MULTIPLE IMAGE
  - **1** AMD
  - **2** OR
  - **3** XOR

OUTPUT OF IMAGES (see *ARCHIVE*)

RASTER TO VECTOR CONVERSION (see *MEASUREMENT*: EDGE, TRACE)

RECTIFICATION (see *RESCALING AND REGION MOVEMENT*)

RESCALING AND REGION MOVEMENT (M MOSAIC)

I	INPUT LIST FILE	**O**	OUTPUT LIST FILE
R	REGIONS OF INTEREST	**<ENTER>**	INITIATE MOSAIC OPERATION
E	EDIT LIST	**<ESC>**	EXIT/ABORT
T	TRANSPARENCY (0-127 or enter U for undefined)		

RESET SYSTEM (<CTRL-R>)

1	RESET REGISTERS	**3**	REDISPLAY MENUS
2	RESET LUTS	**4**	RESET SYSTEM

RESTARTING PROGRAM WITHOUT CLEARING IMAGE BUFFER

IMAGE - r

SCALAR OPERATIONS (see *OPERATIONS*)

VECTOR TO RASTER CONVERSION (see *MEASURE*: EDGE, TRACE)

UTILITIES (<ALT-U>)

1	CLEAR ALL	**3**	COPY IMAGE BUFFER
2	CLEAR GRAPHICS	**4**	MAP LUT TO DATA

WHOLE IMAGE OPERATIONS (see *OPERATIONS*)

---

* U.S. GOVERNMENT PRINTING OFFICE: 1990 – 773-205/20014

This report has been reproduced directly from
the best available copy.

Available to DOE and DOE contractors from
the Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37832
Prices available from
(615) 576-8401, FTS 626-8401

Available to the public from
The National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Rd.
Springfield, VA 22161

Microfiche A01

Page Range	NTIS Price Code	Page Range	NTIS Price Code	Page Range	NTIS Price Code	Page Range	NTIS Price Code
001-025	A02	151-175	A08	301-325	A14	451-475	A20
026-050	A03	176-200	A09	326-350	A15	476-500	A21
051-075	A04	201-225	A10	351-375	A16	501-525	A22
076-100	A05	226-250	A11	376-400	A17	526-550	A23
101-125	A06	251-275	A12	401-425	A18	551-575	A24
126-150	A07	276-300	A13	426-450	A19	576-600	A25
						601-up*	A99

*Contact NTIS for a price quote

Los Alamos

Los Alamos National Laboratory
Los Alamos,New Mexico 87545