PROC10 -- AN IMAGE PROCESSING SYSTEM FOR THE PDP10: OPERATION AND DESCRIPTION

NCI/IP Technical Report #8

December 16, 1976

Peter Lemkin, Bruce Shapiro,
Richard Gordon, Lewis Lipkin

National Cancer Program

U.S. Department of Health, Education, and Welfare / National Institutes of Health / National Cancer Institute

NCI/IP-76/06

PROC10 -- AN IMAGE PROCESSING SYSTEM FOR THE PDP10: OPERATION AND DESCRIPTION

NCI/IP Technical Report #8

December 16, 1976

Peter Lemkin, Bruce Shapiro,
Richard Gordon, Lewis Lipkin

Image Processing
Division of Cancer Biology and Diagnosis
National Cancer Institute
National Institutes of Health
Bethesda, Maryland 20014

"We here highly resolve . . ."

NCI/IP-76/06

PROC10

AN IMAGE PROCESSING SYSTEM FOR THE PDP10:

OPERATION AND DESCRIPTION

NCI/IP Technical Report #8

P. Lemkin, B. Shapiro, R. Gordon, L. Lipkin

Image Processing Unit
Division of Cancer Biology and Diagnosis
National Cancer Institute
National Institutes of Health
Bethesda, Md. 20014

December 16, 1976

ABSTRACT

PROC10, an interactive image processing system, runs on a PDP10 computer. It can manipulate picture, mask, boundary, boundary transform and computing window data structures. PROC10 provides many operations on and between these data structures. Images and boundaries may be displayed on several different types of terminals including the DEC GT40, Tektronix 4012 and 4023 terminals, and ASR33.

T A B L E O F C O N T E N T S

SECTION 1

Introduction

PROC10, an interactive image processing system, currently runs on the PDP10 at the National Institutes of Health. It can manipulate picture, mask, boundary, boundary transform and computing window data structures. PROC10 provides a wide range of operations on and between these data structures. Images and boundaries may be displayed on several different types of terminals including the DEC GT40, Tektronix 4012 and 4023 terminals, and ASR33. This document is oriented toward describing the system from the point of view of a user. PROC10 is written in PDP10 SAIl [VanL73].

It is an enhanced version of the PROCES interactive image processing program [Lem75] which runs on the PDP8e part of the Real Time Picture Processor (RTPP) ([Lem74], [Carm74], [Lem76a]) at the Image Processing Unit, National Cancer Institute. PROC10, written in SAIL [VanL73], includes all of the RTPP PROCES operations. The philosophy of the PROC10 system is that one brings image, mask or boundary DDTG [Lem76b] data files into core, operates on them and then saves them as DDTG formated data files (to be discussed). Alternatively, data may be read into the program as Ascii numbers (to be discussed) and converted to the format used by PROC10. The final state of a PROC10 session may be saved (including images, masks etc.) and reentered later.

PROC10 can display gray scale images and boundaries on any of four terminals: ASR33 type terminal, Tektronix 4012 (see [Gor76]) or 4023, or DEC GT40 as well as printing images on a lineprinter in various dump modes. Pictures, boundaries and transforms may be written out as PDP10 files. Picrutures and boundaries are in a format which can be read by the RTPP for display on the DICOMED or video displays.

Commands are entered one per line from the display terminal teletype. Alternatively, sequential commands may be entered indirectly via a command file using the DO <file specification> construction.

The system contains many user aids. Type HELP to get further instructions. To get a list of all of the possible commands, type CMD followed by the carriage return key. All commands are terminated by typing the return key.

To find out the specific syntax and semantics of a particular command, type

HELP <specific command>

Commands with the "TOGGLE" postfix act to reverse the current sense of the switch associated with the command. For

example, TIMERTOGGLE turns the command execution time timer on and off with successive invocations.

Commands may be entered as short forms with enough letters (minimum of 3) for the system to guess it uniquely. Further prompts are given if required. Commands are of two distinct forms as expressed in the following BNF-like specifications (where ' _ ' is the backarrow character):

```
[1]     <command> <arguments separated by spaces or commas>
  or
        <command> <name>

[2]     <name>_<name><operator><name>,<opt. args>,<opt. mask>
  or
        <name>_<unary operator><name>,<opt. args>,<opt. mask>

Where:  <name>::= <DSK: file> | <window> | <pix> | <mask> |
                  <boundary> | <boundary transform>
        <window>::= W1 | W2 | .. | W32
        <pix>::= P1 | P2 | .. | P32
        <mask>::= M1 | M2 | ... | M32
        <boundary>::= B1 | B2 | ... | B32 | <segment boundary>
        <segment boundary>::= B33 | B34 | ... | B300
        <boundary transform>::= T1 | T2 | ... | T32
```

1.1 Automatic data structure allocation

As is common in other interactive languages (such as MLAB [Knott73], PRDL [Lem76c]) the use of a data structure name as an output operand (such as P1 in P1_READ MAR001.PIX) will cause computer storage to be allocated for that data structure the first time it is used. This eliminates the need to either allocate storage when the system is loaded or to have the user request (via the program) storage explicitly. This implicit storage allocation holds for pictures, masks, boundaries, boundary transforms and computing windows. This storage is automatically allocated using the LEAP 'ITEMS' with associated integer array datums (For those who are interested in this mechanism, see SAIL manual for discussion of items and datums [VanL73].).

While data structures are automatically created when the name of the structure is used as an output operand, they must be explicitly deleted using the DELETE command. If a data structure such as P3 contains an image which is no longer needed and the user wishes to use P3 to hold the result of another operation, then by just using it in that operation the old contents of P3 are replaced with the new contents. Thus the DELETE command should be used only when no further use of an image or boundary structure is intended.

1.2 Image data structures

Images are referenced by their picture name Pi (i in the interval [1:32]) and are power of 2 size square integer arrays of density values with the largest size being 256x256 pixels and the smallest 16x16. The pixels are packed 4 pixels/PDP10 36-bit word (so that a 256x256 (65K pixels) occupies 16K PDP10 words.) Thus a pixel may have a maximum gray value of 511 taking 9-bits.

1.3 Logical Coordinate System - LCS

The Logical Coordinate System (LCS) of image coordinates is that used by the RTPP and DDTG display systems. The LCS is used to position an image on the display terminal. The LCS has the origin (x,y) = (0,0) at the upper left-hand corner and (x,y) = (1023,1023) at the lower right-hand corner (as a maximum which only exists on certain devices - see [Lem76a]). Positive X is to the right and positive Y is down. There are no negative coordinates. The LCS is specified by setting the upper-left hand corner of the image using the SETLCS <x>, <y> command in PROC10. The default LCS value is (0,0) for the ASR33, TK4023 and (256,256) for the DEC GT40 and Tektronix 4012. The latter default is done to allow the user room on the screen to type a few PROC10 command lines starting at (0,0) after an image is displayed.

Not to be confused witht the full LCS just described is a relative LCS of size [0:n-1,0:n-1] where n is the size of the image. The relative LCS is used in seting the computing window and in the SEGMENT operations as well as other operations.

1.4 Density values

Various operations use density threshold values as inputs. These are supplied explicitly in the arguments of the operation. A global threshold value is available for use by some of the commands (SLICE, AREA, DENSITY, PERIMETER, MSLICE, etc). It is set via the SETDENSITY command or in the EXTREMA operation and used in for a command by specifying the USETHReshold switch. SETDENSITY permits setting the global density threshold.

The maximum computing density value is the largest number (which is a power of 2 less 1, i.e. 1, 3, 7, 15, 31, 63, 127, 255, 511) to be allowed in a gray scale calculation. It can not be larger than 9 bits (511). It is used in all gray scale operations where a new gray scale value is computed such that all computed gray values are clipped to this maximum computing density value. It is specified using the SETDENSITY command and has a default of 255.

Images as defined internally in PROC10 can store up to 9-bits of gray scale, while image files can store up to 8-bits maximum. The density mapping is 0 (minimum density) is white and 255 (maximum density) is black). Internally, grayscale arithmetic operations are performed with clipping such that if the resultant gray value is < 0 it is set to 0; if it is > the maximum computing value it is set to the maximum computing value.

The SHOW command uses two dynamic-range density values (dmax,dmin) which are defined with the SETDENSITY command. The default (dmax,dmin)=(255,0).

Gamma correction is performed using the scaling factor (set using the SETSCALING command). The default scaling is 0. No gamma correction is performed if the scaling is 0. That is, for a display with N possible display gray levels (including blank) the dynamic range of the data is adjusted so that for display level d(x,y) and gray level g(x,y):

```
If scaling=0
    Then d(x,y)=(N/(dmax-dmin))*(g(x,y)-dmin)
    Else d(x,y)=N*scaling*(g(x,y)/(dmax-dmin))
```

1.5 Computation windows

Operations are performed over a computation window defined by the 4-tuple (firstrow, lastrow, firstcolumn, lastcolumn). That is, the computation is performed only over that rectangular region of the image upon which the window covers. The computation window may be saved and restored under the names Wi (i in [1:32]) using the SAVEWINDOW and GETWINDOW commands. All image operations are performed over the computation window. Initially, this window is set to the full picture size (256x256) but may be changed by the SETWINDOW, SETSIZE, or FINDWINDOW operations. If masks are used with picture operations, the actual computation domain is the logical AND of the two.

A default window of 256x256 pixels is used throughout the PROC10, RTPP, and DITG systems. When the image size is changed in PROC10 to another size NxN using the SETSIZE <N> operator, then the computing window is reset to the full window (NxN) for the new image size. Note that N will be rounded up to the first power of 2, such that N is less than or equal to 2**j. This window is the default computing window in PROC10. To avoid confusion, it is noted here that when sampling is mentioned in PROC10, it refers to the sampling for the display but not for general computation, sampling is only done on display operations and in the ZOOM operation.

1.6 Mask data structures

Masks are power of 2 size square arrays (as are images) of 0's and 1's. Most picture operations may be performed under a mask which is to say that only those parts of the images lying within the mask will be operated upon. If a mask Mi is mentioned in an image operation then the operation is performed on only those pixels for which Mi(r,c)=1. An example of the syntax is:

```
P2 _ GRAD8 P1,M7.
```

Masks may be created using various mask generators some such as MCIRCLE, SPHERE, SQUARE and MRECTANGLE are parametrically specified while others such as MSLICE and MSEGMENT create masks from performing operations on images.

1.7 Resultant images

The image resulting from an image operation is a displayable gray scale image. The one exception is the SEGMENT operator. This operator produces an image consisting of labeled components which have connected region pixel values corresponding to the component number. Component numbers are in the range [1:253] whereas background pixels are defined by a 0 value. The mask MSEGMENT operator will generate a mask from such a connected component image given the image and the number of the connected component. That is, those portions of the segmented image containing the desired component number will cause a 1 to be placed in the corresponding mask position.

The resultant pixels of image and mask operations are a subset of all possible pixel locations in the output image or mask (i.e. of maximum image size). In the case of pictures, this domain is restricted by the computing window and optional masks. In the case of masks, the domain is restricted by the computing window and the mask generator if used. Thus, if the same output operand (such as P3 or M4 etc.) is used for several different operations, then a composite image or mask is created. This is possible since performing an operation on an existing output operand does not zero it before doing that operation.

1.8 Boundary and boundary transform data structures

A boundary is a 2xN array defined by the user with name Bi (i in [1:32] for user defined boundaries) or Bi (i in [33:300]) defined by the SEGMENT operation where n is the number of boundary pixels.

The picture operator MAKPIX maps a one-dimensional

boundary Bi into a two-dimensional picture by coloring in pixels in Pi. Boundaries may be operated on using either boundary or boundary transform operators. The latter have data structure names (T1, T2, ...). The transform operators include the circle, complex and centroid Fourier, and centroid Walsh transforms ([ShapB76a], [ShapB76b]) as well as their inverses.

1.9 Display of Images and boundaries

PROC10 offers two display modes (not to be confused with types of display terminals). These are gray scale display of images invoked by doing SHOW Pi, and line drawing displays of boundaries (or parts of boundaries) invoked by doing SHOW Bi. The parameter 'sampling distance' (set by SETSAMPLING) is used to specify the picture sampling distance for doing a grayscale image SHOW. For gray scale images, as was mentioned before in the section on density values, linear or gamma corrected density mapping is performed on the image gray values to form the display density values.

The command PRINT Pi causes pictures within the computing window to be printed on the lineprinter or teletype as either 8 level gray scale, single character hexidecimal or as (0-511) decimal numbers. Both SHOW Pi and PRINT Pi will do output only from that part of Pi inside the computing window.

1.9.1 Omni pictures

If either the GT40 or Tektronix 4012 display terminals are used, it is possible to have more than one picture displayed at a time. This mechanism is implemented using the PDP10 OMNIGRAPH software [CCB76]. In OMNIGRAPH, pictures correspond to data structures called 'display files'. In PROC10, these display files are referenced by Pi or Bi data structure names. The PARAMETERS command will list the names of the active posted (displayed) and unposted display files which have been taken off of the display.

The PROC10 default is to have only one active OMNI picture at a time. To display multiple pictures (boundaries, or histograms), the AUTOCMNITOGGLE command may be invoked. When on, every displayable data structure picture, histogram or boundary will be assigned a unique Omni picture display file when that structure is requested to be displayed. Furthermore, the image will remain on the display screen until a request is made by the user to unpost or kill it (UNPOSTOMNI or KILLOMNI). An unposted Omni picture may be restored to view by doing a POSTOMNI on the previously unposted picture. In the case of the Tektronix 4012 terminal, these display commands are not activated until the DOOMNI command is invoked.

1.9.2 PROC10 Movies

It is possible to construct a sequence of picture frames which may be displayed in a particular order. This structure is called a movie. Only one movie may be constructed at a time. The movie can be shown on any terminal. However, if a GT40 is used and Omni numbering is on, once pictures are sent to the GT40, future showings of the movie will preceed at a rapid rate. If on the otherhand the display is not setup like this, then the pictures must be transmitted from the PDP10 frame by frame and no rapid frame change rate is possible.

1.10 State of PROC10

The state of PROC10 in between picture operations may be determined by the user. The command PARAMETERS may be used to print the values of the density setting, active computing and sampling windows; the display terminal type; lists of posted, unposted and movie picture names; and state of the automatic titling and OMNI switches.

The command ACTIVEDATA may be used to print the names, titles, and associated parameters of the pictures, boundaries, masks and saved computing windows defined in the system to date.

Various other parameters may be set using one of the SET- commands such as SETDENSITY, SETTITLE etc. Typing a SET- command without arguments causes the old parameter values to be typed followed by a request for the new values (if different).

1.11 Data File Format

Pictures are (2**N)x(2**N) pixel 8-bit gray level images. Masks are (2**N)x(2**N) pixel 1-bit images. Boundaries are variable length arrays of (x,y) 8-bit bytes with a (0,0),(0,0) at the end of the list to denote end of list. Transforms are one-dimensicnal 36-bit arrays with every N words of data constituting an entry depending on the type of transform being represented.

PROC10 is able to read PDP10 data files in both DDTG format and as a list of Ascii numbers. The former mode incorporates a file header which serves as a check on the validity of the data (since data file typing is used) as well as providing the data structure size (i.e. image size) and title information. The NUMBER mode (used with the READ command) allows access to PROC10 from scanners, etc., different from those producing DDTG formated files. NUMBER input mode accepts (2**Nx2**N) Ascii gray values specified in the file as numbers separated by spaces or commas where carrage return/line feeds are ignored. The size 2**N must be set previous to

executing the READ command by doing a SETSIZE command. PROC10 only writes DDTG formatted files.

Picture files are currenly being transmitted between the RTPP and the PDP10 via PDP10 DECTAPE or MAG10/MAG8e [Shap76c].

The Optronics rotary scanning densitometer produces raster scan data with windows much larger than 256x256. A program written by B. Trus and R. Gordon called OP.EXE[606,552] with command file OP.CMD[606,552] is used to extract 256x256 pixel square windows for use as "NUMBER" formated files for input to DDTG.

The DDTG format is described in the DDTG users manual [Lem76b] as well as IO.SAI in a table listing the formats and characteristics of the different data file types. It is fairly easy to write conversion programs to take power of 2 size gray scale images in other formats and generate DDTG compatible files using procedures in IO.SAI. Such files could then be accessed by PROC10.

1.12 Neighborhood definition

The current neighborhood of an image is a 3x3 rectangular array of image gray values which is tesselated through the image according to the particular operation. The pixels are labeled as follows for the PROC10 operations which will be discussed:

```
3 2 1
4 8 0
5 6 7.
```

1.13 Border definition in neighborhood operations

In certain neighborhood operations which map neighborhoods to pixels (such as GRAD4/8, LAPLACE8, EXPAND, SHRINK, etc.) the border pixels of the computing window are not defined. The system default is to set these pixels to zero. A better default which may be used in the future is to set these border pixels to adjacent values.

1.14 PROC10 command files

All teletype command input to PROC10 may be specified indirectly through a PDP10 command file. The DO <file specification> command is used to direct PROC10 to accept input from the specified file rather than from the teletype. Command files are useful in applying a sequence of operations to an

image after the sequence has been interactively developed.

The ENTER2001 and LEAVE2001 commands allow the user to go back and forth between PROC10 and the PDP10 monitor (via a pseudo teletype) without detaching the PROC10 job. DO files can then easily be created or modified with a text editor such as TECO or SOS while in 2001 mode.

1.15 Addition of new operations and data structures to PROC10

Currently, new operations are added by editing the new procedures and command information into the PROC10 parser and interpreters as well as implementing the actual operation in a 'worker' package. The system must be recompiled, loaded and saved.

1.16 SAIL modules used in PROC10

The following list of PDP10 files are required to build a PROC10.EXE core image file. The files are listed here in a tree-like manner to reflect the structure of the system (to some extent). Some modules such as PRCMAX, PRCINV, PRCWRK etc. are used by many other modules so that this structure is not apparent.

[1] Globally used files
PROC10.SAI - the main
DEFINE.REQ - macro definitions used everywhere
GETABL.SAI (.REQ) - get break table number
IO.SAI (.REQ) - DDTG/PROC10 data file I/O pkg
BOUND.SAI (.REQ) - user bounded input pkg
        DARRAY.SAI (.REQ) - display histogram on OMNI terminal
        ARINFO.SAI (.REQ) - ITEM array debugging procedure
        CVADJ.SAI (.REQ) - array debugging procedure
        CVT.SAI (.REQ) - floating string conversion with
                no trailing 0's
SYS:DISPRM.SAI - OMNI External declation pkg
SAIFIX.MAC - SAIL fix.

[2] The following are the worker routines and external variables for the mini-interpreters
PRCMAX.SAI (.REQ) - macro definitions of array sizes
PRCINV.SAI (.REQ) - global INTERNAL variables
PRCWRK.SAI (.REQ) - data structure alloc/dealloc, parser

        [2.1] Special command interpreter
        SINTRP.SAI (.REQ) - interpreter
                SPAK.SAI (.REQ) - worker routines
                SAVER.SAI (.REQ) - reenter pkg
                PTYPKG.SAI (.REQ) - pseudo TTY: pkg

        [2.2] Picture operator interpreter
        PINTRP.SAI (.REQ) - interpreter

```
            PPAK.SAI (.REQ) - picture and mask operations
            HLFTON.SAI (.REQ) - gray scale display pkg
                    GTDISP.SAI (.REQ) - GT40 handler
                    TK4012.SAI (.REQ) - TK4012 handler
                    TK4023.SAI (.REQ) - TK4023 handler
                    CROSSH.SAI (.REQ) - TK4012 cross hairs
            PIXDMP.SAI (.REQ) - picture LPT: dumper

    [2.3] Mask operator interpreter
    MINTRP.SAI (.REQ) - interpreter
            (Also uses PPAK)

    [2.4] Boundary and arc operator interpreter
    BINTRP.SAI (.REQ) - interpreter
            1DPAK.SAI (.REQ) - boundary/boundary-
                                    transforms pkg
            LINPAK.SAI (.REQ) - geometric operations pkg
                    ANGNRM.F4 - Fortran angle normalization
                    FORT.SAI (.REQ) - SAIL EQV of some
                            Fortran builtin functions
                    CPAK.SAI (.REQ) - Complex arith. pkg
            BDISP.SAI (.REQ) - boundary display pkg
```

The system consists of a simple line-scan parser (ANALYZE!CMD in PRCWRK.SAI) which detects data structure types both by the operator and the operand character (ie. Pi is picture, Bi is boundary, etc.). Semantic checking is performed to discriminate what data type is involved when operator names have more than one meaning (as with ZERO, DELETE, etc.).

Each data structure has a separate sub-interpreter (SINTRP, PINTRP, MINTRP, BINTRP) except in the case of boundary transforms (Ti) which are included in the boundary interpreter (BINTRP) since the two data types are closely tied together. Each interpreter maps the parsed scan line from string space to item space and then dispatches the specified operation (generally to a procedure in a worker package such as SPAK, PPAK, OR 1DPACK) via a CASE statement.

The system is rebuilt as follows:

[1] Compile each .SAI file with (H) sharable switch.
[2] Load the system with LOAD PROC10
[3] Save the system with SAVE DSK:PROC10

SECTION 2

PROC10 Operators

The following four subsections list the special, picture, mask and boundary operators. Note that some operators such as READ, WRITE, DELETE, SHOW, etc. appear in several of these operator lists. The particular command implied is determined by the context in which the operator is used. Thus DELETE W1 will delete window #1 while DELETE P2 will delete image 2.

2.1 PROC10 Special Commands

PROC10 special commands operate across all data structures or are generally data structure independent, compared to the specific picture, mask and boundary operators to be discussed in later sections.

2.1.1 HELP

The HELP command is used to supply information about the PROC10 system in general (no arguments used) or the syntax and semantics of specific commands. In the latter case, the HELP is followed by the command in question. The file PROC10.HLP is searched for numbered paragraphs for the specified command. If the command does not exist in PROC10 a message to that effect is printed.

HELP <Opt. command name>

2.1.2 ACTIVEDATA - STATE

Print the names and related parameters of the entire data base, selected data structures, or of a particular data structure (i.e. P3 or B6 etc). If a boundary was created using the SEGMENT operation, then an association exists between the resultant connected component image and each boundary associated with a connected component. This association has a property list consisting of (component number, first row of boundary, first column of boundary, area, number of boundary pixels, density, boundary name, touching computing window predicate, component image name). The property list for each associated boundary is printed with the connected component image information.

ACTIVEDATA <opt. Pix, Mask, Boundary, Transform or Window> --or--
<Opt. specific Pi, Mi, Bi, Ti, or Wi>

2.1.3 AUTOTITLETOGGLE

AUTOTITLETOGGLE turns the automatic titling on or off each time the command is executed. When on, auto-titling will use as the title of the current output data structure the command used to generate it. When it is off, the title is requested from the user.

AUTOTITLETOGGLE

2.1.4 CMD - HELP

CMD prints the lists of all the legal PROC10 commands available. It may also be used with a modifier to print a subset of these commands.

CMD <Opt. Commands or Picture or Mask or Boundary>

2.1.5 DELETE

DELETE deletes from the data base the previously saved computing window Wi. The saved window Wi was the 4-tuple (first row, last row, first column, last column).

DELETE <Window Wi>

2.1.6 ENDSESSION

The ENDSESSION command exits PROC10 and return to the PDP10 monitor. The core image may then be saved at PDP10 monitor level by doing a SAVE <session file name> or NSAVE <session file name>. The session core image may be restarted at a future time by doing a GET <session file name>, REENTER PDP10 monitor command sequence.

ENDSESSION

2.1.7 GETWINDOW

GETWINDOW is sued to restore a previously saved computing window. It sets the current computing window (first row, last rows, first column, last column) and image size to that of the previously saved computing window Wi.

GETWINDOW <Window Wi>

2.1.8 PARAMETERS

The PARAMETERS command prints the values of various parameters which comprise the state of the PROC10 system. These include: WHITENOISE mean and standard deviation of the generator densities; global threshold; SHOW min and max display densities, display sampling distance, display terminal type, gray scale scaling ratio (for display); computing window; autotitling switch; autoCMNI numbering switch; pictures in the post, unpost and movie lists.

PARAMETERS

2.1.9 SAVEWINDOW

SAVEWINDOW saves the current computing window (first/last rows, first/last columns) and image size (a title for this saved state will be requested) as a window data structure.

SAVEWINDOW <Window Wi>

2.1.10 SETDENSITY

Various density related parameters are used in many PROC10 operations. In particular, SETDENSITY may set the global threshold (for use with SLICE, AREA, DENSITY, etc.) and maximum black density in bits (i.e. 8 bits is 256 gray levels). It also sets the min and max display densities used in the SHOW operation.

SETDENSITY <thresold dens>, <compute density precision>,
<min display density>, <max display density>

2.1.11 SETSAMPLING

The display sampling distance refers to the inter-pixel distance used in image displays with the SHOW command. If it is >0 then the sampling square with the sample pixel in the upper left hand corner averaged and this average displayed. If the sampling distance is <0 then the sample pixel is displayed without averaging. Note that when sampling is done, thin edges may not appear due to the position of the edge relative to the sampled pixel.

SETSAMPLING <distance: (>0) to avg, (<0) to not avg display)

2.1.12 SETSIZE

Image computations are performed on images of fixed sizes. The current image size must be a power of 2. This size is set using the SETSIZE command which sets the working image size to the nearest power of 2 which will hold the image of the size specified. The computing window is opened up to the full window for this computed power of 2.

SETSIZE <Image size i.e. 16 through 256>

2.1.13 SETTERMINAL

The SETTERMINAL command sets the current display device to the type specified. Note that if your terminal is neither a DEC GT40, Tektronix 4012 or 4023 then assign the ASR33 teletype like terminal. The DEC GT40, Tektronix 4023 and ASR33 have 8 simulated density levels while the Tektronix 4012 has 17 simulated density levels.

SETTERMINAL <Terminal type ASR33, GT40, 4012, 4023>

2.1.14 SETTITLE

Every data structure (picture, boundary, mask, boundary transform, window) has an associated title which is used during I/O, display (via SHOW), or during state interrogation (ACTIVEDATA). The SETTITLE allows the user to change (or specify) the title of the specified data structure.

SETTITLE <Picture, Mask, Boundary, Transform,
or Window name>

2.1.15 SETWINDOW

The computing window is a 4-tuple (first-row, last-row, first-column, last-column) which is used in most image and mask operations to specifiy the region of active computation. It may be specified or changed using the SETWINDOW command.

SETWINDOW <First row, last row, first column,
last column>

2.1.16 SETLCS

The position of the display window upper left hand corner is defined in the logical coordinate system (LCS). Theupper left hand corner of the image is denoted by (Xp,Yp) in PROC10. The LCS as was mentioned before has (0,0) as the upper left hand corner and (1023,1023) as the lower right hand corner. The SETLCS command changes the values of (Xp,Yp). The Tektronix 4012 display has an active LCS region (0:779,0:779), the DEC GT40 (0:769,0:769), and the Tektronix 4023 (0:80,0:24). As the range of the LCS for the ASR33 and Tektronix 4023 is very small it only makes sense to change the LCS for the DEC GT40 or Tektronix 4012.

SETLCS <Display XP upper L.H.C.>,<YP upper L.H.C>

2.1.17 TERSE

Two question and answer interaction modes are available (verbose and terse). The TERSE command sets PROC10 to a terse question and answer mode. Terse mode is the default.

TERSE

2.1.18 TIMERTOGGLE

TIMERTOGGLE turns the timing switch on or off each time it is executed. When on, it prints the CPU, RUN and (CPU/RUN)*100% times taken for each user directed operation.

TIMERTOGGLE

2.1.19 VERBOSE

Two question and answer interaction modes are available (verbose and terse). The VERBOSE command sets PROC10 to a verbose question and answer mode for more informative prompts. Terse mode is the default.

VERBOSE

2.1.20 DO - I/O

The DO command executes an Ascii command file which contains a list of commands and expected responses to command questions. It replaces the user commands and responses usually entered from the teletype. The use of the Do files facilitates the implementation of image processing procedures.

DO <Opt. Dev:><Command file><Opt. [Proj,Prog]>

2.1.21 ENTER2001

It is sometimes useful to leave the PROC10 command structure and enter the PDP10 monitor. This is done by issuing the ENTER2001 command and having subsequent teletype I/O communicate with the PDP10 monitor through a pseudo teletype. Upon entering the pseudo teletype connection all output send to you by the PDP10 is prefaced with a "<" and all input expected of you is promted by a ">". Having set up the pseudo teletype channel, you may log in again. This permits the interrogation of directories, text editing, etc. without leaving the PROC10 core image. Typing LEAVE2001 returns you to PROC10. Don't forget to logoff while talking to the PDP10 monitor before returning to PROC10. Also, if you control/C out of 2001, the 2001 job you may have created will become detached.

ENTER2001 (type LEAVE2001 to return)

2.1.22 AUTOOMNITOGGLE

The AUTOOMNITOGGLE command turns the automatic Omni 'display file' generation switch on or off each time the command is executed. When on and the DEC GT40 or Tektronix 4012 displays are being used, each picture is assigned a unique OMNI display file. This permits having more than one picture on the display at a time.

AUTOOMNITOGGLE

2.1.23 KILLOMNI

Omni pictures created when the AUTOOMNITOGGLE is on may be deleted using the KILLOMNI command. This will delete the omni picture specified. If the ALL switch is specified, then delete all Omni pictures and clear the screen. It is primarily used when the Tektronix 4012 or DEC GT40 display is active.

KILLOMNI <OMNI picture name>, <Opt. ALL switch>

2.1.24 UNPOSTOMNI

The UNPOSTOMNI command temporarily removes the specified Omni picture currently posted on the display (DEC GT40 or Tektronix 4012 displays only).

UNPOSTOMNI <OMNI picture name>

2.1.25 POSTOMNI

The POSTOMNI command restores the Omni picture specified which had previously been unposted with the UNPOSTOMNI command.

POSTOMNI <OMNI picture name>

2.1.26 DOOMNI

The DOOMNI command causes POSTOMNI and UNPOST omni picture operations to be performed which were previously stacked for use on a Tektronix 4012 terminal. This command only makes sense for boundaries. Note that DOOMNI is not needed for DEC GT40 operation for which it is a null operation.

DOOMNI <Opt. ERASE before doing Kill, Post
cr Unpost commands>

2.1.27 SETSCALING

The SETSCALING command sets the gray scale display scaling ratio between max/min densities for nonlinear scaling. If it is set to 0 then linear scaling is used. The default value is 32. Note that if a display has N density levels, the blank is included. Gamma correction is done when the scaling value is non-zero. The correction is performed after averaging (if averaging is used). The correction is as follows, for N display density levels; dmax, dmin (display densities), for display density d(x,y), and clipped pixel gray value g(x,y) (to dmin:dmax):

If scaling=0
Then d(x,y)=(N/(dmax-dmin))*(g(x,y)-dmin)
Else d(x,y)=N*scaling**(g(x,y)/(dmax-dmin))

SETSCALING <Opt. scaling ratio between max/min,
0 for linear>

2.1.28 NEWMOVIE

A movie consists of an ordered sequence of frames. To clear this list in preparation for making a new movie, the NEWMOVIE command is issued. It deletes the existing movie frame list consisting of picture or boundary data structure names.

NEWMOVIE

2.1.29 APPENDMOVIE

The APPENDMOVIE command adds the list of OMNI picture names to the end of movie list. The movie will show picture frames in the order that they are appended. Non-Omni picture names may be specified (i.e. Pi or Bi which were never shown with AUTOOMNITOGGLE on) but a warning message will be printed.

APPENDMOVIE (List of OMNI picture names)

2.1.30 RUNMOVIE

The RUNMOVIE command shows the list of frames on the current display terminal. The frames may have been transmitted previously for rapid motion (on the DEC GT40), otherwise (for the ASR33, Tektronix 4023, Tektronix 4012 terminals) the picture will be retransmitted each frame.

RUNMOVIE <Opt. n, (do every n'th frame)>

2.1.31 SPLICEMOVIEFRAME

The SPLICEMOVIEFRAME command permits inserting a specified Omni picture after another specified frame.

SPLICEMOVIEFRAME <(after) Frame #>,<Omni picture name>

2.1.32 REMOVEMOVIEFRAME

TheREMOVEMOVIEFRAME permits removing a specified Omni picture from the movie.

REMOVEMOVIEFRAME <Frame #>

2.1.33 SETBOUNDARYSCALEFACTOR

The SETBOUNDARYSCALEFACTOR command permits the expanding or shrinking of a boundary displayed on the tektronix Tektronix 4012 or DEC GT40. The scale factor may be set from 0.1 to 10 times the size. The default display size is 1.

SETBOUNDARYSCALEFACTOR <0.1 to 10X default 1>

2.2 Picture Operators

The following operators have image data structures as the domain of their major operands. The output image (if required) may be a previously created image. Alternatively, it may be a newly created image with default gray scale values of zero.

Operations are performed inside of the computing window and the logical AND of the computing window and mask if a mask is specified. Output image pixels not operated on are not changed. Thus, by using different computing windows and/or masks, different parts of an output image may be created by different image operators.

2.2.1 +

Two images consisting of pixels Pj(r,c), Pk(r,c) may be added pixel by pixel such that (r,c) is selected from the computing window (and also in mask Mi if Mi is specified) and the resulting ((Pj(r,c)+Pk(r,c)) Min maximum computing density) stored in Pi(r,c).

<Pi> _ <Pj> + <Pk>, <Opt. Mask Mi>

2.2.2 MINUS

Two images consisting of pixels Pj(r,c), Pk(r,c) may be subtracted pixel by pixel such that (r,c) is selected from the computing window (and also in mask Mi if Mi is specified) and the resulting ((0 Max ((Pj(r,c)-Pk(r,c))) Min maximum computing density) stored in Pi(r,c).

<Pi> _ <Pj> MINUS <Pk>, <Opt. Mask Mi>

2.2.3 DIFFERENCE

The absolute pixel difference between two images consisting of pixels Pj(r,c), Pk(r,c) may be computed such that (r,c) is selected from the computing window (and also in mask Mi if Mi is specified).The resulting ((0 Max ((|Pj(r,c)-Pk(r,c)|)) is tested agains the specified threshold. If it is less than the threshold then 0 is stored otherwise the absolute differnce just computed.

<Pi> _ <Pj> DIFFERENCE <Pk>, <threshold>, <Opt. Mask Mi>

2.2.4 *

To images consisting of pixels Pj(r,c), Pk(r,c) may be multiplied pixel by pixel such that (r,c) is selected from the computing window (and also in mask Mi if Mi is specified) and the resulting (((Pj(r,c)*Pk(r,c)) Min maximum computing density) stored in Pi(r,c).

<Pi> _ <Pj> * <Pk>, <Opt. Mask Mi>

2.2.5 /

Two images consisting of pixels Pj(r,c), Pk(r,c) may be divided such that (r,c) is selected from the computing window (and also in mask Mi if Mi is specified) and the resulting ((0 Max ((Pj(r,c)/Pk(r,c))) MIN maximum computing density) value stored in Pi(r,c). If the divisor is 0 (i.e. Pj(r,c)/0), then Pi(r,c) is set to the maximum density.

<Pi> _ <Pj> / <Pk>, <Opt. Mask Mi>

2.2.6 MAX

A maximum picture Pi is created by taking the maximum at each pixel in two image Pj and Pk. Each pixel at (r,c) is selected from the computing window (and also in mask Mi if Mi is specified) and the resulting [Pj(r,c) Max Pk(r,c)] stored in Pi(r,c).

<Pi> _ <Pj> MAX <Pk>, <Opt. Mask Mi>

2.2.7 MIN

A minimum picture Pi is created by taking the minimum at each pixel in two image Pj and Pk. Each pixel at (r,c) is selected from the computing window (and also in mask Mi if Mi is specified) and the resulting [Pj(r,c) Min Pk(r,c)] stored in Pi(r,c).

<Pi> _ <Pj> MIN <Pk>, <Opt. Mask Mi>

2.2.8 SCALE

An image Pj may be multiplied by a scaler using the SCALE command. SCALE multiplies pixels in image Pj by the positive scaler <value> and stores the resultant pixels in image Pi such that the pixels operated on are inside the computing window (and the mask Mi if mentioned). The resultant pixel value Pj*scaler is clipped at the maximum computing density.

<Pi> _ <Pj> SCALE <scalar value>, <Opt. Mask Mi>

2.2.9 ROTATE

ROTATE rotates pixels in the Pj image about (row center, Col center) the specified angle and stores them in Pi(r,c) such that (r,c) is inside the computing window (and under mask Mi if specified). The transformation is:

dc = c - col center,
dr = r - row center,
r' = row cent + dc*COS(angle) + dr*SIN(angle),
c' = col cent + dr*COS(angle) - dc*SIN(angle),
r''= (image size Min r') Max 0,
c''= (image size Min c') Max 0,

then,

Pi(r'',c'') = Pj(r,c)

for all (r,c) inside of the computing window and mask Mi (if specified). Thus pixels in (r,c) which would be outside of the range [0:image size] are mapped to the corresponding extrema.

Such a transformation is sensitive to distortions introduced by the use of a square image sampling function.

<Pi> _ <Pj> ROTATE <Row cent>,<Col cent>,<angle in degrees>

2.2.10 COPY

COPY copies image Pj into image Pi such that only those pixels inside of the computing window (and inside of the mask Mi if mentioned) are copied.

<Pi> _ COPY <Pj>, <Opt. Mask Mi>

2.2.11 AVG4

AVG4 averages the four-neighbor pixels of Pj(r,c) and stores the local average into Pi(r,c) for (r,c) in the computing window (and in the mask Mi if mentioned). That is: I8=(I0+I2+I4+I6)/4.

<Pi> _ AVG4 <Pj>, <Opt. Mask Mi>

2.2.12 AVG8

AVG8 averages the eight-neighbor pixels of Pj(r,c) and stores the local average into Pi(r,c) for (r,c) in the computing window (and in the mask Mi if mentioned). That is: I8=(I0+I1+I2+I3+I4+I5+I6+I7)/8.

<Pi> _ AVG8 <Pj>, <Opt. Mask Mi>

2.2.13 GRAD4

GRAD4 takes the four-direction 8-neighbor gradient for pixels of Pj(r,c) and stores the local gradient into Pi(r,c) for (r,c) in the computing window (and in the mask Mi if mentioned). The GRAD4 computes the four direction vectors:

```
    1  2  1      -1  0  1       0  1  2       2  1  0
    0  0  0      -2  0  2      -1  0  1       1  0 -1
   -1 -2 -1      -1  0  1      -2 -1  0       0 -1 -2
       Dx            Dy            D45           D135
```

Then Pi(r,c)=Max(|Dx|,|Dy|,|D45|,|D135|). If the DIRECTION option is used, then Pi(r,c) =1:4 where 1 is Dx, 2 is Dy, 3 is D45, and 4 is D135.

<Pi> _ GRAD4 <Pj>, (Opt. DIRECTION switch),
<Opt. Mask Mi>

2.2.14 GRAD8

GRAD8 takes the eight-neighbor gradient used by Kirsch for pixels of Pj(r,c) and stores it into Pi(r,c) for (r,c) in the computing window (and in the mask Mi if mentioned). If the DIRECTION switch is used, the output image consists of the direction of the gradient at each pixel. This is coded as a 0 where no gradient was taken, 1:8 being the chain code direction (as specified in the paragraph defining the 'neighborhood') plus 1. That is:

For j=[0:8],
Let Sj = 5*|I(j) + I(j+1) + I(j+2)| -
3*|I(j+3) + I(j+4) + I(j+5) + I(j+6) + I(j+7)|,

Then,

I8 = MAX(S0,S1,...,S7).
direction = index of maximum direction + 1;

The maximum GRAD8 value is computed as well. If after the operation is completed, the maximum gradient value is > maximum allowed gray value then the GRAD8 must be recomputed with each GRAD8 value multiplied by a scale factor (max allowed gray value/previous maximum GRAD8 value) before it is stored in the output image.

<Pi> _ GRAD8 <Pj>, <Opt. DIRECTION>, <Opt. Mask Mi>

2.2.15 FILLPINHOLES

Often, an image will have salt and pepper (shot) noise. This appears as pinholes in an image. The FILLPINHOLES operations will remove all pinholes which deviate from their 4-neighbors by a specified density difference and replace that pixel with its 8-neighbor average. Boundary points are not checked. For a normal image a density difference of 20 would be a good first approximation. The algorithm is approximated by:

```
density=If |I8-I0| > d Or |I8-I2| > d Or
          |I8-I4| > d Or |I8-I6| > d
               Then
               (I0+I1+I2+I3+I4+I5+I6+I7)/8.
<Pi> _ FILLPINHOLES <Pj>, <Density difference>,
          <Opt. Mask Mi>
```

2.2.16 SLICE

For all pixels Pj(r,c) such that the grayscale value g(r,c) of Pj where [minimum density < g(r,c) leq maximum density], set Pi(r,c) to Pj(r,c) otherwise set Pi(r,c) to zero inside of the computing window (and mask Mi if used). If the USETHReshold switch is used, then the current threshold set with SETDENSITY or EXTREMA is used as the minimum density and the maximum computing density is used as the upper bound.

```
<Pi> _ SLICE <Pj>,(USETHReshold switch
          or <dens. min>,<dens max>), <Opt. Mask Mi>
```

2.2.17 NOT

The NOT operator takes the grayscale complement of image Pj into image Pi such that only those pixels inside of the computing window (and inside of the mask Mi if mentioned) are copied. That is:

```
Pi(r,c) = max density (black) - Pj(r,c).
<Pi> _ NOT <Pj>, <Opt. Mask Mi>
```

2.2.18 EXPAND

EXPAND expands the boundary border pixels of image Pi (which by definition have a 0 value) a maximum of <number pixels> out from the boundary. This is done, for each border pixel, by replacing the 0 valued border pixel with the average of the those other non-zero boundary pixels. This expansion is performed only on that part of the image which is inside the computing window (and mask Mi if specified).

```
<Pi> _ EXPAND <number pixels>, <Opt. Mask Mi>
```

2.2.19 SHRINK

SHRINK contracts border pixels of Pi inside the computing window (and mask Mi if specified) and store the result in Pi. The algorithm is taken from Rosenfeld's 8-neighbor thinning algorithm given in [Ros75].

<Pi> _ SHRINK <number pixels>, <Opt. Mask Mi>

2.2.20 SHIFT

It is possible to shift pixels Pj by a specified vector into Pi if a resultant pixel is inside the computing window (and mask Mi if specified). Pixels outside of the mask/window area are ignored. That is:

If (r-delta y,c-delta x) in computing window ^ mask Mi
Then
Pi(r,c) = Pj(r-delta y, c-delta x)

<Pi> _ SHIFT <Pj>, <delta x>, <delta y>, <Opt. Mask Mi>

2.2.21 SEGMENT

Find all of the connected components of image Pj such that the background pixels of input image Pj have previously been set to 0. The labeled components are stored in output image Pi as pixels whose value is the component number (ranging from 1 to 253) with new associated boundaries having sequential names Bq (q > 32).

When a boundary is created an association is also created between the resultant connected component's image and each boundary associated with a connected component (Pi*Bq=seglist). This association has a property list consisting of (component number, first raster row, first raster column, area, number of boundary pixels, density, boundary name, touching computing window predicate, component image name). This property list may be printed using the ACTIVEDATA command.

If the NOBOUNDARIES switch is used, then the boundaries (Bq) generated during the segmentation are not saved.

If the NOFILLHOLES switch is used, then do not fill in the holes inside of connected components. The default is to fill in such holes as the connected components will often be used to generate masks (MSEGMENT command).

If sizing values (size!lower:size:upper) are specified then only those boundaries whose number of boundary pixels is within these limits are acquired. The algorithm is an adaptation of the boundary follower given in Rosenfeld 'Picture Processing by Computer', Academic Press, 1969, chapter 8.

<Pi> _ SEGMENT <Pj>,<Opt. size!lower, size!upper>,
<Opt. NOBOUNDARIES>,<Opt. NOFILLHOLES>,
<Opt. mask Mi>

2.2.22 WHITENOISE - GENERATOR

The WHITENOISE cperator generates an image Pi inside the computing window (and mask Mi if specified) such that the gray scale values are normally distributed about the specified mean with specified standard deviation.

<Pi> _ WHITENOISE <std dev>, <mean density>,
<Opt. Mask Mi>

2.2.23 ZERO

The ZERO command sets all pixels of Pi inside the current computing window (and mask Mi if specified) to zero.

<Pi> _ ZERC <Opt. Mask Mi>

2.2.24 DELSQPIX - SCALAR

The DELSQPIX command computes the scaler value of the sum of the squares of the pixel gray value difference i.e. |Pj(r,c)-Pk(r,c)|**2 for all (r,c) in the computing window (and under mask Mi if specified).

DELSQPIX <Pj>, <Pk>. <Opt. Mask Mi>

2.2.25 FINDWINDOW - SCALAR

The FINDWINDOW command searches a picture Pi from the current computing window for the minimum size rectangle window within the computing window which contains border pixels above the specified density threshold. It then asks if you would like to (a) pass this new window to the RECTANGLE mask generator parameters or to (b) pass the new window to the current computing window. The density threshold may also be specified using the SETDENSITY command.

FINDWINDOW <Pi>, (USETHReshold switch or <threshold>,
<Opt. Mask Mi>

2.2.26 HISTOGRAMPIX

The histogram of Pj inside of the computing window and mask (if specified) is displayed on the currently selected display terminal. If the Omni mode is being used, it will generate an omni picture with the name HIST!Pi. If the option 'avg # bins' value x is specified, then the histogram will collapse every x bins of the histogram and thus give a coarser histogram. If the R or C option is specified, Then the histogram is the Row or Column gray scale profile of Pj. Such a profile is computed by summing the gray values along each row (R) or column (C).

If the Tektronix 4012 display is used with the crosshairs option enabled, the crosshairs may be used to read values from the display.

<Opt. Ti>_HISTOGRAMPIX <Pj>, <Opt. avg# bins>,
<Opt R or C>, <Opt. Mask Mi>

2.2.27 SHOW - I/O

Display the gray scale image Pj inside of the computing window and mask (if specified) on the currently selected display terminal. If the NUMBER option is used, then print the decimal gray values of the sampled image (with a maximum width of 18 pixels) on the teletype and do not do the gray scale display. If the Tektronix 4012 terminal was specified with the crosshairs option selected, the cross hairs on the terminal may be read repeatedly by PROC10 at the end of the SHOW. The cross hairs option must be set (via SETTERMINAL). The cross hairs will be read and the (row,column) reported to the user at the typing of any character except Q which will return control back to the PROC10 command input loop.

The image or boundary is displayed in a window whose upper left hand corner (LCS (Xp,Yp)) is specified by the SETLCS. The window is specified by the computing window which may be set via either SETWINDOW or FINDWINDOW. SETSAMPLING sets the sampling rate to be used inside of the computing window. If the sampling rate is > 0 then corresponding sub regions between sample points are averaged and the average called the sample and displayed. If the sampling rate is < 0 then no averaging is done.

SETDENSITY specifies the minimum and maximum gray values to be used to define the dynamic range for the display. The scaling factor is defined with the SETSCALING command for use with linear scaling (if 0), or with gamma correction. This is described in more detail in the previous section on Density Values.

SHOW <Pj>, (Opt. NUMBER option), <Opt. Mask Mi>

2.2.28 READ - I/O

The READ command is used to read a picture from the disk into PROC10. If the size of the image is different from the current size, it gives you the option of reading in the image anyway or cancelling the READ. If you desire to read the image anyway, then the old Pi must be deleted so that a new Pi of the correct size may be created. A question is asked: DELETE Pi? to which you must respond YES.

If the NUMBER switch is used, then the input file is assumed to be a sequential list of decimal numbers (in Ascii format) corresponding to the sequential list of numbers in a top down, left to right raster scan. This option facilitates getting non-DDTG formats into PROC10.

<Pi> _ READ <Opt DEV:><Pix file><Opt. [Proj,Prog]>,
<Opt. NUMBER for Ascii data files>

2.2.29 WRITE - I/O

The WRITE command is used to write a picture from PROC10 to the disk file specified. If no title is associated with the image, then request a title before writing the file.

<Opt DEV:><Pix file><Opt. [Proj,Prog]> _ WRITE <Pj>,
<Opt. Mask Mi>

2.2.30 DELETE

The DELETE command deletes a picture and its associated property list (but not the boundaries referenced in its property list which must be deleted with a DELETE <Bi>) from PROC10. This enables PROC10 storage to be recovered (which at 16K words for a full 256x256 pixel square images is considerable).

<Pi> _ DELETE

2.2.31 AREA - SCALAR

The AREA command computes the total area of Pj, in pixels, of those pixels > the specified threshold and inside the computing window and mask Mi if specified.

AREA <Pj>, (USETHReshold switch or <threshold>,
<Opt. Mask Mi>

2.2.32 DENSITY - SCALAR

The DENSITY command computes the sum of the pixel gray values of Pj for those pixels > the threshold and inside the computing window and mask Mi if specified.

DENSITY <Pj>, (USETHReshold switch or <threshold>,
<Opt. Mask Mi>

2.2.33 PERIMETER - SCALAR

The PERIMETER command computes the total perimeter and number of boundary pixels of image Pj for those pixels constituting objects > the threshold and inside of the computing window and mask Mi if specified. Perimeter is defined to be the total number of pixels which are boundary transition points. If a boundary transition is a diagonal then that point is counted as sqrt(2) rather than 1.

PERIMETER <Pj>, (USETHReshold switch or <threshold>,
<Opt. Mask Mi>

2.2.34 MOMENTS - SCALAR

The MOMENTS command computes and prints all (x,y) 0'th to 3'rd order moments of Pk inside the computing window (and under mask Mi if specified) such that for gray value g(x,y) at pixel Pk(x,y):

Mij = SUM(over x,y) g(y,x)*(x^i)*(y^j).
MOMENTS <Pk>, <Opt. Mask Mi>

2.2.35 LAPLACE8

The LAPLACE8 command computes the eight-neighbor Laplacian for pixels of Pj(r,c) and stores it into Pi(r,c) for (r,c) in the computing window (and in the mask Mi if mentioned). That is:

I8 = |I8 - (I0+I1+I2+I3+I4+I5+I6+I7)/8|.
<Pi> _ LAPLACE8 <Pj>, <Opt. Mask Mi>

2.2.36 INSERT

INSERT inserts a smaller image Pj into a larger one Pi at a position specified by the computing window (and under mask Mi if specified). This allows working on parts of a larger image and then reconstructing the larger image with the INSERT command when the operaticns on the smaller parts are finished.

INSERT may also be used to construct images from other images. The size of the smaller image Pj is fixed because it already exists. The size of the larger image is the current image size and may be defined by setting the image size with SETSIZE. Note that the mask Mi must be the same size as the larger image.

<(larger) Pi> _ INSERT <(smaller) Pj>, <Opt. Mask Mi>

2.2.37 EXTRACT

EXTRACT extracts a smaller image Pi from a part of a larger one Pj such that the pixels of the extracted image are inside of the computing window (and under mask Mi if specified). Thus the computing window specifies the size of the new smaller image Pi (if Pi does not exist previous to ding an EXTRACT). This allows working on parts of a larger image and then reconstructing the larger image when the operations on the smaller parts of finished.

The size of the larger image Pj is fixed because it already exists. The size of the smaller image Pi is the current image size and may be defined by setting the image size with SETSIZE and creating a zeroed image with (Pi_ZERO), or by setting the size of the computing window. Note that the mask Mi must be the same size as the larger image.

<(smaller) Pi> _ EXTRACT <(larger) Pj>, <Opt. Mask Mi>

2.2.38 EXTREMA - SCALAR

The EXTREMA command computes the maxima and minima modes of the gray scale histogram of Pj inside of the computing window (and under the mask Mi if specified). The switches R and C may be used to specify a Row or Column histogram. The i'th histogram entry for R (C) is the sum of the row (column) gray values in Pj. The i'th minimum may be requested to be saved as the new density minimum (for use in automatic slicing).

EXTREMA <Pj>, <opt. i: Min[i]==>threshold or USEMEAN>,
<optional R or C>, <optional Mask Mi>

2.2.39 LINCOMB

The linear combination of two images may be computed such that Pi(r,c)=(0 Max (Aj*Pj(r,c)+Bk*Pk(r,c) Min maximum computing density)) inside of the computing window (and under mask Mi if specified). Note that all images must be the same size. If a constant is not defined, then it defaults to 0. Thus if Aj is non-zero and Bk is zero the LINCOMB operation acts like the SCALE operation.

<Pi> _ <Pj> LINCOMB <Pk>, <Real Aj>, <Real Bk>,
<Opt. Mask Mi>

2.2.40 LISTSEGMENTS - STATE

LISTSEGMENTS lists the ACTIVEDATA status of a particular image Pj. This command is the same as doing an ACTIVEDATA Pj.

LISTSEGMENTS <Pj>

2.2.41 ZOOM

The ZOOM command is used to zoom up or down (increase or decrease) the magnification of input image Pj and store the resultant image in Pi. The source image is that available under the computing window and optional mask Mi. The size of the output image must be the same as that of the input image. If the magnification is > 1 then the zoom is performed by repeating pixels in a square tesselation. If the magnification is < 1 then the zoom is performed by averaging areas between sample points.

<Pi> _ ZOOM <Pj>, <Zoom magnification 1/256:256>,
<opt. Mask Mi>

2.2.42 MAKEPIX

The MAKPIX command is used to create an image from a boundary by setting Pi(r,c) to the maximum gray value for all (r,c) in a boundary Bi such that (r,c) is inside the computing window (and under the mask Mi if specified). If a gray value is specified, then fill the inside of the boundary with this value othersize don't fill it and use the maximum gray value for the Pi(r,c). This allows one to write out boundaries in an image.

<Pi> _ MAKEPIX <Boundary Bi>, <Opt. gray value to fill with>, <Opt. Mask Mi>

2.2.43 PRINT - I/O

The PRINT command prints picture Pi on the line printer (logical device LPT: which may be assigned as the disk by doing an .ASSIGN DSK: LPT: at PDP10 monitor level, otherwise it print immediately). The output file has the print name PiXnnn.LPT. If '.ASSIGN TTY: LPT:' is requested at PDP10 monitor level, then the printout will go to the teletype instead of the LPT:. The number nnn is incremented (starting from 000) each time the PRINT command is used.

The Decimal mode prints out a maximum of 64 32x32 subwindows of Pi where the 3 digit gray value for each pixel is printed. Each page has a heading consisting of the picture file name, picture title and (first/last rows, first/last columns) of the section of the image printed on the current page. Only those pages falling within the computing window will be printed. This is useful in cutting down the amount of printout. If the TTY option is specified, the output will go to the teletype instead of the logical lineprinter.

The single hexidecimal mode prints the hexidecimal value of the top 4 bits of an 8-bit gray value for each pixel as (0-9,A-F). The default mode prints the top 3-bits of the 8-bit gray scale pixel value as a pseudo gray scale using the character set: (<space> . , : ! / & #).

PRINT <Pi>, <Opt. 'D'ecimal, 'S'ingle hexidecimal,
(default is 8 character grayscale)>,
<Opt. TTY for printing on the teletype>

2.2.44 TEXTURE1 - SCALAR

The TEXTURE1 command computes the texture measure 1 of all Pi inside of the computing window (and mask Mi if specified). Texture measure 1 is the histogram of the horizontal run length sizes for pixels g(r,c) such that g > the specified threshold.

TEXTURE1 <Picture Pi>, (USETHRhreshold switch or
<gray scale threshold>, <Opt. Mask Mi>

2.2.45 TEXTURE2 - SCALAR

The TEXTURE2 command computes the texture measure 2 of all Pi inside of the computing window (and mask Mi if specified). For a given texture sample, a symmetric matrix is constructed such that each element b(u,v) of this matrix indicates the number of times an element of the sample with gray value u has a right-hand neighbor with the gray value v [Ros70]. The coarser the texture, the greater the tendency for a point in the texture sample to be followed by a point with a like or similar gray value. Thus the greater the coarseness, the greater will be the tendency of the [b(u,v)] matrix to have its high values concentrated near the main diagonal.

TEXTURE2 <Picture Pi>, <Opt. Mask Mi>

2.2.46 TEXTURE3 - SCALAR

The TEXTURE3 command computes the texture measure 3 of all Pi inside of the computing window (and mask Mi if specified).

TO BE ADDED

TEXTURE3 <Picture Pi>, (USETHRhreshold switch or <gray scale threshold>, <Opt. Mask Mi>

2.2.47 FILTER

The FILTER command computes the neighborhood product between Pj and the specified real valued 9 element neighborhood at each pixel of Pj in the computing window and under the mask if specified. The neighborhood elements I0:I8 are defined according to the ordering imposed by our definition of neighborhood (see Neighborhood definition). Each neighborhood pixel Pi(r,c) is computed as:

$$Pi(r,c) = \sum_{k=1}^{9} Pjk * dlist(k).$$

<Pi> _ FILTER <Pj>,<Opt. Mask Mi>, <9 Real values I0:I8>

2.3 Mask Operators

The following operators have mask data structures as the domain of their major operands. The output mask (if required) may be a previously created mask. Alternatively, it may be a newly created mask with default gray scale values of zero.

Operations are performed inside of the computing window ANDed with a mask generator (such as MCIRCLE) if specified. Output mask pixels not operated on are not changed. Thus, by using different computing windows and/or mask generators, different parts of an output mask may be created by different mask operators.

2.3.1 AND

AND computes the conjunction of two masks Mj and Mk such that Mi(r,c)=1 if both Mj(r,c)=1 and Mk(r,c)=1.

<Mi> _ <Mj> AND <Mk>

2.3.2 OR

OR computes the disjunction of two masks Mj and Mk such that Mi(r,c)=1 if either Mj(r,c)=1 or Mk(r,c)=1.

<Mi> _ <Mj> OR <Mk>

2.3.3 MINUS

MINUS computes the logical difference of masks Mj and Mk and stores it in Mi.

<Mi> _ <Mj> MINUS <Mk>

2.3.4 COPY

COPY copies mask Mj into mask Mi.

<Mi> _ COPY <Mj>

2.3.5 NOT

NOT computes complement of mask Mj such that Mi(r,c) = If Mj(r,c)=1 Then 0 Else 1 and stores the result in Mi.

<Mi> _ NOT <Mj>

2.3.6 ZERO

ZERO sets all mask pixels to zero.
<Mi> _ ZERO

2.3.7 READ - I/O

The READ command reads a mask from the disk into PROC10. If the size of the mask is different from the current image size, it gives you the option of reading in the mask anyway or cancelling the READ. If you desire to read the mask anyway, then the old Mi must be deleted so that a new Mi of the correct size may be created. A question is asked: DELETE Mi? to which you must respond YES.

If the NUMBER switch is used, then the input file is assumed to be a sequential list of decimal numbers (in Ascii format) corresponding to the sequential list of numbers in a top down, left to right raster scan. This option facilitates getting non-DDTG formats into PROC10.

<Mi> _ READ <Opt DEV:><Mask file><Opt. [Proj,Prog]>,
<Opt. NUMBER for Ascii data files>

2.3.8 WRITE - I/O

The WRITE command writes a mask from PROC10 to the disk file specified. If no title is associated with the mask, then request a title before writing the file.

<Opt DEV:><Mask file><Opt. [Proj,Prog]> _ WRITE <Mj>

2.3.9 DELETE

The DELETE command deletes the mask Mi and returns the freed storage to PROC10.

<Mi> _ DELETE

2.3.10 MCIRCLE - GENERATOR

The MCIRCLE command generates a circular mask (anded with the computing window) of the specified radius and centered at the row and column specified.

<Mi> _ MCIRCLE, <radius>, <row center>, <column center>

2.3.11 RECTANGLE - GENERATOR

The RECTANGLE command generates a rectangular mask (anded with the computing windcw) of the specified size and centered at the row and column specified.

<Mi> _ RECTANGLE <row side>, <col side>,
<row center>, <col center>

2.3.12 SPHERE - GENERATOR

The SPHERE command generates a circular mask (anded with the computing window) which is the 2D slice of a sphere at the z center with the specified radius and centered above or below (+/- z) the row and column specified.

<Mi> _ SPHERE <z center>, <radius>,
<row center>, <column center>

2.3.13 SQUARE - GENERATOR

The SQUARE command generates a square mask (anded with the computing window) of the specified size and centered at the row and column specified.

<Mi> _ SQUARE <size>, <row center>, <col center>

2.3.14 WHOLE - GENERATOR

The WHOLE command generates a mask of all ones of the current image size inside of the computing window.

<Mi> _ WHOLE

2.3.15 MSLICE - GENERATOR

Generate a threshold mask by setting mask pixels Mi(r,c)=1 if image pixels Pj(r,c) are within the threshold slice range [dmin < Pj(r,c) leq dmax] otherwise set Mi(r,c)=0. If the USETHReshold switch is used, then the current threshold set with SETDENSITY is used as the minimum density and the maximum computing density is used as the upper bound.

<Mi> _ MSLICE <Picture Pi>, (USETHReshold switch or
<min dens>, <max dens>)

Various mask operations which are not explicitly available in the mask domain but are in the picture domain (such as ROTATE, SHIFT, EXPAND, SHRINK, etc.) may be performed by converting a mask to a black and white picture, performing the operation on the image and then using MSLICE to convert the picture back to a mask.

[1] Create a picture from the mask,

```
Pi_ZERO
Pi _ NOT Pi,Mi
```

[2] Perform the picture operation (abreviated *),

```
Pj _ * Pi
```

[3] Convert the image back to a new mask,

```
Mj _ MSLICE Pj,0,255
```

2.3.16 MSEGMENT - GENERATOR

Generate a connected component mask by setting mask pixels Mi(r,c)=1 if image pixels Pj(r,c) are within the connected component specified by the segmented image Pj (output of the SEGMENT command) and segment number referring to the connected component. If the pixel is not in the connected component, then set Mi(r,c)=0.

<Mi> _ MSEGMENT <Segmented Pix Pi>, <segment number>

2.3.17 AREA - SCALAR

AREA computes the mask area in pixels by counting 1's.

AREA <Mi>

2.3.18 PERIMETER - SCALAR

PERIMETER computes the total perimeter of mask Mj. Perimeter is defined to be the total number of pixels which are boundary transition points (i.e. going from a 0's background into a 1's solid object). If a boundary transition is a diagonal arc than that point is counted as sqrt(2) rather than 1.

PERIMETER <Mi>

2.4 Boundary Operators

The following operators have boundary data structures as the domain of their major operands. The output boundary or boundary transform (if required) may be a previously created boundary or boundary transform. Alternatively, it may be a newly created boundary or boundary transform which assumes the size computed by the operation. Boundaries which were never defined and are used as input boundaries have zero length. domain.

2.4.1 COPY

COPY copies boundary Bj into boundary Bi.
<Bi> _ COPY <Bj>

2.4.2 ZERO

ZERO Zeros boundary Bi.
<Bi> _ ZERO

2.4.3 READ - I/O

The READ command reads a boundary file into Bi or transform file into Ti from the PDP10 disk. Check to make sure it is a valid data structure file. Print the title after reading the file.

If the NUMBER switch is used, sequential decimal Ascii (x,y) pairs of numbers are read in until either a (0,0) is seen or the end of file mark is seen.

<Bi or Ti>_READ <Opt DEV:><file name><Opt. [Proj,Prog]>,
<Opt. NUMBER for Ascii data files>

2.4.4 WRITE - I/O

The WRITE command writes a boundary or transform file from PROC10 to the PDP10 file system. If no title is associated with the data structure, request a title before writing the file.

<Opt DEV:><file name><Opt. [Proj,Prog]>_WRITE <Bj or Tj>

2.4.5 DELETE

The DELETE command deletes the boundary Bi or transform Ti and return storage to PROC10.

<Bi or Ti> _ DELETE

2.4.6 SHOW - I/O

Display boundary Bj inside of the computing window on the currently selected display terminal with the computing window positioned at the logical coordinate LCS(Xp,Yp). If the DEC GT40 or Tektronix 4012 terminals are used, the SETBOUNDARYSCALEFACTOR command may be used to shrink or expand the drawing by a factor of 0.1 to 10 (X1 is the default). A special line drawing mode is available on the Tektronix 4023 display which is considerably faster than using ASR33 line drawing simulation mode (by filling in images).

SHOW <Bj>,<Opticnal FASTVECTOR switch for 4012 display>

2.4.7 AREA

The AREA within a boundary is computed using an algorithm which approximates the area with subtended polygons. The area algorithm is taken from [Sham75].

AREA <Bj>

2.4.8 PERIMETER - SCALAR

PERIMETER computes the perimeter and length of a boundary. The length is just the number of points which comprise the boundary. The perimeter points which are diagonal are counted sqrt(2) rather than 1.

PERIMETER <Bj>

2.4.9 CIRCLETRANSFORM

The Circle Transform of a boundary Bi ([Shap76a], [Shap76b]) is computed given the sampling distance along the boundary. The output of the transform consists of a list of triples. The first component is a signed radius of curvature for that segment. A positive sign indicates concavity while a negative sign indicates convexity. The second component indicates the angular deflection (in radians) between two adjoining segments. The third component indicates the arc length of the segment. If a Tektronix 4012 or DEC GT40 display is available, then DISplaytheanalysis switch will cause the sample points to be delimited with small circles and the corresonding fitted circles (if the OSCulatingcircledisplay switch is mentioned) to be displayed. The process may be stoped after each fit for viewing by using the WAITaftereachfit switch.

<Ti> _ CIRCLETRANSFORM <Bi>,<Sampling distance>,
<Opt DISplaytheanalysis>,
<Opt. WAItaftereachfit if DISplayanalysis>,
<Opt. OSCulatingcircledisplay if DISplayanalysis>

2.4.10 ICIRCLETRANSFORM

ICIRCLETRANSFORM computes the inverse circle transform starting at the specified angle from a list of triples produced from a CIRCLETRANSFORM operation.

<Bi> _ ICIRCLETRANSFORM <Tj>, <Starting angle>

2.4.11 SUBARCS

Copy the specified sublist of ordered arcs from Tj into a new transform Ti.

<Ti> _ SUBARCS <Tj>, <From arc p>, <To arc q>

2.4.12 LISTTRANSFORM - I/O

LISTTRANSFORM lists the type and parameters of the specified transform as well as its specific values.

LISTTRANSFORM <Tj>

2.4.13 FOURIERTRANSFORM

FOURIERTRANSFORM takes the complex Fourier transform of Bj over the spatial frequency range specified.

<Ti> _ FOURIERTRANSFORM <Bj>, <lower omega>,<upper omega>

2.4.14 IFOURIERTRANSFORM

IFOURIERTRANSFORM takes the complex inverse Fourier transform of Tj over the spatial frequency range specified and constructs a new boundary Bi.

<Bi> _ IFOURIERTRANSFORM <Tj>, <lower omega>,<upper omega>

2.4.15 WALSHTRANSFORM

WALSHTRANSFORM takes the centroid WALSH transform of Bj over the spatial frequency range specified by the number of samples.

<Ti> _ WALSHTRANSFORM <Bj>, <number samples>

2.4.16 IWALSHTRANSFORM

IWALSHTRANSFORM takes the centroid inverse WALSH transform of Tj over the spatial frequency range specified by the number of samples and constructs a new Bi.

<Bi> _ IWALSHTRANSFORM <Tj>, <number samples>

2.4.17 CENTFOURIERTRANSFORM

CENTFOURIERTRANSFORM takes the centroid FOURIER transform of Bj over the spatial frequency range specified by the number of coefficients.

<Ti> _ CENTFOURIERTRANSFORM <Bj>, <number coefficients>

2.4.18 ICENTFOURIERTRANSFORM

ICENTFOURIERTRANSFORM takes the centroid inverse FOURIER transform of Tj over the spatial frequency range specified by the number of coefficients and constructs a new Bi.

<Bi> _ ICENTFOURIERTRANSFORM <Tj>

2.4.19 LISTBOUNDARY - I/O

LISTBOUNDARY lists the first and last boundary (x,y) points of Bi to indicate closure. It then lists all of the boundary points.

LISTBOUNDARY <Bi>

SECTION 3

References

Carm74. Carman G, Lemkin P, Lipkin L, Shapiro B, Schultz M, Kaiser P:A real time picture processor for use in biological cell identification - II hardware implementation. J. Hist. Cyto. Vol 22, 1974, 732:740.

CCB76. Computer Center Branch:DECsystem-10 Omnigraph Display Manual. DCRT, NIH, Bethesda, Md. 20014., April 1976.

Gor75. Gordon R, Silver L, Rigel D S:Halftone graphics on computer terminals from storage display tubes. Proc. Soc. Information Display. In press.

Knott73. Knott G, Reese D:MLAB - an On-line Modelling Laboratory. NIH, DCRT, Bethesda, Md., Dec., 1973.

Lem74. Lemkin P, Carman G, Lipkin L, Shapiro B, Schultz M, Kaiser P:A real time picture processor for use in biological cell identification - I systems design. J. Hist. Cyto. Vol 22, 1974, 725:731.

Lem75. Lemkin P, Shapiro B.:PROCES - An Image Processing Program for the PDP8e. NCI/IP-75/02 Technical Report #1, NTIS PB244264/AS. July 1975.

Lem76a. Lemkin P, Carman G, Lipkin L, Shapiro B, Schultz M:The real time picture processor - description and specification. NCI/IP-76/03 Technical Report #7, NTIS PB25268/AS, March 1976.

Lem76b. Lemkin P:DDTG - Functional Specification for the RTPP Monitor/Debugger. NCI/IP-76/02, Technical Report #2, NTIS PB250726, Feb 1976.

Sham75. Shamos M:Geometric Complexity. 7'th Annual ACM symposium on the Theory of Computing, 1975.

Shap76a. Shapiro B, Lipkin L:The circle transform: an articulatable shape descriptor. In prep.

Shap76b. Shapiro B, Lipkin L:The use of Orthogonal Expansions for Biological Shape Description. Univ. Md. TR-, Aug 1976.

Shap76c. Shapiro B, Lemkin P:A 9 track magtape intermediary between the PDP8e and PDP10 computers. NCI/IP Technical Report #20, Sept. 1976.

Ros69. Rosenfeld A:Picture Processing by Computer. Academic Press, 1969, Chap 8.

Ros70. Rosenfeld A, Troy B:Visual Texture Analysis. Univ.

Md. Computer Science Center TR-70-116, June, 1970.

Ros75. Rosenfeld A, Davis L S:A Note on Thinning. Univ. Md. Computer Science Center TR381, May 1975.

VanL73. VanLehn, K:Sail User Manual. Stanford Artificial Intelligence Laboratory memo AIM-204, July 1973.

INDEX