INTRODUCTION

The electronic imaging system for which this manual is written is an experimental one. The hardware and software components from which it is comprised have undergone many changes since the inception of the system, and it is probable that, since the technology of image processing is an active and emerging one, there will be more changes to the hardware and the software, perhaps even in the near future. Trying to write a manual for a system that is in such a dynamic developmental stage has been a challenge, indeed, and this the ninth, and I am sure not the last version.

Although an effort has been made to anticipate future changes, it is inevitable that new technologies will be made available that will cause the system to be changing for many years to come. Because the system is experimental, work done on the system will also be experimental, and the user should try to familiarize him or her self in a conceptual rather than a "cook-book" sense. In this way, future changes will be easier to adapt to.

Finally, this manual addresses a dedicated system, and it is the grouping of the hardware and software components that is of concern in shaping the way that the manual is written. With this in mind, the user should be prepared to refer to the user manuals of the individual components which comprise the system. This is especially true of the IBM XT, which is a standard production model, and which uses standard IBM DOS. The DOS and user manuals are available and should be referred to. A brief glossary of commands, including some of the DOS commands are contained in Apendix A of this manual. Apendix B is a trouble shooting chart, and Apendix C is a brief synoptic description of each program, along with the proper syntax for each.

UNIVERSITY OF COLORADO ART IMAGE PROCESSING SYSTEM
-THE HARDWARE, A BRIEF DESCRIPTION-

Before beginning to use the equipment and programs described in this manual, it is recommended that the user read through the manual in its entirety. The second time through, while trying to work with the equipment, everything will make more sense--reading through once, first, will pre-dispose the user to a more through understanding.

SOME PRELIMINARY CAUTIONS

-DO NOT HAVE LIQUIDS ON THE WORK TABLE
-DO NOT USE SYSTEM IF:
    -It is over 80 degrees F.
    -there are thunderstorms (or other fluctuations in power.

SOME PRELIMARY CAUTIONS (cont.)

-DO NOT BLIND CAMERA
  -Start in the stopped down position
  -Use a target without the light source turned on in place of negative or slide in order to focus with a wide camera aperture.

BE CERTAIN THAT YOU ARE AWARE OF ALL OF THESE CAUTIONS BEFORE BEGINNING TO USE THE IMAGE PROCESSING EQUIPMENT.

HARDWARE INVENTORY

-IBM XT model 286 Computer
-Colorado Video Model 491 Video Frame store with Colorado Video model 745 Host Adapter.
-Sony Color Monitor.
-Dage Video Camera.
-Custom made RGB color wheel.
-Cold light source.
-Polaroid copy table with light florescent light sources.

THE IBM XT model 286

It is very important to familiarize yourself with the IBM XT BEFORE you begin to use this image processing system. Accompanying this manual is a short description of the XT, as well as a description of general syntax rules that will enable you to "converse" with the digitizer. (See Apendix A) In order to use the programs that are in the XT, you must be in the "work" mode. This is achieved by entering cd \work after the prompt. The computer will then offer the prompt c: \work>. This prompt will allow you you to enter the commands that execute the programs and their respective functions. It is important to be sure that there is enough memory space to store an image before trying to capture a new one. This is achieved by looking at the directory of work. (see your syntax manual) For a single image, there must be 209920 bytes free. Manipulating images usually requires at least three or four other imges, (masks, constants, merges, etc.), therefore, be sure that you clear as much memory as possible before beginning to work at manipulating the image. This is achieved by discarding images in the work file that are not currently being used. An image is discarded by typing in "erase" then a space, and then the name of the image to be discarded. For example, if you wanted to discard an image called "picture", you would type in, after the prompt, "erase picture", and then push the enter key. By renaming a new image with the name of an old image, the old image is also erased.

The IBM screen functions to give you information about the image that you are working with. In the case of analyzing an image, it gives you numeric values, and in the case of manipulating an image it prompts you with respect to when the machines are working, when they are ready for more commands, whatever problems

are occuring, as well as storing the image and keeping track of it by name in various directories. (Refer to your syntax manual)

THE DIGITIZER

The digitizer in this system is a Colorado Video model 491 frame store, as described above. It has very few moving parts, all of which are for the user to control the contrast of the image, (this is done with "white level" and "black level" control knobs, or for the user to choose which channel of the memory (red, green, or blue) will be in place to receive the image. Attatched to the side of the digitizer, is a small switch, which changes the line between the Sony monitor and the digitizer between the "a" and "b" channels. When digitizing color images, this will be especially important to know about.

THE SONY MONITOR

The Sony color monitor is where the image that is being processed will be displayed. It is connected to the digitizer and to the Dage video camera through a series of video cables. The controls on the front panel, (hidden by a small door) effect which channel on the digitizer are being effected, and whether the line is open from the camera or the digitizer. When the "rgb" button is depressed, there is a "live" signal going into the Monitor from the camera--this signal does not reach the digitizer unless the "line b" button is depressed. While the "line b" channel is being utilized, the image that is being sent to the digitizer can be seen only while the "freeze" button on the digitizer is depressed. Once that button is released, the image which is being displayed on the monitor will no longer be live, and it can then be "captured" by the digitizer.

THE DAGE VIDEO CAMERA

The Dage video camera in this system is designed for surveillance. As a result, it is equipped with a video tube which is sensitive to low light. The f stop on the camera should be closed to the minimum size before turning the camera on. After the camera is turned on, its image can be seen on the Sony monitor only when "line b" is depressed on the monitor, or when the "freeze" button is depressed on the digitizer while the "rgb" button on the monitor is engaged. Focussing, and lining up of the image is best done while the "line b" button is engaged, as that will leave the users hands free to operate the camera and place the objects to be photographed in position.

THE COLOR WHEEL

A wheel containing three filters, a red, a blue, and a green is fitted under the lense of the Dage video camera. Color images can be constructed by filtering red, green, and blue biased images and then displaying them on their respective channels

using the rgb programs listed below. The wheel turns freely, and it is important, when using a filter, to be sure that it is well centered in front of the lense, otherwise a vignetting can occur.

SURGE SWITCHES

There are two safety surge switches which intervene between the building's power source and the power chords of all the equipment in this system. Before operating any of the hardware, these switches must be in the "on" position, and it is very important that after working on the system, these switches be turned off.

The above list of hardware represents the image processing setup for which this brief manual is designed. The video camera sends an image to the Colorado Video Model 491 Frame Store with the Colorado Video Model 745 Host Adapter. This device is capable of converting the image that is put in by the camera into digital information. From here forward, the Colorado Video device will be referred to as the digitizer. The digitizer displays the image on the Sony Monitor either as a live image (what the camera is concurrently looking at), or as a digitized image. As a digitized image, the digitizer is able to send the digitized information to the IBM XT. Furthermore, the digitized image can be quantitatively analyzed and manipulated, and then restored to the magnetic memory as a modified image. The programs were written for the quantitative analysis and manipulation of the image. The following is a description of their dedicated applications. Using the programs that are described below, the user will be able to acquire the basic skills of digitizing and anlysing images, as well as becoming familiar with the vocabulary and routines that will be necessary for using the full potential of this system.

QUICK REFERENCE HELP I
-SUPPLYING POWER TO ALL EQUIPMENT

1. Turn on two main surge switches
   i. One behind digitizer
   ii. One above and behind Camera stand
2. Turn on Computer
   i. Large red togle on right side facing computer
   ii. Computer will "self-check" and give prompt: C:>
3. Turn on digitizer
   i. Large red toggle on front panel
4. Turn on Sony Monitor
   i. Push in type chrome switch on front panel
5. Turn on Dage video camera
   i. Small toggle on top of camera, red light when on.
   ii. Select light source, and use switch on surge panel.
6. Turn on printer only when it is to be used.
   i. pull tray out from bottom left front panel
   ii. pull out on/off knob--wait for "print" light

QUICK REFERENCE HELP II
-DIGITIZING AN IMAGE--MONOCHROMATIC-

1. Make sure that all equipment has power
2. Get into "work" mode
    i. Turn on computer
    ii. wait for completion of "self-check" cycle
    iii. Type in cd \work
    iv. Prompt will now read C: \work>
    v. You are now in the "work" mode
3. place black paper under lens of camera
4. push "line b" button on Sony monitor
5. Toggle between "freeze" button and "man seq" button on digitizer until monitor screen is blank, and red light is over the "g" on the frame buffer.
6. Frame image under camera, making sure to remember that bottom 1.5 inches of monitor will not be in final image.
7. push "rgb" button on monitor
8. push and hold "freeze" button on digitizer.
9. While holding the "freeze" button, adjust contrast of image, then release when ready.
10. Choose name for image
11. Type in save name of image then press return.
12. Transfer your image to your own floppy disk
    i. Format your floppy disk
        a. insert floppy disk into drive a
        b. type in format a:
        c. follow DOS prompts
    ii. type copy name of image a:
13. Clean database by removing your image file
    i. type erase name of image

You have now digitized an image and saved it onto your own floppy disk. in order to place it back into the work file, you must copy it from your floppy disk back onto the work file. The following procedure will accomplish this:

1. type a:
2. type copy name of file c:
3. type in c:

You have now moved your image from your floppy disk back onto the hard drive. If you would like to examine that image on the video screen use the following procedure.

1. type disp name of image

Your image will now be displayed on the screen of the Sony video monitor.

THE PROGRAMS

Typing in the name of a program, and then striking the "enter" key will produce the syntax necessary for the correct entry of the parameters of the program. Until you are used to using the various programs, (and perhaps significantly afterwards!), you may find it quite useful to type in the name of the program to see the correct way of entering in the other information relating to the function you wish to carry out.

The programs relating to image processing with this system can be classified into three types:

- -Image capture--display of captured image
- -Image analysis
- -image manipulation

All of the programs in all of these categories are accessed while in the "work" directory.

CAPTURING AN IMAGE

An image is digitized by the digitizer, and can be stored in the memory of the IBM computer. The freezing of the image as described above is actually the digitizing of the image, and is referred to as a "capture" of the image. The command used to store the image is "save" After turning on the video camera, focussing, composing, and "capturing", or digitizing the image, the save command is entered into the IBM. A name must be given the imge, and recalling the image will be done by referring to that name.

-SAVE The save comand records the image in the memory of the computer. e.g. C: work>SAVE Picture. We have entered the "save" cammand and have named the image "picture". If the work directory is called up now, "picture" will be a file, and it will occupy 209990 bytes.

-DISP The DISP command will take the captured image out of the memory of the computer and display it on the Sony monitor. e.g. C: work>DISP Picture. We have entered the DISP command and have asked the computer to display the digitized image named "Picture" on the Sony monitor.

With these commands, images can be digitized, stored, retrieved, and then displayed. The "Save" and "Disp" commands are the basic access commands for the images, and they are also the commands that save and access any modified images. Refer to the workbook for practice in using these comands.

ANALYZING AN IMAGE

There are two programs designed for analyzing the image:

"HIST" command produces a histograph. A histograph is a list of pixels separated into 7 pixel value ranges. The dark tones have a low pixel value, the light tones a high one. The range is between 0 and 255.

"LOOK" command lists pixel values
specified by a range of x y coordinates--it is similar in that it gives pixel values between 0 and 255, but you can choose the exact location of the pixels you would like to examine.

-HIST The hist command produces a list of ranges of numbers followed by the number of pixels that fall into the 8 pixel range of those numbers. e.g. C: work>HIST Picture will produce a list numbered 0-7, 8-14, etc. Following each number will be a number representing the number of pixels which in the image named "picture" have the pixel values falling within that range.

-LOOK The look command also quantifies the image in terms of pixel values but it is much more specific, as it can identify even a single pixel.
e.g. C: work>LOOK Picture 100 100 100 100 will single out the pixel at the x y coordinates of 100 x 100. (100 pixels from left, and 100 pixels down from top)

e.g. C: work>LOOK Picture 100 100 100 105 will look at 5 pixels, 100 pixels from the left and from 100 to 105 from the top. Practice several of the exercises in the workbook to become comfortable with identifying precise pixel values.

This is also a good time to try overlaying the "grid"* image for identifying specific sections of the image. The grid should always be displayed in the r or b channels. By using "look" to compare high key areas of the image with middle or low tones, you can get a sense for how much to increase or decrease values in specific areas. It will be seen that this type of information becomes vital in manipulating the image. It is important to use the workbook to acquire skill in identifying areas and relating them to pizel values.
MANIPULATING THE IMAGE

It is very important to become familiar with the image analysis programs, and to know the distributions of tones within in image before attempting to effect a change on the image. It is good practice to always make a histogram of your image to refer to this distribution in order to effect meaningful changes. THE WAY THAT THE IMAGE LOOKS ON THE DISPLAY MONITOR IS NOT NECESSARILY THE WAY THAT IT IS STORED IN THE FRAME STORE.

TONE TRANSFORMING PROGRAMS

In black and white image processing, there are three basic programs that effect tone transformation, they are CONS, TRAN, and TRAN2. Associated with these, and used in conjunction with them are PARMS, MERGE, MSKAVE, MSKMAX, AND MSKMIN.

It is important to stress that in this image processing system, the way that transformed images emerge is that a new image is made to either replace or be combined with the old image in order to effect the new image. The old image, (with the old name), will be retained by the processing system until it is removed from the memory. Sometimes, this is all effected in a single complex command to the computer, in which the old image is invoked, an new image is made, and then combined with the old to make a third image. The change of image is not seen, in this case, until the third image is displayed. For this reason, it is good practice, when working with a single image concept, to name all the images with similar prefixes, and then change the name by using numeral or alphabetic endings for consecutive images. For example, if you capture an image of a boat, and decide to call it "boat", perhaps the consecutive, overlayed or added, subtracted, etc. images might be called "boat1", or "boat2", etc. The usefulness of this practice will become more evident as you generate more images. Below is a brief description of these programs, along with some theoretical examples.

-TRAN The TRAN program transforms the given image by re-assigning pixel values, using a three operation mathematical formula. This is done across the full tonal range of the image. The mathematical formula is: the new pixel value = (old pixel value) times a chosen value, divided by a chosen value, and then added to by a chosen value.

By typing in the TRAN command, you will produce its syntax on your screen. The syntax for TRAN is:
tran infile outfile mult div add [min max] [xmin x max ymin ymax]

This means that when you type in the TRAN command, the computer next expects the following information:

Infile - the name of the image you wish to transform, e.g., picture.

outfile - the name you will assign to the newly generaged image, e.g., picture1

mult - a number by which you wish to multiply all pixel values

div - a number by which you want to divide all pixel values.

add - a number which you want to add to all the pixel values after they have been transformed by the multiplication and division process above.

[min max] these paramaters will limit the effect of the equation to within the values between 0 and 255, which you specify. Because they are in brackets, they are optional. If you do not specify, the equation will be carried out over the full range.

[xmin xmax ymin ymax] these parameters will limit the effect of the equation to the locations on an xy coordinate system. Because they are in brackets, they are optional.

Now, let us try inserting some values into a TRAN function. If, for example, "picture" is a negative image, and we wish to transform it to positive, we can do it with the TRAN program. We must first determine what values must be inserted into the formula to convert all the values to their equivalent low tones and high tones. Since we know that the lowest tones on our system are assigned the value 0 and the highest tones are assigned the value 255, we can formulate our equation to invert all of the given tones. If we multiply everything by 1, and then divide by a -1, and then add 255, all the tones will be transformed. What used to be 255 will become 0 and what used to be 0 will become 255, etc. Put in place, our example will now look like this:

```
tran picture picture1 1 -1 255.
```

This will produce a new image, "picture1" by name, that will be the negative of "picture" If "picture" was a negative image to begin with, then "picture1" will be a positive.

If you are not arithmatically or mathematically inclined, there

is a very useful program to help you figure out the values to insert into the TRAN formula. That program is called PARMS. PARMS actually stands for parameters, and it is designed to allow you to input the value changes which you would like to effect, and it will give you the parameters to insert into the TRAN program. The syntax for the PARMS program looks like this:

parms val1 old val1 new val2 old val2 new

The values that must be inserted into this program to plug into the example we had obove, are as follows:

val1 old 0 in "picture"
val1 new 255 in "picture1" (this is what you want the new value to be.
val2 old 255 in "picture"
val2 new 0 in "picture1" (this is what you want the new value to be.

Now our command looks like this:

parms 0 255 255 0

The computer will then makes the calculations necessary, and will produce for you three values. Its response looks like this:

1,-1,255

The next step is to insert these values in the TRAN program, and it will produce the desired effect.

-CONS The CONS command is used to make an "image" of constant pixel values. The resultant display, of course, will be a homogeneous tone on the screen. This homogeneous tone can then be added to or subtracted

from another image, thereby boosting or reducint the tonal value of the image overall, or in specified local or tonal areas. When used overall, this command can be useful in pushing highlights up or shadows down when there is not a good representation of tones at either end of the spectrum.

Used locally, or by tone value limits, this command can have a "solarizing" effect by blocking up or dropping out high or low values. IT IS VERY IMPORTANT TO NOTE THAT THIS PROGRAM MUST BE USED IN CONJUNCTION WITH ONE OF THE MERGE PROGRAMS DESCRIBED BELOW IN ORDER TO BE USEFUL. MAKING A CONS IMAGE WILL ONLY PRODUCE A HOMOGENEOUS TONE THAT CAN BE ADDED OR SUBTRACTED, IN TOTAL OR SELECTIVELY FROM SOME OTHER IMAGE.

The syntax for CONS is:

```
cons outfile pixelvalue

cons - name of program
outfile - what you will call this tone screen
pixelvalue - the intensity of the pixel tone
```

e.g. cons con30 30

This will make a homogeneous tonal screen of the intensity 30 pixelvalue. This can then be added or subtracted from another image to effect a change in that image.

At this point, it is important to become familiar with the merging programs of this image processing system. CONS, for example, is not useful to effect a change in an image, until it can be combined with a heterogenous image. (one that is not just a tone screen)

MERGE The merge command combines two images to create a third image. There are four options for ways in which the merge command will combine the images, they are ADD, SUBTRACT, OR, AND AVERAGE.

The syntax for the MERGE command looks like this:

```
merge infile + infile = outfile [min max] [xmin xmax ymin ymax]
merge infile - infile = outfile [min max] [xmin xmax ymin ymax]
merge infile : infile = outfile [min max] [xmin xmax ymin ymax]
merge infile . infile = outfile [min max] [xmin xmax ymin ymax]
```

merge - name of the command
+,-,:,. - add, subtract, average, and or functions
infile - the name of one image in the merge
infile - the name of another image in the merge
outfile - the name given to the new image being generated
[min max] - limits of pixel values to be effected, (these are optional and need not be used.)
[xmin xmax ymin ymax] - limits of location to be effected. (these are optional and need not be used.)

Using the example of the CONS file from above and the example of the "picture" image, the following is an example of a possible merge command:

```
merge picture1 + CONS30 = picture2
```

This will now produce a new image, called picture2, which will be like picture1, but all of its pixel values will be 30 pixelvalues higher. If we had subtracted the two, the pixelvalues would have been lowered by 30 rather than raised.

TRAN2

The TRAN2 program is designed to change the shape of the contrast curve of the image. It is used in conjunction with information derived from a histogram.

The histogram of an image describes that image in terms of the spread, across the tonal range, of various pixels. A well digitized image will have the tonal range spread evenly from the darkest tones to the lightest tones. The shadows will not drop out, and the highlights will not block up. If, for example, your histogram indicates that the low pixel values are overwhelmingly lower in number than the high tones, it will be evident, just from examining that image, that the image is dark. Likewise, if there is an overwhelmingly higher number of very high pixel values, the image may be blocked up in the highlights.

The curve can be changed by moving the top of the curve from a high or low place towards a more moderate place. For example, if most of the tones in a given image gravitate towards pixel values between 40 and 60, then we can assume that the high tones and the low tones are probably deficient. If we move the point of this curve from the 40-50 range more towards the 120 to 130 pixel value range, the tonal range of the image will be more even.

This is effected by operating on three values. A percentage of change is chosen for three tonal values, and then that percentage is interpolated across the entire tonal range, altering the curve of the tonal range of the image.

Let us now look at the syntax for the TRAN2 command:

-TRAN2 Syntax:

```
tran2 infile outfile pct1 pct2 pct3 x_pct1 x_pct2 x_pct 3
     [min max] [xmin xmax ymin ymax]
```

tran2 is the name of the program
infile is the name of the original image
outfile is the name given to the new image being generated
pct1 pct2 pct3 These are variables entered in percent, and designate how much the pixel values are to be changed. To lower values, one would enter less than 100 percent, to raise values, one would enter more than 100 percent.
x_pct1 x_pct2 x_pct3 These are the actual pixel values to effect the change. x_pct1 would be the lowest value, while x_pct3 would be the highest. The curve would then be defined by the values, according to the change in percent indicated, that are given to these three points in the curve.

For example:

If you have an image that has high contrast--the tones are

weighted around low pixel values and high pixel values, you can change the curve to make a more even toned image by boosting the low pixel values towards the middle, and by lowering the number of high pixel values and also moving them towards the middle. To do this, you would pick three points, a low point, a middle point, and a high point, for example: 0, 128, and 255. These are the values that you would choose for the program to operate on, so they are the second set of values (the x_pct1, x_pct2, x_pct3) that would be entered. The first set of values need to correspond, and also to indicate what degree of change is desired. In this case, we might insert 80 (that is 80%, which will lower the number of low values by 20%), perhaps 130 for the middle value (that is 130%, which will increase the number of middle values, around pixel value 128, by 30%), and 75, for the last value (which is 75%, which will lower the number of high pixel values by 25%)

This example would look like this when it is entered:

```
tran2 picture picture2 80 130 75 0 128 255
```

where "picture" is the original image and "picture2" is the desired result.

I.E., tran2 is the name of the program
Picture is the infile
picture2 is the outfile
80 is the percent to change values around 0
130 is the percent to change values around 128
75 is the percent to change values around 255

It is important to experiment with changing the values that you want the program to operate on. E.g., try changing the 0 to 25, the 128 to 75, the 255 to 195, etc. As the tones are pushed more and more towards the middle, naturally, the contrast is reduced. In the same way, contrast can be increased by pushing the number of pixels in certain value areas towards the high and low pixel value ranges. If it is possible, it is often actually easier to re-digitize an image to make this program work more smoothly. Since this program was primarily designed to even out contrast, i.e., to provide a more substantial middle tone, it is important to have a wide tonal range to begin with.

THE MASKING PROGRAMS

There are three programs to creat masks which can be displayed as new images, combined with the original images using the MERGE programs, or combined with other images. Mask images are softer than the original image, and depending on which one is used, they are able to effect the lines within an image.

The way that a mask image is made, each pixel in the original image is made to operate on specified pixels surrounding it, assigning new values and then creating a second image with those new assigned values. For example: A mask of "picture", using the mskhave program, will take each pixel from "picture", search for the surrounding pixels, either 3,5,7, or 9 of them (the number is specified by the user), and then averages all of those values, i.e., the original pixel, along with the specified number of the other pixels. The value that is the average is then assigned to the place of the original pixel. This is done for each pixel in the image.

The result is a softened image, with the higher number of surrounding pixels chosen for averageing causing a softer image. Therefore, if 3 is chosen, the new imge will be less soft than if a 5 or 7 or 9 is chosen.

The mskmax and mskmin programs work in precisely the same way, but they are able to emphasize lines in the image, for rather than choosing any pixels surrounding each pixel, they specify the lightest or darkest pixels surrounding the original pixel. This has the effect of lightening or darkening the original image, and also of causing the lines within an image to be intensified, towards the darks in the case of a mskmin and towards the lights in the case of mskmax.

EXAMPLES

MSKHAVE the syntzx for mskhave is:

```
mskhave infile outfile masksize [min max]
```

mskhave is the name of the program
infile is the name of the original image
outfile is the name assigned to the resultant image
masksize is the number of pixels to be averaged, these must be 3, 5, 7, or 9.
[min max] is the specified values, in pixel values, to which this operation is to be limited.

e.g. mskhave picture picture1 3

will produce a new image from picture, called picture1 in which each pixel value will be averaged with 3 surrounding pixel values, and the resultant average will become the new pixel value

for that place. The result will be a very similar image called picturel, which will look softer than the original image called picture.

MSKMIN The syntax for mskmin is:

```
mskmin infile outfile masksize [min max]
```

mskmin is the name of the program
infile is the name of the original image
outfile is the name assigned to the new image
masksize is the number of pixels to be averaged, these must be 3,5, 7, or 9.
[min max] is the specified values, in pixel values, to which this operation is to be limited.

In the case of mskmin, rather than averaging to any surrounding pixel values, this program will look for the pixels surrounding the original, but with the lowest values, and then average those values. The result is also that of softening the image, but also, the lines will be exagerated by bunching up values in dark line areas.

MSKMAX works in precisely the same way as mskmin, but rather than averaging the darkest, or lowest pixel values, the highest values are chosen, and therefore, the lines in highlight are exagerated rather than the dark.

All masks have the effect of softening the image. These can be used with the MERGE program to give the user control over the extent of line definition, softness, acutance, etc.

DIGITIZING A COLOR IMAGE

In this system, color images are stored using three channels: the r, (or red) the b (or blue), and the g (or green) channels. Each image must be named separately, and then displayed on its appropriate channel, overlapping the previous images, in order to produce a color image. The following procedure is used to capture a color image:

a. Prepare the system in the same way that you would for capturing a black and white image.
b. push the line b button on the sony monitor
c. Focus, compose, and adjust the lighting on the image.
d. push the rgb button on the sony monitor, make sure that the box to the left of the digitizer is set on "a"
e. Blank out the camera by placing a solid piece of cardboard in front of the lens.
f. toggle "man seq" and "freeze" buttons on the digitizer until the screen is blank on the monitor, and the select memory light lands on g.
g. Place the green filter in position under the lens of the camera.
h. push the freeze button on the digitizer and adjust the white and black levels.
i. save the image
k. push "man seq" button until the select memory light lands on r.
l. place red filter under lens.
m. position toggle on box to left of digitizer on b
n. push freeze button on digitizer and adjust the white and black levels.
o. save the image--be sure to name it differently from the green captured image
p. push "man seq" button on the digitizer until the select memory light lands on b.
q. place blue filter under lens
r. push freeze button on digitizer and adjust the white and black levels.
s. save the image.

To display the image in color, display each image in its corresponding color channel. For example if you had named the red image r1, the blue image b1, and the green image g1, then you would display r1 on the red channel, the b1 on the blue channel, and the g1 on the green channel.

Save syntax for color image: Save rgb a_b_c
Disp syntax for color image: Disp a(r)
Disp b(g)
Disp c(b)

MANIPULATING COLOR IMAGES.

This system uses a red, blue and green color system. Anytime that an adjustment is made to one color, an adjustment must be made to the other two colors. This would make image processing in rgb (red green blue system), very awkward, and so there is a program to convert to another system, known as hls (hue, lightness, and saturation), in which the different parameters are not as interdependent.

The program that is used to do this is hls_rgb. Remember, while manipulating a color image, a total of six images is required, three for the rgb version of the image, and three for the hls version of the image. the syntax for rgb_hls is as follows:

rgb_hls r_infile g_infile b_infile h_outfile l_outfile s_outfile.

Once the hls files are created, all of the black and white manipulations which are possible, that is to say all of the programs in the system, can then be executed on a color image using the following simple procedure.

Treat the blue channel of hls as though it is a black and white image. Do not be disconcerted if the image that you display looks blurred, the blue channel does not focus in the same way as the other color frequencies. Once the adjustments are made, using the hls version of the color image, it is then time to reconvert the hls image back into an rgb image. The changes made to the blue channel will be interpreted as general changes to the color image, for in hls, an adjustment to one channel does not off-set the other two channels. When reconverted to rgb, the changes remain in effect.

To change back from hls to rgb, use the following program.

hls_rgb h_infile l_infile s_infile r_outfile g_outfile b_outfile.

It is a useful practice to rename the rgb outfiles with the same name as the original rgb images, otherwise, you will have nine images in the memory just to adjust one single color image. If the results are not satisfactory, you can go back to the rls mode of the color image and make more corrections. After the image is satisfactory, the memory should be cleared up before attempting to process another image.

```
CONS
CONS      Usage: cons outfile pixelvalue
CONS
CONS      Cons is used to generate a 512x410 image file with all pixels
CONS      values equal to the value specified by the pixelvalue parameter.
CONS      Pixelvalue must be between 0 and 255.
CONS

DISP
DISP      Usage: disp infile [channel]
DISP
DISP      Disp is used to display 512x410 image file.  Disp must be used
DISP      with a Colrado Video Frame Store.  The channel parameter must
DISP      be r, g or b.  When the channel parameter is not specified the
DISP      image is displayed on the green channel.
DISP

GRID
GRID      Usage: grid outfile
GRID
GRID      Grid is used to create an reference grid image file.  Grid lines
GRID      are drawn every 100 pixels.
GRID

HELP
HELP      Usage: help [program]
HELP
HELP      Help displays information about the program specified in the
HELP      program parameter.  Help run without the program parameter
HELP      displays information on all the available programs.
HELP

HIST
HIST      Usage: hist infile [xmin xmax ymin ymax]
HIST
HIST      Hist generates a histogram of an image from an image file.
HIST      A histogram of a subsection of an image is output when the
HIST      xmin xmax ymin ymax parameters are input.
HIST

HLS_RGB
HLS_RGB   Usage: hls_rgb h_infile l_infile s_infile
HLS_RGB                  r_outfile g_outfile b_outfile
HLS_RGB
HLS_RGB   Hls_rgb converts a color image from hls to rgb.  It requires
HLS_RGB   as input three files containing the hls channels of the color
HLS_RGB   image.  It outputs three image file containing the rgb channels
HLS_RGB   of a color image.
HLS_RGB

LOOK
LOOK      Usage: look infile xmin xmax ymin ymax
LOOK
LOOK      Look is used to look at pixel values of an image file.  The
LOOK      location of pixels to be displayed is specified by the user in
```

```
LOOK     device coordinates.  Colorado Video Frame Store coordinates are
LOOK
LOOK     X -->      0 ... 511
LOOK     Y          .
LOOK     |          .
LOOK    \/          .
LOOK               410
LOOK

MERGE
MERGE    Usage: merge infile + infile = outfile [min max][xmin xmax ymin ymax]
MERGE           merge infile - infile = outfile [min max][xmin xmax ymin ymax]
MERGE           merge infile : infile = outfile [min max][xmin xmax ymin ymax]
MERGE           merge infile . infile = outfile [min max][xmin xmax ymin ymax]
MERGE
MERGE    Merge reads two image files, combines them and writes the
MERGE    resulting image to an image file.  Merge combines the two input
MERGE    images in one of four ways.  The + sign is used to add the two
MERGE    input images, the - sign is used to subtract the second input
MERGE    image from the first input image.  The : symbol is used to "or"
MERGE    the two input images; the output image is produced by taking pixel
MERGE    values from the second image when pixel values are in the specified
MERGE    range and by taking pixel values from the first image when pixel
MERGE    values are not in specified range.  The range is specifed by the
MERGE    pixel value parameters min and max.  The . symbol is used to average
MERGE    the two input images.  Subsections of the two input images are
MERGE    merged if the user specifies the xmin xmax ymin ymax arguments.
MERGE

MSKAVE
MSKAVE   Usage: mskave infile outfile masksize [min max]
MSKAVE
MSKAVE   Mskave reads an input image file. For each pixel it averages
MSKAVE   pixel values surrounding it and writes the result to the output
MSKAVE   image file.  The masksize specifies the number of pixels on the
MSKAVE   sides of the matrix to be averaged.  The masksize parameter must
MSKAVE   be 3, 5, 7, or 9.  Mskave takes a long time to execute when the
MSKAVE   larger masksizes are used.
MSKAVE

MSKMAX
MSKMAX   Usage: mskmax infile outfile masksize [min max]
MSKMAX
MSKMAX   Mskmax reads an input image file. For each pixel it finds the maximum
MSKMAX   pixel value surrounding it and writes the result to the output
MSKMAX   image file.  The masksize specifies the number of pixels on the
MSKMAX   sides of the matrix to be used.  The masksize parameter must
MSKMAX   be 3, 5, 7, or 9.  Mskmax takes a long time to execute when the
MSKMAX   larger masksizes are used.
MSKMAX

MSKMIN
MSKMIN   Usage: mskmin infile outfile masksize [min max]
MSKMIN
MSKMIN   Mskmin reads an input image file. For each pixel it finds the minimum
```

```
MSKMIN   pixel values surrounding it and writes the result to the output
MSKMIN   image file.  The masksize specifies the number of pixels on the
MSKMIN   sides of the matrix to be used.  The masksize parameter must
MSKMIN   be 3, 5, 7, or 9.  Mskmin takes a long time to execute when the
MSKMIN   larger masksizes are used.
MSKMIN

PARMS
PARMS    Usage: parms val1_orig val1_trans val2_orig val2_trans
PARMS
PARMS    Parms is used to determine parameters for the program tran.
PARMS    The orginal and transformed values of two pixel values are
PARMS    specifed.  The program output the parmeters to be used in tran
PARMS    to transform an image file.
PARMS

RGB_BW
RGB_BW   Usage: rgb_bw r_infile g_infile b_infile bw_outfile
RGB_BW
RGB_BW   Rgb_bw converts a color image from rgb to a black and white image.
RGB_BW   as input three files containing the rgb channels of the color
RGB_BW   image.  It outputs one image file containing the average of the
RGB_BW   rgb channels.
RGB_BW

RGB_HLS
RGB_HLS  Usage: rgb_hls r_infile g_infile b_infile
RGB_HLS                h_outfile l_outfile s_outfile
RGB_HLS
RGB_HLS  Rgb_hls converts a color image from rgb to hls.  It requires
RGB_HLS  as input three files containing the rgb channels of the color
RGB_HLS  image.  It outputs three image files containing the hls channels.
RGB_HLS

SAVE
SAVE     Usage: save outfile
SAVE
SAVE     Save copies an image from the frame buffer to a disk file.
SAVE     Save is used with a Colrado Video Frame Store.  Save captures
SAVE     an image on the green channel.
SAVE

SCALE
SCALE
SCALE    Usage: scale outfile x_add x_mult x_div y_add y_mult y_div
SCALE
SCALE    Scale generates a grey scale image file from the specified parameters.
SCALE

TRAN
TRAN     Usage: tran infile outfile mult div add
TRAN                [min max] [xmin xmax ymin ymax]
TRAN
TRAN     Tran reads an input image file.  It mathematically transforms each
TRAN     pixel value using the three input parameters, mult, div and add.
```

```
TRAN     The transformation equation is:
TRAN     new pixel value = (((old pixel value) * mult) / div) + add
TRAN     The results are  written to an output image file.
TRAN

TRAN2
TRAN2    Usage: tran2 infile outfile pct1 pct2 pct3 x_pct1 x_pct2 x_pct3
TRAN2                [min max] [xmin xmax ymin ymax]
TRAN2
TRAN2    Tran2 reads an input image file.  It mathematically transforms each
TRAN2    pixel value using the six input parameters.
TRAN2    The three 'pct' parameters specify the amount, in percent, used to
TRAN2    alter pixel values.  The three 'x_pct' parameters specify the pixel
TRAN2    values to operate on.  Linear interpolation is used to transform
TRAN2    all pixel values.  The results are written to the output file.
kTRAN2

TRANS
TRANS    Usage: trans r_infile g_infile b_infile r_outfile g_outfile b_outfile
TRANS                h_add l_pct s_pct [min max] [xmin xmax ymin ymax]
TRANS
TRANS    Trans reads three input files of a color rgb image.  It transforms
TRANS    each pixel value of the three channels.  Hue is altered by using
TRANS    the 'h_add' parameter; hue ranges from 0 to 100.  The value of the
TRANS    'h_add' parameter is added to the hue of the image.  The lightness
TRANS    of the image is changed by the amount, in percent, specified by the
TRANS    'l_pct' parameter.  The 's_pct' parameter is used to change the
TRANS    saturation of the image.  The amount of change in saturation is
TRANS    as a percent.
TRANS

TRANS2
TRANS2   Usage: trans2 r_infile g_infile b_infile r_outfile g_outfile b_outfile
TRANS2               pct1 pct2 pct3 x_pct2  [min max] [xmin xmax ymin ymax]
TRANS2
TRANS2   Trans2 is a color version of trans.  It reads three input files of a
TRANS2   color rgb image.  It transforms the image and writes the results to
TRANS2   to three rgb output files. The parameters are specified like they are
TRANS2   in tran2 except that 'x_pct2' is automatically set to 0 and 'x_pct3'
TRANS2   is automatically set to 255.
TRANS2

ZOOM
ZOOM     Usage: zoom infile outfile origin_x origin_y magnification
ZOOM
ZOOM     Zoom reads an image file, magnifies it by the amount specified
ZOOM     in the magnification parameter and writes the magnified image to
ZOOM     the output file.  The xy origin parameters specifies the location
ZOOM     upper right hand corner of the image.
ZOOM
```