INTERNATIONAL ATOMIC ENERGY AGENCY
UNITED NATIONS EDUCATIONAL, SCIENTIFIC AND CULTURAL ORGANIZATION
INTERNATIONAL CENTRE FOR THEORETICAL PHYSICS
I.C.T.P., P.O. BOX 586, 34100 TRIESTE, ITALY, CABLE: CENTRATOM TRIESTE

H4-SMR 471/08

COLLEGE ON MEDICAL PHYSICS

10 - 28 SEPTEMBER 1990

IMAGE PROCESSING HARDWARE AND PRACTICES USING A PIP (PROGRAMMABLE IMAGE PROCESSING) SYSTEM

P. E. Cruvinel

EMBRAPA/NPDIA
P. O. Box 741
13560 Sao Carlos
Brazil

INTERNATIONAL CENTRE FOR THEORETICAL PHYSICS

TRIESTE (ITALY)

COLLEGE ON MEDICAL PHYSICS
Sept. 1990

IMAGE PROCESSING HARDWARE AND PRACTICES USING
A PIP (PROGRAMMABLE IMAGE PROCESSING) SYSTEM

P.E.Cruvinel

EMBRAPA/NPDIA
P.O.Box 741
13560-Sao Carlos-SP
Brazil

IMAGE PROCESSING HARDWARE AND PRACTICES USING
A PIP (PROGRAMMABLE IMAGE PROCESSING) SYSTEM

Paulo Estevao Cruvinel
EMBRAPA/NPDIA
P.O.Box 741
13560-Sao Carlos-Brazil

A minimal image processing system consists of an image acquisition device,an image memory,a computer that can access this memory and a monitor that can display the contents of memory.This basic system can acquire,processes and displays momochrome images (if the monitor is RGB it is possible display image memory values in color by adding output lookup tables).

The image acquisition device puts an image into the image memory.This operation involves digitizing scanning a continuous image and breaking it into an array of digital intensity values called pixels.In general the signal from the video camera is converter by an A/D converter into a pixel array in the image memory.

The acquisition device can write to the image memory,which can be read and written to by the computer's CPU,and read by the display device.

If the image memory stores an entire video image,it is called a frame buffer or frame store.For acceptable intensity and detail resolution,an average monochrome image must be represented by an array of at least 256 by 256 pixels and each pixel must have at least 6 bits of value.Gray level intensities in the pixel is a function of the N-bit byte.

The display device converts the processed pixels back into spatially organized image intensities.The display device is usually a D/A converter that drives a monochrome or RGB TV monitor.

A lookup table (LUT) changes a pixel's value based on the values in a table.This hardware consists of a memory that has a storage location for each possible pixel value.An input pixel value,in this case,is used as an address into this memory,and the output is the value at that address.An LUT computes an arbitrary

function of one or more variables,with a domain and range limited to the possible pixel values.

An image processing system can have three LUTs that map the image memory to the display device.These LUTs output values to the red,green,and blue channels of a color monitor,based on some input pixel value.This lets the display with different gray levels from the monochrome image in various arbitrary colors or pseudocolor.

The PIP.640-B is a plug in video frame grabber-digitizer board for the IBM PC.It has a resolution of 640X512 pixels in normal (non zoom) mode with eight bits per pixel and a power consumption of approximately 17W.The PIP.640-B is capable of operating in continuous or single frame grab mode and also has built-in video keying capabilites.Frames which have been stored in the on board frame buffer can be loaded into the PC's memory or onto disk.

Conversely,video data,as well as lookup table data,can be written to the PIP from the PC.Pixels can be individually addressed by the PC by using the X and Y address registers or they can be addressed sequentially using an auto-increment register.It has one input and three output lookup tables,each of which has eight maps to choose.There are three input ports and an IBM pin compatible RGB output as well as an internal feedback for diagnostic purposes.

Figure 1 shows the operation of the PIP.640-B in block diagram.

Input signals are selected,in software,from one of three input ports and the PIP.640-B will lock on to the input signal's sync as well as use the sync signal to drive the video functions of the board.

Data,after it passes the sync separator,is subject to an adjustable DC offset voltage and this offset adjustment,with the gain,makes the picture brigter or darker.

The video input is digitized in real time in the A/D converter,producing an eight bit number that is sent to the input lookup table.

Figure 2 shows the input section in block diagram.

The PIP.640-B is assembled with 1Mbyte of frame buffer RAM which is used to store frame grab data.Both the system unit and the CRT-controller have simultaneous transparent access to the frame buffer.

The CRT-controller sends pixel data to the output LUT's.It uses the SY6845E integrated circuit.

This gives the ability to shift the image both vertically horizontally which is useful in the PIP.640-B with its ·X1024 storage area and 640X512 display area.

The frame buffer is accessible (read and write) from the :em unit using X,Y coordinates.The frame buffer address sters can be set also to automatically increment.

Figure 3 shows the frame buffer data access in block ıram.

There are two sources of data for output from the PIP,that he frame buffer and the input LUT.It is possible to select ıer video keying,or simply the output from either the frame 'er or the input LUT.

Figure 4 shows the keying and output section in block ıram.

Image processing includes :

enhancement

restoration

measurement of image elements

classification of image elements

recognition

The knowledge level used in an algorithm can range from iminary conception about the physics of image formation to a ific knowledge about sections in a picture-frame and the uate diagnostic at medical field or about pixel structures.

In these practices a set of tools will be used for image essing as well as algorithms using the following procedures :

a-POINT PROCESSING

b-AREA PROCESSING

c-GEOMETRIC PROCESSING

d-FRAME PROCESSING

In all cases images-based algorithms transform pixel values other pixel values or locations using numerical or logical ations.Pseudocolor will be used.

The image that will be used in these practices (from 2 to was obtained from the dual energy CT technique.The object ath the cross section is the bone mineral phantom with five bration materials:

fat equivalent material,water,50mg/cc,100mg/cc and 200mg/cc K HPO ,respectively and a high energy (140 kVp,100mA,10mm thickness) scan through the abdominal region.

A. POINT PROCESSING

A point process algorithm scans through the image area and uses the pixel value at each point to compute a new value for the point.

This method can be used to enhance or modify pixel values.

If the pixel value and its location are used then it will be useful to correct artifacts or smoothly change pixel values (add or subtraction values) in an image area.

Another possibility with point process algorithm is the use of the histogram since it is an example of image measurement.

An intensity histogram performed on a digitized image results in a graphic interpretation of the number of occurrence of a given pixel intensity value throughout the image.It is possible to create a histogram of a grey scale levels in a selected current window.This facility involves two steps:

-COMPUTING THE ARRAY OF VALUES FOR THE HISTOGRAM

-DRAWING THE HISTOGRAM ONTO THE IMAGE BEING DISPLAYED

However,using point process algorithm,there is only the possibility that it examines a single pixel at a time without changing the pixel's values.

The information provided by the histogram can be used for image enhancement and classification.In other words the histogram represents the relationship between the relative intensity (in general with range from 0 to 255) of the pixel's values and the number of pixels of each relative intensity.

Pseudocoloring of a monochrome image is a point process algorithm.In this case it is necessary to map the values of Xn(argument-the pixel value is the input argument) onto Yn(function drives-three different functions,the red,the green and, the blue) as in a function with the abcissa (X) as the index to the lookup table and the ordinate (Y) as the mapped value.In this way the information stored in the frame buffer is manipulated by changing the values stored in a lookup table (LUT).

These output functions drive the red(R),the green(G) and,the blue(B) guns of a color monitor.

B. AREA PROCESSING

The area process algorithm uses neighbourhood information to modify pixel values.

These kind of algorithms are typically used for spatial filtering and changing an image's structure.It is possible,using area process,to perform :

-AN IMAGE SHARPENING TRANSFORMATION

-A PIXEL AVERAGING TRANSFORMATION

-A HORIZONTAL EDGE DETECTION TRANSFORMATION

-A VERTICAL EDGE DETECTION TRANSFORMATION

-A LAPLACIAN EDGE DETECTION

as well as to perform non-linear area process such as SOBEL filter operation.

Convolution is a classic image processing algorithm used for spatial filtering and it is useful to remove noise,smoothing the image as well as to make edge detections in the image.

The convolution operation replaces a pixel's value with the sun of that pixel's value at its neighbours,each weighted by a factor.The weighting factors are called the convolution kernel.

To convolve an image area with a kernel it is necessary to repeat the operation(convolution) at every pixel position in the image.

At each point or pixel,it is necessary to multiply the kernel values by the image values,sum the results and replace the pixel at the center of the kernel with that value.

The general equation for the convolution operation becomes

$$p(x,y)=\sum_{m,n=0}^{2} k(m,n)*p(x+m,y+n) \qquad (1)$$

where p(x,y) is the image point and k(m,n) is the kernel supposing that the operation is using a 3 by 3 pixel neighbourhood and kernel.Convolving an area of size X by Y with a kernel of size N by M requires X*Y*N*M multiplications and additions.

Another important possibility using area process algorithms is that it performs spatial frequency filtering.

Spatial frequency is the number of times per unit distance

that a pattern repeats.As with a one dimensional signal,an image can be represented by a series of sine and cosine waves with the spatial frequency and amplitude specified for each component.

Therefore,for the image,transformation must be done both horizontally and vertically because it is bi dimensional and thus it has spatial frequencies in both directions.It is possible to detect a certain band of frequencies using a selected kernel.

In general,quickly changing image intensities,are represented by high spatial frequencies,while slowly changing intensities are represented by lower spatial frequencies and if a high spatial frequencies kernel is selected then edges will be detected.On the other hand if a kernel matches lower spatial frequencies it will blur the image.

The Sobel filter as well as the median filter are particular case of area processes algorithms that means,they are non linear area process algorithms which provide a better signal to noise for detecting images elements or features with less computation.

The Sobel filter compares the results of two convolutions to estimate the strength and orientation of edges in the image.For instance,if the two kernels are represented by X and Y then the edge strength and orientation are represented by:

$$STRENGTH=(X*X+Y*Y)^{\frac{1}{2}}$$

$$ORIENTATION=\arctan(Y/X)$$

The median filter replaces the pixel at the center of a neighbourhood of pixels with the median of the pixel value.The neighbourhood values,also the center pixel,are sorted into ascending order and the median value is used to replace the center pixel.In this way it is possible to remove spot noise.

C. GEOMETRIC PROCESSING

The geometric process algorithm changes the spatial arrangement of pixels.It is possible to rotate,stretch,translate the image position,dilate or erode as well as to warp the image.

Geometric process algorithms can be used to improve accuracy as well as to compensate rectangular pixels by adjusting the transformation equations.

D. FRAME PROCESSING

All algorithms that use more than one image are called frame process.For differential CT-scanner analyses this method is useful because it can be used to subtract one image from another.

o it can be used to improve image quality and to detect ion.

It is also useful for static image or a microscopic image lyses ,that means,using N succesive images frames it is sible to reduce noise making an average.

In order to detect motion this algorithm is adequate but the ges processing hardware that can do the subtraction in real e.

TO OPERATE THE VIDEO DIGITIZER BOARD PIP.640-B

To operate the video digitizer board a MS DOS software rary is provided.It is possible to use this software library h microsoft C (vers.4.00) and microsoft FORTRAN (vers.3.31) as l as microsoft PASCAL.In addition to the software library an erpreter program is provided.

The brief overview of the available commands for the video itizer board are shown below:

Initialization Commands

Name	Description
init	Initializes the board to a known state
inifmt	Initializes the board to a specified video format
inisys	Initializes system variables only
format	Selects a video format to use
dfmt	Defines a video format

Hardware Commands

Name	Description
cgrab	Enables or disables continuous frame grabbing
chan	Selects active input channel
clear	Clears the screen to the current draw index
dquad	Selects quadrant on PIP-1024 when in 512 × 512 mode
exit	Resets system unit hardware and exits library
gain	Sets the gain
key	Enable or disable keying
lutm	Selects active map in the LUT
mask	Sets the write protect mask
mode	Tells software about hardware parameters
offset	Sets the offset
outv	Enables or disables video output
pan	Sets the horizontal pan position
panrel	Moves the horizontal pan position by an offset
quadm	Sets the display mode on a PIP-1024
sboard	Selects default board
sbuf	Selects between incoming video and frame buffer
scroll	Sets the vertical pan position
scrorel	Moves the vertical pan position by an offset
setxy	Sets the x and y position for auto-incrementing
snapshot	Takes a snapshot
snapall	Takes a synchronized snapshot using a group of PIPs
statr	Saves the selected board's status
statw	Loads a board's status
sync	Selects the sync source (external or internal)
video	Selects video standard (American or European)
vwait	Waits for vertical blanks
winmode	Sets workspace configuration
zoom	Enables zoom mode

I/O Commands

Name	Description
btransfer	Transfers data between a PIP and the system
colr	Reads a column from the frame buffer
colw	Writes a column to the frame buffer
cpws	copies between workspaces
decode	Decodes a file and write it to the window
encode	Encodes a window and write it to a file
frdisk	Copies a DOS file into the frame buffer
frmem	Copies from system memory to frame buffer
igetb	Reads a byte from a buffer
inp	Reads an I/O port
outp	Writes to an I/O port
pixr	Reads a pixel
pixw	Writes a pixel
putbch	Writes a byte to a buffer
rowr	Reads a row from the frame buffer
roww	Writes a row to the frame buffer
setwindow	Defines the current window
setxy	Sets position in frame buffer
todisk	Copies the frame buffer to a DOS file
tomem	Copies from frame buffer to system memory
winfrdisk	Copies to window from disk
winr	Copies the current window to a buffer
wintodisk	Copies from widow to disk
winw	Copies a buffer to the current window

LUT Commands

Name	Description
dhisto	Draws a histogram in the frame buffer
histo	Calculates a histogram from the current window
initlut	Initializes LUT to grey scale ramp
lutd	Initializes a LUT
luts	Initializes a LUT to a ramp
modhist	Modifies a histogram through mapping
scaling	Defines a mapping function

Imaging Commands

Name	Description
average	Pixel averaging using a 3 × 3 kernel
con3	General convolution around a 3 × 3 kernel
dilate	Dilation routine
erode	Erosion routine
fil3	Convolution around a 3 × 3 kernel
filtype	Select the type of normalization in filtering
horfilter	Horizontal edge detection using a 3 × 3 kernel
ker3	Defines a arbitrary 3 × 3 kernel
lp1 & lp2	Laplacian edge detection using a 3 × 3 kernel
map	Transforms the frame buffer using a software LUT
median	3 x 3 median filter
prewitt	Prewitt edge detection using 3 × 3 kernel
sh1 & sh2	Image sharpening using a 3 × 3 kernel
sobel	Sobel edge detection using 3 × 3 kernel
verfilter	Vertical edge detection using a 3 × 3 kernel

Graphic Commands

Name	Description
border	Draws a border
circ	Draws a circle
circf	Draws a filled circle
clear	Clears and sets the frame buffer to current index
elips	Draws an ellipse
elipf	Draws a filled ellipse
grid	Draws a variable sized grid
linerel	Draws a line to a relative point
lineto	Draws a line to a specified point
moverel	Moves the current point to a relative point
moveto	Moves the current point to a specified point
rect	Draws a rectangle
rectf	Draws a filled rectangle
setind	Specifies the drawing index
tchar	Writes a character
text	Writes a string
tfont	Selects a character font

Interpreter specifc commands

Name	Key	Description
dmemory		Displays 256 bytes of memory
execute	ALT	Executes a macro
getfile		Read data file into memory
getmacro		Reads a file to define a macro
help	F5	Displays detailed help for interpreter
macro	SFT Fn	Creates specified macro
mmemory		Modifies memory contents
pause		Waits until a key is struck
prflag		Enables/disables output to system console
putfile		Writes contents of memory to a file
print		Displays a message
ptime		Prints the system time
quit	F8	Quits the interpreter
record		Sends all commands to an output file
repeat		Executes a macro a number of times
savemacro		Saves a macro to a file
	F1	Gives context sensitive help
	F2	Lists available commands
	F3	Recalls previous command
	F4	Recalls stored line
	F6	Gives detailed command descriptions
	F7	Gives the display work buffer address
	F9	Reads and executes a list file
	F10	Executes DOS comands
	→	Moves cursor right
	←	Moves cursor left
	Home	Moves cursor to beginning of line
	End	Moves cursor to end of line
	Ins	Toggles insert/overstrike mode
	Esc	Cancels current line
	↑	Recalls next stored command
	↓	Recalls previous stored command
	Pg Up	Displays list of stored commands
	CTRL Fn	Displays specified macro

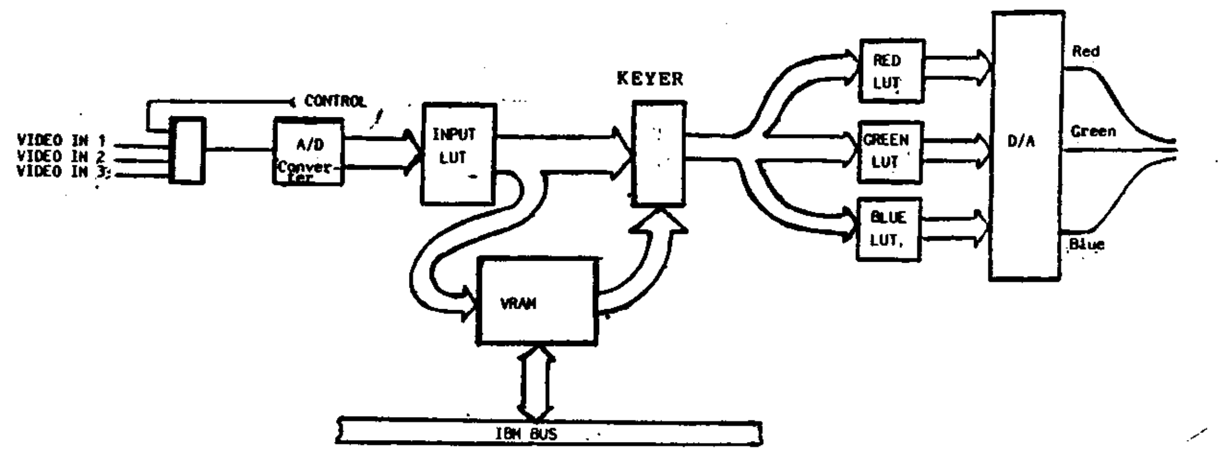

FIGURE 1- THE OPERATION OF PIP.640-B IN BLOCK DIAGRAM

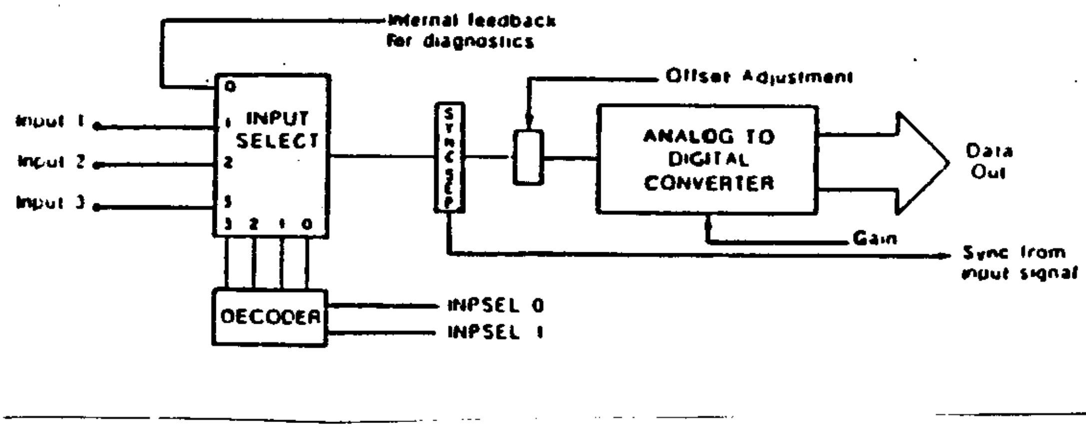

FIGURE 2- INPUT SECTION IN BLOCK DIAGRAM

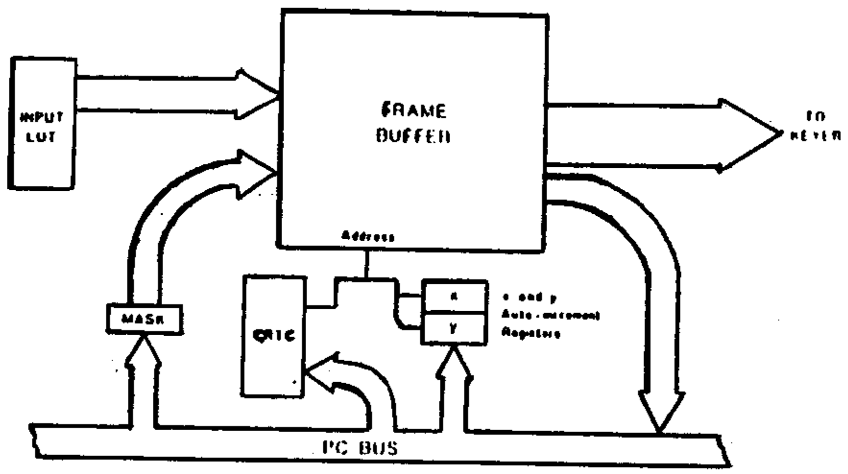

FIGURE 3- FRAME BUFFER DATA ACCESS IN BLOCK DIAGRAM

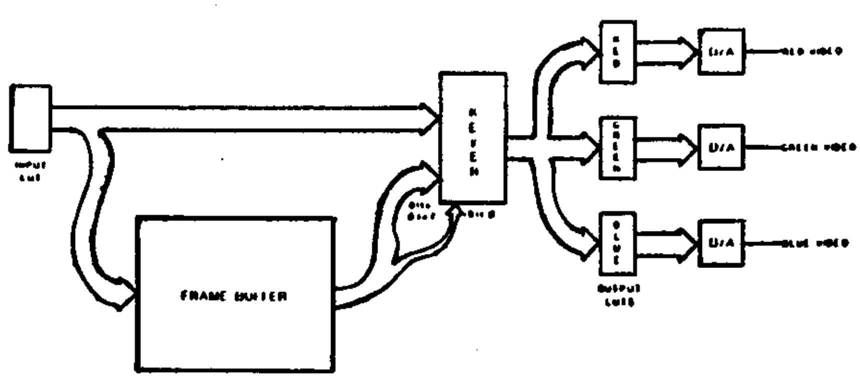

FIGURE 4- KEYING AND OUPUT SECTION IN BLOCK DIAGRAM

Practice 1:

How is it possible to initialize the PIP board in the system unit to a known state as well as to interface a camera and save the image,from the frame buffer,on disk?

Command descriptions:

a) inifmt 26c 1 1 0 1 0

This command initializes the PIP board in the system unit to a known state

inif(mt)base_addr,mode,speed,class,vid_type,zoom

base_addr: this parameter specifies the offset of the I/O address used.The hexadecimal value (26c)H must be entered if the board has an offset of zero at any of the base addresses.

mode: this parameter tells the software whether the PIP card is strapped to allow for zoom resolution or not

0 --> compatibility

non-zero --> zoom resolution allowed

speed: this parameter specifies the clock speed.

0 --> PIP.512 or PIP.1024 installed

non-zero --> PIP.640-B installed

class: 0-6 specifies the video format

class	American	European	Resolution	PIP type
0	512 x 480	512 x 512	zoom or full	all
1	640 x 480	640 x 512	zoom or full	640 only
2	not allowed	512 x 576	zoom or full	640 or 1024
3	not allowed	640 x 576	zoom or full	640 only
4	1024 x 1024	1024 x 1024	full only	640 or 1024
5	user defined		zoom or full	all
6	user defined		zoom or full	all

vid_type: this parameter specifies wheter the video format is to be a 50Hz or 60Hz format

0 --> American standard video

non-zero --> European

zoom: this parameter specifies whether the format is to use full resolution or zoom resolution

0 --> full resolution

non-zero --> zoom resolution

b) setwindow 0 0 511 511

this command specifies the lower left and upper right corners of the window

setw(indow) x1 y1 x2 y2

x1,x2: these parameters indicate the x coordinates of corners 1 and 2 of the display window

y1,y2: these parameters indicate the y coordinates of corners 1 and 2 of the display window

c) clear 0 7

this command clears the screen by setting one of the unused maps of the input LUT to the current draw index and then taking a snapshot (taking a snapshot in continuous grabbing mode is equivalent to a NO-OP)

clear snap umap

snap: this parameter selects one the 8 input LUT maps to be the active map following the clear operation (decimal values from 0 to 7)

umap: this parameter selects one of the 8 maps of the input LUT to be set to the current draw index (decimal values from 0 to 7)

d) pause

this command is used to halt the interpreter's operation until a key is strick

e) channel 2

this command selects the active video input port

ch(annel) channel

channel: selects the input channel

0 --> selects channel 0
1 --> selects channel 1
2 --> selects channel 2
3 --> internal loopback

f) sync 1

this command is used to select the source of the sync signal

sy(nc) mode

mode: 0 --> internal sync
1 --> external sync

g) sbuf 1

this command is used to select the video output source

sbu(f) mode

mode: 0 --> the output of the input LUT is displayed,keying is disabled
1 --> the output of the frame buffer is displayed,keying is allowed

h) wintordisk 4096 image.bin 506a -1

this command is used to copy the contents of the current display window to disk

wint(odisk)bsize<x> file<s> off<x> seg<x>

bsize: this parameter specifies the size of an intermediate buffer that MS-DOS requires to make the transfer(the optimal buffer size is 4096 bytes)

file<s>: this parameter provides a file name for for the transfer

workbuffer(off<x>): this parameter gives the offset of the intermediate buffer within segment

seg<x>: this parameter specifies the segment that contains the intermediate buffer(use -1)

Practice 2:
How is it possible to copy a disk file to the current display window?

Command description:

winfrdisk 4096 imag3.bin 506a-1
this command is used to copy a disk file to the current display window

winf(rdisk) bsize<x> file<s> workbuffer<x> seg<x>

bsize<x>: this parameter specifies the size of an intermediate buffer that MS-DOS requires to make the transfer (the optimal buffer size is 4096 bytes)

file<s>: this parameter provides a file name for the transfer

workbuffer<x>: this parameter gives the offset of the intermediate buffer within segment

Practice 3:
How is it possible:

to declare and store a macro?
to execute a macro?
to set a pre-defined window?
to read and write the value of a pixel in the frame buffer?

a) to declare and store a macro:

macro macro_number
macro_number --> this parameter is the indentifying number of the macro.It is a number from 0 through 19 and its purpose is to permit future identification of a particular sequence of commands.

savemacro file_name<s> macro_number
this command is used to store a macro in a disk file

file_name: the name of the file in which the macro will be stored

macro_number: a number from 0 through 19 specifying the macro to be saved

b) to execute a macro:

getm(acro) file_name<s> macro_number
this command is used to transfer a macro stored in a file to the interpreter

file_name<s>: it is the name of the file which contains the macro

macro_number: it is a number from 0 through 19 specifying the macro to be transferred.

exe(cute) macro_number
this command is used to execute a sequence of commands defined in a macro

macro_number: this parameter specifies the macro to be executed

c) to set a pre-defined window:

winr(ead) workbuffer<x> seg<x>
this command is used to copy the contents of the current display to a buffer

workbuffer<x>: this parameter specifies the address of the buffer where the contents of the window will be stored

seg<x>: this parameter specifies the segment portion of the memory buffer address

(this command will return the number of bytes that have been transferred)

putfile file_name<s> address<x> count
this command is used to copy the contents of a buffer located in the system's memory into a disk file

file_name<s>: the name of the file into which the conte of the buffer is to be copied

address<x>: this parameter specifies the address of t buffer to be copied

count: this parameter specifies the number of by to be transferred from the buffer to file

cm(emory) address<x> count
this command is used to clear to (00)H the contents of a portion of the interpreter's 16K buffer

address<x>: this parameter specifies the address of t initial memory location to be cleared

count: this parameter specifies how many memo locations will be cleared starting at addr

getf(ile) file_name<s> address<x>
this command is used to copy the contents of a file into a specified buffer(it is too important that the buffer contains enough space to contain the contents of the file)

file_name<s>: the name of the file to be copied

address<x>: this parameter specifies the address of the buffer where the content of the file is to b copied

(this command will return the number of bytes transferred from the file)

```
winw(rite) workbuffer<x> seg<x>
           this command is used to copy the contents of a  buffer
           to the current display window

           workbuffer<x>:    this  parameter contains the address of
                             the  buffer containing the data to be
                             transferred to the  current display window

                  seg<x>:     this parameter specifies the segment
                              portion of the memory buffer address


           (this command will return the number of bytes that has
            been transferred)

gr(id) incx incy
       this  command  is  used to draw a grid using  the  current
       index inside the current window

                  incx:    the number of pixels between the lines
                           of the grid in the x direction

                  incy:    the number of pixels between the lines
                           of the grid in the y direction
```

d) to read and write the value of a pixel in the frame buffer:

Command descriptions:

pixr(ead) x y
this command is used to read the value of a pixel in the frame buffer

x,y: x and y coordinates of the pixel to be read

pixw(rite) x y value
this command is used to write a pixel to the frame buffer

x,y: x and y coordinates of the pixel to be written

value: value to be written to the frame buffer(from 0 to 255,a decimal value)

Practice 4:
How is it possible to calculate a histogram of the grey level in the current window?

Command descriptions:

a) hi(stogram) buffer<x>
this command is used to calculate a histogram of the grey level in the current window

buffer<x>: this parameter specifies the array of 2! elements which is used to store the his! value

b) dh(isto) max<x> x y scale gcol tcol buf<x>
this command draws a histogram in the frame buffer

max<x>: it specifys the count of the most frequer occuring pixel value in the histogram (count shoud be the same as the value by the histo function)

x,y: these parameters are used to specifie tl and y position of the histogram

scale: this parameter specifies the height pixels of the highest bar of the histogr

All other values are drawn relative to this heigt

gcol,tcol: these parameters specifie the index(color) to be used to draw the bar and the labels of the histogram.

buf<x>: address of the buffer where the histogram created by the histo command is stored.

Practice 5:

How is it possible to calculate a histogram of the grey level using differents windows at the same image?

Command description:

recta(ngle) x1 y1 x2 y2

this command is used to draw a rectangle having (x1,y1) and (x2,y2) as opposite corners

x1,x2: this parameter specifies the first and the second x coordinate

y1,y2: this parameter specifies the first and the second y coordinate

To calculate a histogram of the grey levels using different windows at the same time it will be always necessary, first, to use the setwindow command.

Practice 6:

How is it possible to initialize a lookup table map (the use of pseudocolor) as well as how is it possible to combine different functions drives (R-red, G-green, B-blue)?

a) to initialize a lookup table

Command descriptions:

scaling x1 y1 x2 y2 buffer<x>

The function scaling maps the values of Xn onto Yn with X as the index to the LUT and Y as the mapped value.On the basis of this command it is possible to emphasize one part of an image at the expense of another.By making multiple calls to scaling it is possible to break the LUT into several sections.

x1,x2: the first and the last address in buf to be scaled

y1,y2: the first and the last value (the low and the upper end of the scale) to be stored at buf[x1] and buf[x2] respectively

buffer<x>: it is the address of the buffer where the result of the scaling function is to be stored

lutd(define) map color start length buffer<x>

this command,using the values stored in a buffer,initializes a lookup table map.The input LUT is load from the buffer in reverse order with respect to the other LUTs.

map: this parameter specifies which of the 8 maps of any one particular LUT is to be load with the data.

color: this parameter specifies the lookup table which will be affected by the change of data
0 input lookup table
1 blue lookup table
2 green lookup table
3 red lookup table
4 all output lookup table

start: it indicates which of the 256 bytes of the selected map will serve as the starting point to inicialization (from 0 to 255 inclusive)

length: this parameter indicates how many bytes will be rewritten using the scaling function (it must be between 1 and 256 inclusive)

buffer: this parameter contains the location of the buffer containing the values of the initialization

b) to combine different functions drives and to restore the original image

This programe will initialize map 0 of the blue lookup table starting at byte 0 going up to 255 (with the scaling function stored in location (506a)H. In a similar way map 0 becomes now the green LUT and all the blue has now turned to cyan (blue and green together). Once again map 0 becomes now of the red LUT and the initial images of a monitor is now the result of three output signals.

It is very important to observe that the value in the LUTs determine the color intensity while the difference between the three values controls the color.

A white pixel of value 255 is the sum of:

one blue pixel of value 255
one green pixel of value 255
one red pixel of value 255

A black pixel of value 0 is the sum of:

one blue pixel of value 0
one green pixel of value 0
one red pixel of value 0

After the next steps the programe restore the original image because it was used a inverse scaling function and all others three LUTs.

Practice 7:

How is it possible to convolve the image in the current window with a pre-defined kernel?

Convolutions is a special class of image enhancement technique known as spatial filtering.

Spatial filtering is typically used for edge enhancement or noise reduction prior to edge or object detection.

7.1- low pass filtering

Low pass filtering of an image produces an output image in which high spatial frequency components have been attenuated. It is often used as a smoothing operation to remove visual noise.

Command descriptions:

ker3 k1 k2 k3 k4 k5 k6 k7 k8 k9 buffer<x>
this command is used to generate the 3X3 convolution kernel used by the con3 command

k1...k9: these parameters specify the values within the kernel as below

$$k = \begin{bmatrix} k1 & k2 & k3 \\ k4 & k5 & k6 \\ k7 & k8 & k9 \end{bmatrix}$$

buffer<x>: this parameter specifies the storage location for the kernel con(3) source dest buf3<x>

this command convolves the image in the current window in the source workspace and copies the result to an identical window in the destination workspace.It use a kernel previously stored in a pre-defined buffer

$$k = \begin{bmatrix} k1 & k2 & k3 \\ k4 & k5 & k6 \\ k7 & k8 & k9 \end{bmatrix}$$

PA	PB	PC
PD	PE	PF
PG	PH	PI

pixels in the current window

$$PE = \frac{k1PA+k2PB+k3PC \ \ldots \ +k9PI}{(k1+k2+ \ \ldots +k9)}$$

If k1 k2 ... k9 = 0 the division is not performed and con3 takes the absolute value of the result and truncates it to 255

source: this parameter selects the workspace in which the convolution is to be performed

dest: this parameter selects the workspace into which the convoluted image will be copied

buf3: this is the location of the buffer where the kernel is stored.

7.2 High pass filtering

High pass filtering of an image produces an output image in which high spatial frequency components are accentuated.It is often used in the enhancement of edges.

7.3 Laplacian

Laplacian edge enhancement produces an output image in which high spatial frequency components,such as edges,are highy accentuated and low spatial frequency components are sharply attenuated.It is useful for extraction of object edges or boundaries.

Command descriptions

lp1 source dest

lp2 source dest

These commands perform a Laplacian edges detection (horizontal and vertical edge detection at the same time) using the kernels below:

lp1

$$k=\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

lp2

$$k=\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

source: sourse workspace for the operation

dest: destination workspace for the operation

7.4 Sobel filtering

The Sobel filter compares the results of two convolutions to estimate the strength and orientation of edges in the image.

Command descriptions:

sobel source dest
- source: the source workspace
- dest: the destination workspace

7.5 Horizontal and vertical edge detection

Horizontal edge enhancement produces an image in which the horizontal edges are higly accentuated and the vertical edges are sharply attenuated.

Vertical edge enhancement produces an image in which the vertical edges are highly accentuated and the horizontal edges are sharply attenuated.

Command descriptions

ho(rfilter) source dest ve(rfilter)source dest

These commands perform a horizontal and a vertical edge detection transformation on the current window respectively

source: source workspace for the operation

dest: destination workspace

They use a 3X3 convolution with the kernels below:

$$kh=\begin{bmatrix}-2 & -2 & -2\\ 0 & 0 & 0\\ 2 & 2 & 2\end{bmatrix} \qquad kv=\begin{bmatrix}-2 & 0 & 2\\ -2 & 0 & 2\\ -2 & 0 & 2\end{bmatrix}$$

7.6 Pixel averaging transformation

Command descriptions:

average source dest

This command performs a pixel averaging transformation on the current window. It uses a 3X3 convolution with the kernel below:

$$k=\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

source: It specifies the workspace in which the averaging operation is to take place

dest: It specifies the workspace in which the results of the operation will be copied

7.7 Median filter operation

Command descriptions:

me(dian) source dest

This command performs a 3X3 median filter operation on the current window

source: the source workspace

dest: the destination workspace

Practice 8:

How is it possible to use at the same time point and area process algorithms?

Practice 9:

How is it possible to dilate or erode the contents into a pre-defined window?

Command descriptions:

di(late) source dest

This command performs a dilation on the contents of the window in the source workspace. It uses a 3X3 structuring element to perform the dilation

source: is the source workspace for the dilation operation

dest: is the destination workspace for the dilation operation

er(ode) source dest

This command performs an erosion operation on the contents of the window in the source workspace. It uses a 3X3 structuring element to perform the erosion.

source: is the source workspace for erode operation

dest: is the destination workspace for the erode operation

Practice 10:

How is it possible to use zoom?

Command descriptions:

zoom mode
This command selects either full resolution or zoom resolution and allow 4 images to be stored in a quadrant. The position of images within a quadrant is set by the pan and scroll commands.

mode: 0 --> standard video

non zero --> zoomed video

scror(el) offset
This command is used to shift the image relative to its current position on the screen.

Scrolling is done in blocks of sixteen pixels.

offset: this parameter specifies the relative scroll of the window. Both positive and negative values are accepted.

panr(el) offset
This command is used to shift the displayed image relative to its current position on the screen.

Panning is performed in blocks of eight pixels.

offset: this parameter specifies the number of pixels by which the image will move. Both positive and negative values are accepted.

REFERENCES

CESAREO,R.; STORELLI,L.M.F. "A miniature X-ray tomography scanner employing radioisotopic sources",presented at the Int.Conf. on Applications of Physics to Medicine and Biology,Trieste,Italy,Mar 30 - Apr 3,1982.

GRIGER,M.L.;DOI,K."Investigation of basic imaging properties in digital radiography I - Modular transfer function" Medical Physics,vol.11,n 3,p.287 - 294, 1984.

GRIGER,M.L;DOI,K.;FUJITA,H."Investigation of basic imaging properties in digital radiography.7.Noise.Wiener spectra of II-TV digital imaging systems" Medical Physics,vol. 13,n 2,p.131 - 138,Mar/Apr,1986.

HALARICK,R.M; SHANMUGAM,K.;DINSTEIN,I "Textural features for image classification".IEEE Transactions on Systems,Man and Cybernetics,vol.3,n 6,p.610 - 621,Nov.1973.

CRUVINEL,P.E;CESAREO,R;CRESTANA,S;MASCARENHAS,S "X-and Gamma-rays computerized minitomograph scanner for soil science".IEEE Transactions on Instrumentation and meassurement (to be published in october 1990).

HUANG,H.K Elements of digital radiology; a professional handbook and guide.Englewood Cliffs:Prentice-Hall,1987.370p

MATROX-Electronic Systems,Canada.PIP video digitizer board hardware and user manual 289-MH-00 Rev 3.Canada,fev 1988.1v.